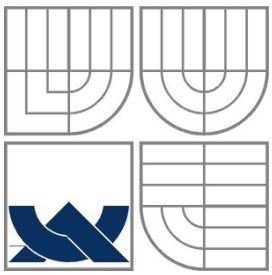
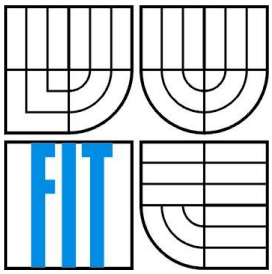


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SÉMANTICKÁ ANOTACE FORMÁTU OPENDOCUMENT

SEMANTIC ANNOTATION OF THE OPENDOCUMENT FORMAT

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JAN NAKLÁDAL
AUTHOR

VEDOUCÍ PRÁCE — ING. MAREK SCHMIDT
SUPERVISOR

BRNO 2010

Zadání

Sémantická anotace formátu OpenDocument

1. Seznamte se s formáty ODF, RDF a modelem metadat ve formátu ODF verze 1.2 a architekturou libovolného software schopného editovat formát OpenDocument Text.
2. Navrhněte nástroj umožňující zobrazení a editaci vhodné podmnožiny funkcionality RDF metadat v textovému obsahu (In Content Metadata).
3. Implementujte tento nástroj jako rozšíření existujícího editoru formátu OpenDocument Text.
4. Demonstrujte použitelnost výsledného systému a diskutujte omezení implementace vůči potenciálním možnostem modelu metadat ODF verze 1.2.

Abstrakt

V této práci je obsažena analýza, návrh a implementace aplikace jako rozšíření pro kancelářský balík OpenOffice.org. Tato aplikace umožňuje zobrazit uživateli metadata formátu RDF nacházející se v textovém obsahu a jejich editaci. Využitím této funkcionality můžeme zobrazit podrobnější informace, o lidech, místech nebo předmětech zmíněných v dokumentu a vložit je přímo do dokumentu samotného. Praktickým využitím jsou například kontaktní informace o osobě zmíněné v textu.

Abstract

This work is about analysis, design and implementation of application as an extension to office suite OpenOffice.org. This application allows users to view metadata RDF format found in text content and editing By using this functionality, we can see specific information about people, places or objects mentioned in the document and paste it directly into the document itself. Practical we can use this for more details information about the person used in text.

Klíčová slova

Metadata, RDF, OpenOffice.org, Add-On, OpenDocument, Java

Keywords

Metadata, RDF, OpenOffice.org, Add-On, OpenDocument, Java

Citace

Jan Nakládal: Sémantická anotace formátu OpenDocument, bakalářská práce, Brno, FIT VUT v Brně, 2010

Sémantická anotace formátu OpenDocument

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Marka Schmidta
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Jan Nakládal
18.05.2010

Poděkování

Děkuji tímto panu Ing. Markovi Schmidtovi za vedení mé odborné práce a odbornou pomoc.
Konzultace s panem Ing. Markem Schmidtem byli pro mě vždy obrovským přínosem a motivací
k další práci.

Obsah

1 Úvod

Dokumenty, jako takové, jsou vytvářeny proto, aby uchovávali a sdělovali informace zamýšlené jejich autorem. Existují nejrůznější formy dokumentů jak fyzické (starodávná rytina, papírová kniha), tak modernější elektronické dokumenty. Dále v této práci budu uvažovat pouze elektronické dokumenty. Všechno tyto nejrůznější druhy dokumentů jsou přizpůsobené k uchovávání určitých informací a to v určité formě. Například informace o rostoucích cenách se dá vyjádřit jak textovou podobou, tak i tabulkou hodnot nebo názorným obrázkem - nejlépe grafem.

To, aby mohl autor předat informace v nejlepší možné podobě, nabízejí dokumenty možnosti jak podobu této informace měnit a přizpůsobovat. Textový dokument nabízí například možnost strukturovat text do sekcí pomocí odstavců. Pro větší přehlednost jsou zde různé nadpisy a možnosti zvýraznění textu. Všechna tato data nazýváme metadata, která se ukládají v dokumentu společně s textem, nesoucím hlavní zamýšlené informace. Osoba, která přijde do styku s takovým dokumentem, dokáže po jeho přečtení vstřebat jak celkovou podobu vzhledu dokumentu, tak prezentovanou informací samotnou.

Pokud stejný dokument zpracovává stroj pomocí nějakého programu, nečiní mu problém získat informace o dokumentu uložená formou metadat, protože tato data jsou pro strojové čtení navržena a uzpůsobena. Naopak stroj není schopen extrahovat samotnou hlavní informaci obsaženou v textu dokumentu, protože tato informace je formulována v podobě vhodné pro lidské vnímání, nikoliv strojové. Logickou snahou je zahrnout tyto informace, kvůli kterým dokument vznikl, do nějaké formy tak, aby tyto informace mohl číst jak člověk, tak stroj. Tou vhodnou formou jsou právě metadata.

Můžeme se ptát k čemu je dobré, aby stroj dokázal získat informace obsažené v dokumentu. Pokud budu chtít danou informaci získat, patřičný dokument, obsahující tuto informaci, si přečtu sám. To je sice pravda, ale nyní se zamysleme nad otázkou, kde ten správný dokument naleznu, když ho nemám a ani nevím, kde se nachází. V současnosti existuje takové množství dokumentů, že není v lidských silách je všechny třídit a nějak katalogizovat pro vyhledávání. A i kdyby to bylo možné, tak budou parametry vyhledávání v čase neměnné, omezené pouze na formu, pro kterou byl katalog navržen. Co je pro člověka neřešitelný úkol, dokáže stroj provést v rozumném čase za předpokladu, že dokáže zjistit informace obsažené v jednotlivých dokumentech.

Využití takovéhoto přístupu je mnohem více. V této práci se zabývám možností využití metadat pro rozšiřující informace, které mají vztah k různým objektům zmíněném v textu. Uvažujme na příkladu: Napíši textový dokument o mé služební cestě do zahraniční. Uvedu zde, že jsem jednal s nějakým panem *X* o věci *Y,* vztahující se k nějakému místu *Z*. Hlavní informací, kterou chci tímto dokumentem zdělit je, jak jednání probíhalo a k jakému jsme dospěli výsledku. Představme si osobu čtenáře, která si čte můj dokument a zjistí z něj výsledek mého jednání, ale tento čtenář by se rád dozvěděl něco více o panu *X* nebo projednávané věci *Y* či místu *Z*. Kdybych uvedl tyto informace přímo do textu, jenom bych odváděl od tématu a čtenář by mohl snadno ztratit hlavní myšlenku. Pokud ale uvedu tyto informace do metadat, například na danou osobu *X* uložím kontakt, k věci *Y* uložím odkaz kde jsou k nalezení další informace nebo k místu *Z* přidám přesné geografické informace jeho polohy. Nebudou tyto informace rušit samotné čtenářovo čtení a naopak pokud jej budou zajímat, stroj je dokáže vyčíst a poskytnout mu je.

Mým cílem v této prací tedy je navrhnout a vytvořit program, který umožní stroji využít metadata v dokumentu způsobem, jež je uveden v předcházejícím příkladu. K tomu využiji některý ze stávajících programů umožňující práci s textovými dokumenty.

2 Specifikace zadání

Mým cílem v této práci je navrhnout a vytvořit aplikaci umožňující uživateli lépe využít možnosti metadat vázaných na obsah, tedy *in content metadat* v dokumentu. V tomto případě formátu *OpenDocument v1.2* a navázání těchto *in content metadat* na uživatelská data uložená ve formátu *RDF metadat* přímo v dokumentu. Budu se zabývat textovými dokumenty, a proto v mém případě půjde o *metadata vázaná na text*. Aplikaci implementuji jako rozšíření stávajícího software pro editaci daného formátu dokumentu. Z tohoto zadání vyvstávají určité požadavky.

2.1 Požadavky na software

Mám-li implementovat aplikaci jako rozšíření již existujícího textového editoru schopného editace dokumentu ve formátu OpenDocument v1.2, musíme zvolit textový editor plně podporující daný formát dokumentu s možnostmi umožňujícími přístup a využití metadat daného dokumentu.

2.2 Požadavky na funkčnost

Můj program musí být schopen využít nabízený přístup k metadatům a zprostředkovat tento přístup uživateli a to přehlednou a do jisté míry intuitivní formou možnosti vytváření, editace a ukládání metadat do dokumentu. Naproti tomu neméně důležitou funkcí, kterou musí aplikace splnit, je zobrazit uživateli metadata obsažená v dokumentu.

3 Analýza

Tato část mé práce je věnována přehledu teoretických znalostí, které jsou potřebné pro vytvoření aplikace, splňující všechny výše zmíněné požadavky kladené na její funkčnost. Vysvětluji zde všechny důležité pojmy a terminologie, které zmiňuji ve zbytku práce.

3.1 Metadata

Metadata můžeme definovat jako *data o datech*. Jedná se o data, nesoucí, doplňující, případně rozšiřující informace o jiných datech (pro přehlednost je nazvěme hlavních datech). Metadata mívají podobu uzpůsobenou k automatizovanému programovému zpracování. Nejsou primárně určena přímo pro lidské čtení, není to vyloučeno, ale aby byl člověk schopen číst metadata přímo musí mít nastudován jejich formát a strukturu.

V úvodu jsem nastínil jedno z možných využití metadat v dokumentu a to jako nositele informací o grafické prezentaci hlavních dat dokumentu. Výhoda strojové čitelnosti metadat a význam, jaký začíná tato vlastnost nabývat, však metadata pozvolna posunují dál a to do pozice být nositelem doplňujících, posléze hlavní dat, obsahující nejdůležitější informace dokumentu. Tento směr je evidentní. Umožní v budoucnosti plně využít strojové síly pro zpracování všech, uživatelem, potřebných informací.

Metadata jsou obecným pojmem a vždy musí existovat nějaký formát, který definuje jejich strukturu, jednotlivé vlastnosti a z toho plynoucí využití. Takových formátů existuje nepřeberné množství obecných nebo zaměřených na konkrétní praktické využití. Nás bude nejvíce zajímat specifický typ metadat nazvaný RDF Metadata, která budou využita ve spolupráci s metadaty vázanými na text pro uložení doplňujících informací pro daný blok textu.

3.2 RDF

RDF je zkratka pro Resource Description Framework, volně přeloženo jako *rámec pro popisu zdrojů*. Jedná se o obecný mechanismus umožňující jednoduchý a jednotný zápis metadat. Byl navržen organizací W3C v roce 1999 a upraven v roce 2004. RDF je vlastně model metadat. Přesněji modelem trojic, *předmět – predikát – objekt.* Tato trojice a vzájemné vztahy mezi jejími uzly jsou navrženy tak, že je možné pomocí nich vyjádřit jakékoliv *tvrzení* o objektu. Uzel může být literál nebo jiný zdroj. Zdrojem se rozumí adresa URI odkazující na prvek, který může být Literál nebo jiný zdroj. Například tvrzení: „Jan Nakládal je autorem této práce“ by se dalo pomocí RDF trojice vyjádřit takto: „tato práce – autor – Jan Nakládal“. Pomocí toho, že objekt může být zároveň v jiné trojici předmětem dokážeme vytvářet celé RDF grafy a popisovat tak složitější struktury.

RDF metadata nejsou závislá na konkrétní implementaci, ale pro zápis RDF metadat se nejčastěji používá xml dokument (RDF/XML), existují ale i další dokumenty používané pro zápis RDF.

RDF/XML je ve skutečnosti pouze rámcem pro popis dalších jazyků, pro lepší přehlednost se proto využívají tzv. *Klasifikační schémata*. Jedná se o soubory definovaných vlastností a oborů hodnot těchto vlastností. Je vytvořeno množství těchto klasifikačních schémat a vždy je tu možnost vytvořit si schéma vlastní. Můžeme si to představit jako typy metadat. Chce-li například uložit informace o nějaké osobě použijeme k tomu například klasifikační schéma **foaf:person**[1]. Pro popis firmy existuje schéma **foaf:organization**[2]. Pro nějakou událost existuje například **cal:vevent[3]**.

Tvrzení: *tato práce – autor – Jan Nakládal* by se dala pomocí RDF/XML zapsat například takto:

```
<?xml version=1.0"?>

<rdf:RDF
      xmlns:rdf="http//www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:si="http://www.w3schools.com/rdf/">

<rdf:Description rdf:about="ibp.odt">
      <si:title>Sémantická anotace formátu OpenDocument
      </si:title>
      <si:author>Jan Nakládal</si:author>
</rdf:Description>

</rdf:RDF>
```

<rdf:RDF – je root element pro RDF/XML soubor. Obsahuje definici jednotlivých namespace použitých v dokumentu. *Klasifikační schéma* je v RDF/XML dokumentu definováno pomocí namespace. Základním klasifikačním schématem je definice samotného RDF, nacházející se na adrese *http//www.w3.org/1999/02/22-rdf-syntax-ns#*, pomocí namespace *rdf*. Druhým klasifikační schéma je ukázkové schéma definující atributy title a author, v příkladu definováno namespace *si*.

<rdf:Description rdf:about="ibp.odt"> - tento zápis určuje, že vnořené elementy se vztahují k *ibp.odt* což je soubor s touto prací.

<si:title> - obsahuje informaci o názvu práce

<si:author> - obsahuje informaci o autorovi práce.

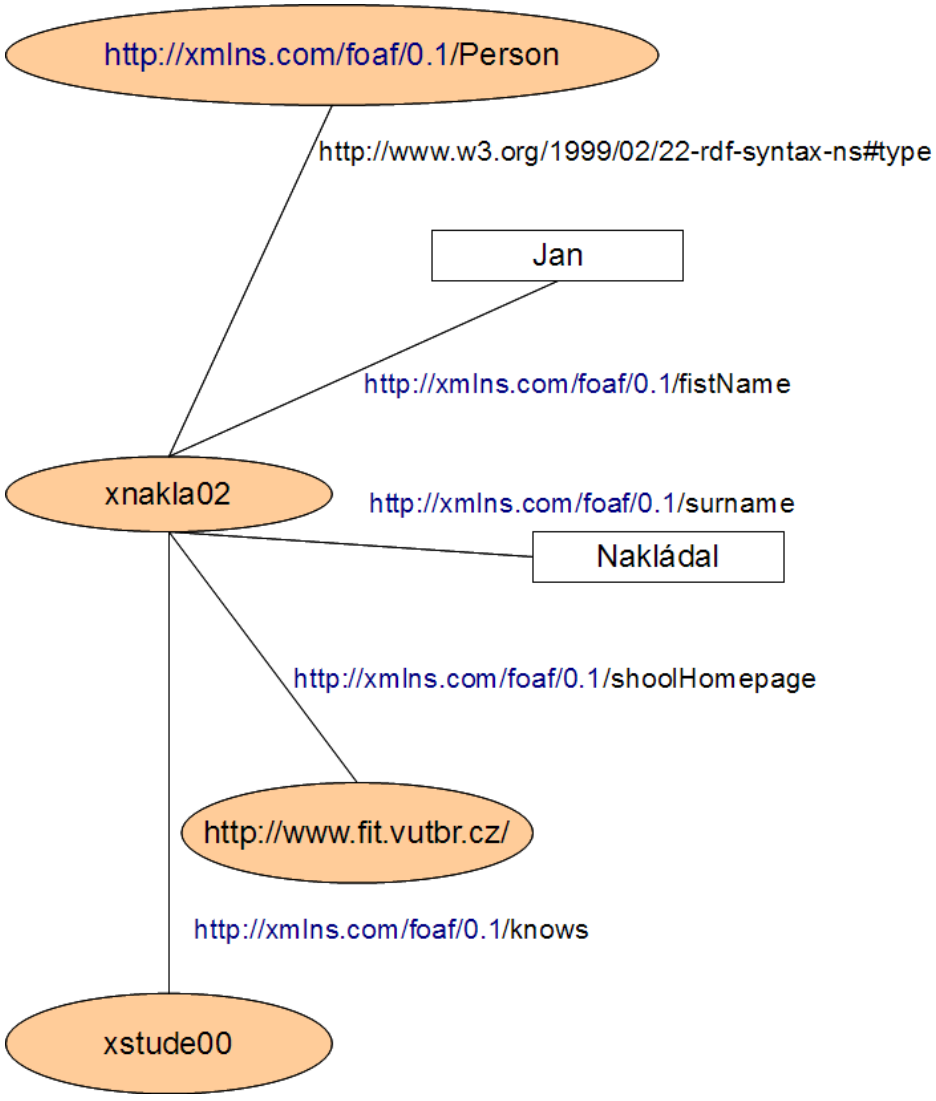
Pro uložení jakýchkoli dat do RDF metadat stačí pouze vytvořit si patřičné klasifikační schéma popisující jednotlivé atributy a obory hodnot těchto atributů. Vytvořené klasifikační schéma zpřístupnit na internetu a při použití se na něj odkázat. Není vždy nezbytně nutné vytvářet si vlastní klasifikační schémata. Dále jsou uvedeny příklady některých již existujících klasifikačních schémat, která lze použít.

3.2.1 Typ metadat *osoba*

Pro uložení informací o osobě do metadat můžeme použít klasifikační schéma **foaf:person**[1]. Jedná se typ nebo také třídu metadat obsahující například:

knows – atribut popisující vzájemný vztah mezi osobami
schoolHomepage - atribut zdroje školy osoby
workHomepage - atribut zdroje zaměstnání osoby
firstName - atribut obsahující křestní jméno
surname - atribut příjmení
familyName - atribut obsahující rodné jméno

Například pokud chci uložit informaci o mém jméně, příjmení, adresu školy a to, že znám studenta xstude00 bude RDF graf vypadat jako na *Ilustrace 1*



Ilustrace 1: Příklad RDF grafu osoby

Ilustrace 1 používá elipsy jsou jednotlivé zdroje jenž mohou být použity v RDF tvrzení na pozici předmětu nebo objektu. Spojnice grafu tedy predikáty jsou definovány v klasifikačních schématech. Obdélníky znázorňují Literály.

RDF/XML by zápis mohl vypadat například takto:

```
<?xml version=1.0"?>

<rdf:RDF
      xmlns:rdf="http//www.w3.org/1999/02/22-rdf-syntax-ns#">

<foaf:Person rdf:about="xnakla02"
      xmlns:foaf="http://xmlns.com/foaf/0.1/">
<foaf:firstName>Jan</foaf:firstName>
<foaf:surname>Nakládal</foaf:surname>
<foaf:schoolHomepage rdf:resource="http://www.fit.vutbr.cz/" />
<foaf:knows
      rdf:resource="http://www.stud.fit.vutbr.cz/~xstude00" />
</foaf:Person>

</rdf:RDF>
```

V uvedeném příkladu si můžeme všimnout, že použité klasifikační schéma lze definovat až v samotném prvku je to dáno možnostmi definování namespace v xml dokumentech. Výhodou definice klasifikačního schématu v root prvku RDF je, že pro celý rdf dokument je klasifikační schéma uvedeno pouze jednou.

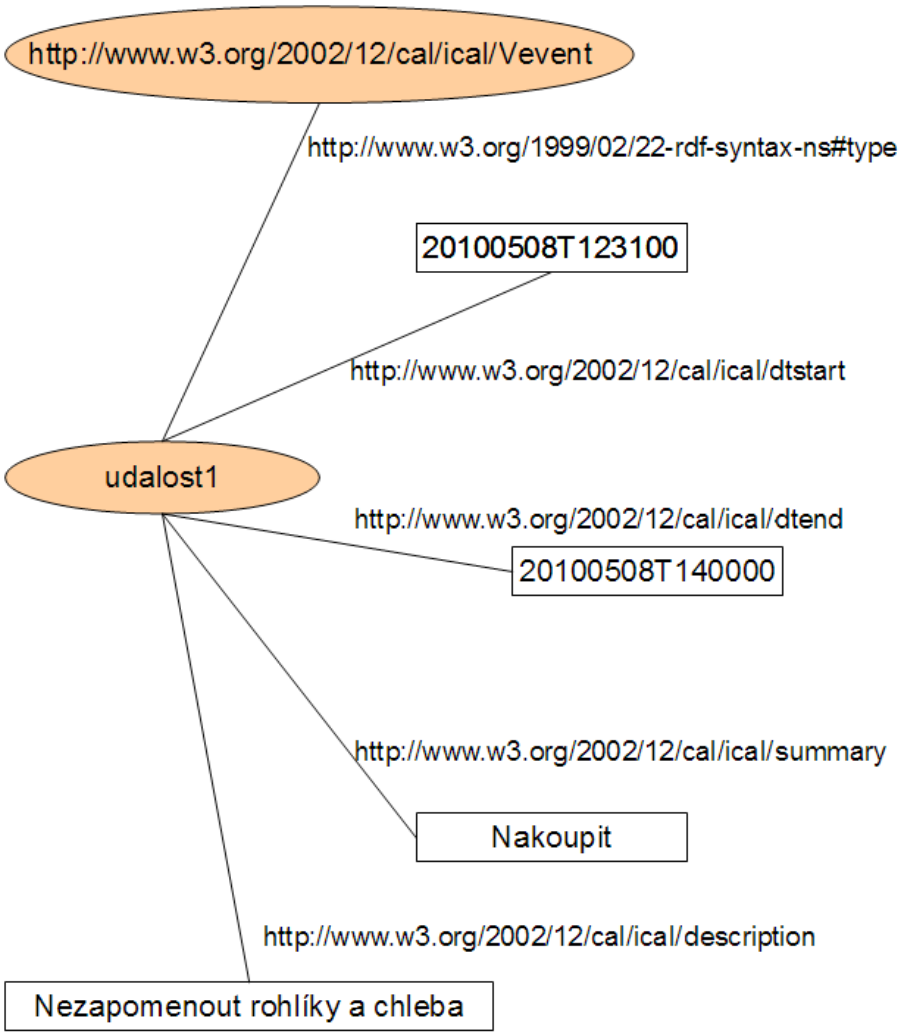
U atributů **foaf:schoolHomepage** a **foaf:knows** je vidět zápis jednoznačné URI zdroje. Kdežto ostatní atributy v příkladu jsou Literály.

3.2.2 Typ metadat *událost*

Pro uložení informací o nějaké události do metadat můžeme použít klasifikační schéma **cal:vevent[3]**. Jedná se typ nebo také třídu metadat obsahující například:

dtstart – datum a čas začátku události
dtend – datum a čas konce události
summary – název nebo stručný popis události
description – rozšiřující popis události

Uvažujme příklad nákupu: dne 8.5.2010 od 12:30 do 14:00 mám jít nakoupit rohlíky a chleba. V RDF grafu bude tato událost znázorněna jako na Ilustrace 2



Ilustrace 2: Příklad RDF grafu události

V *Ilustrace 2* elipsy opět znázorňují jednotlivé zdroje použité jako předmět nebo objekt v tvrzení, Spojení jsou predikáty definované klasifikačními schématy a obdélníky znázorňují literály.

RDF/XML zápis této události vypadá takto:

```
<?xml version=1.0"?>

<rdf:RDF
      xmlns:rdf="http//www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:calc="http://www.w3.org/2002/12/cal/ical/">

<calc:vevent rdf:about="udalost1">
<calc:summary>Nakoupit</calc:summary>
<calc:dtstart>20100508T123100</calc:dtstart>
<calc:dtend>20100508T140000</calc:dtend>
<calc:description>Nezapomenout rohlíky a chleba
</calc:description>
</calc:vevent>

</rdf:RDF>
```

Za povšimnutí stojí formát v jakém je datum a čas události zapsán v atributech **dtstart** a **dtend**. Tento formát není náhodný a je přesně definován v daném klasifikačním schématu[3] pro událost.

3.3 OpenDocument

OpenDocument, znám také pod zkratkou ODF, je otevřený formát pro různé kancelářské dokumenty jako jsou například textové, tabulkové a prezentační. Úplný název je Open Document Format for Office Applications. Byl vytvořen organizací OASIS (Advancing Open Standarts for the information society), a standardizován ve verzi 1.0 v roce 2006 standardem *ISO/IEC 26300*. Jde o formát založený na technologiích xml a gzip.

Kompletní dokument je pomocí gzip zabalený archív obsahující definovanou adresářovou strukturu. Tyto adresáře slouží různým účelům, například pro uložení vkládaných obrázků. Nejdůležitějšími soubory v celém dokumentu jsou manifes.xml obsahující záznam o všech adresářích a souborech v daném dokumentu a content.xml jenž obsahuje samotný hlavní obsah dokumentu včetně formátovacích značek řešených xml elementy typických pro daný typ dokumentu (textový, tabulkový a jiný).

3.3.1 Formát OpenDocument verze 1.2

Verze OpenDocument 1.2 byla vydána v prosinci 2009. Jednou z hlavních novinek této verze je přechod metadat v dokumentu na RDF metadata a s tím souvisejí plná podpora tohoto formátu metadat. Pro zpětnou kompatibilitu zůstává zachován manifest.xml, ale vzniká manifest.rdf, nesoucí stejné data, ale již ve formátu RDF/XML obohacená o další možnosti například uložení informací o jednotlivých RDF grafech obsažených v jednotlivých souborech.

Příklad souboru manifest.rdf z textového dokumentu

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

<ns1:StylesFile rdf:about="styles.xml"
 xmlns:ns1="http://docs.oasis-open.org/ns/office/1.2/meta/odf#"
 />

<ns2:Document rdf:about=""
xmlns:ns2="http://docs.oasis-open.org/ns/office/1.2/meta/pkg#">

 <ns2:hasPart rdf:resource="content.xml"/>
 <ns2:hasPart rdf:resource="styles.xml"/>
</ns2:Document>

<ns3:ContentFile rdf:about="content.xml"
 xmlns:ns3="http://docs.oasis-open.org/ns/office/1.2/meta/odf#"
 />

</rdf:RDF>
```

3.3.2 Uživatelská RDF Metadata v OpenDocument verze 1.2

Důležité elementy jsou obohaceny o podporu vlastnosti id umožnující jedinečné označení elementu. To umožňuje provázat s takovýmito elementy RDF metadata pomocí URI. Jedinečnost URI

v dokumentu řeší IRI, což je speciálně navržená relativní cesta umožnující odkazování se na prvky v rámci balíku daného dokumentu a zároveň splňující požadavky kladené na URI.

Příklad metadat vázaných na text pomocí id atributu v souboru content.xml :

...<text:meta xml:id="id24505">Pavel</text:meta>...

Do manifest.rdf se přidá záznam o uživatelském souboru obsahující metadata:

```
...
<ns1:MetaDataFile
 xmlns:ns1="http://docs.oasis-open.org/ns/office/1.2/meta/odf#"
 rdf:about="data.rdf">
 <rdf:type rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
</ns1:MetaDataFile>
...
```

V tomto případě se jedná o soubor s názvem *data.rdf*, který obsahuje metadata typu *http://xmlns.com/foaf/0.1/#Person*

Do samotného souboru *data.rdf* se přidají metadata vázaná na text:

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

 <ns1:Person xmlns:ns1="http://xmlns.com/foaf/0.1/"
      rdf:about="content.xml#id24505">
  <ns1:name>Pavel</ns1:name>
  <ns1:nick>xstude00</ns1:nick>
  <ns1:schoolHomepage rdf:resource="http://www.fit.vutbr.cz/"/>
  <ns1:surname>Student</ns1:surname>
 </ns1:Person>
</rdf:RDF>
```

Metadata obsahují informace vztahující se k elementu v souboru context.xml s id=24505 v našem příkladu tedy k textu *Pavel*. Všechny neúplné URI jsou automaticky považované za relativní URI a jsou automaticky doplněna pomoci IRI daného balíku dokumentu.

3.4 OpenOffice.org

OpenOffice.org je OpenSource kancelářský balík navržený a vyvíjený celou komunitou členů za účelem bezplatné a plnohodnotné alternativy za konkurenční komerční produkty. Je programován převážně v programovacím jazyce JAVA což velice usnadňuje dosáhnout jeho cíle být multiplatformní.

V současnosti (květen 2010) existuje poslední verze OpenOffice.org 3.2, která používá OpenDocument v.1.2 a jako novinku přidává novou funkcionalitu použití inContentMetadata (metadata vázaná na text). Protože OpenOffice.org je rozšířený, existuje u něj možnost vytváření rozšíření a nově implementuje i funkčnost manipulace s metadaty, kterou nezbytně potřebuji pro vytvoření zadaného programu. Rozhodl jsme se použít kancelářský balík OpenOffice.org pro implementaci mé aplikace.

3.4.1 Rozšířitelnost OpenOffice.org

Aplikace OpenOffice.org byla od samého začátku navrhována a realizována pro široké možnosti pozdější rozšířitelnosti. Důraz, který je kladen na nezávislost OpenOffice.org na operačním systému je přívětivě promítnut i do širokého výběru podporovaných programovacích jazyků, v který lze psát rozšíření této aplikace. Jsou to například C, C++, Java, Python a další.

Pro rozsáhlost toho celého balíku kancelářských aplikací jde OpenOffice.org ještě dále a nabízí čtyři různé typy rozšiřujících aplikací.

3.4.2 Rozšíření Client Application

Ve své podstatě se jedná o standardní klientskou aplikaci typickou dle zvoleného programovacího jazyka, která využívá nabízené možnosti OpenOffice.org za pomoci OpenOffice.org API a UNO komponent. Obojí je popsáno níže. Nejde tedy o rozšíření OpenOffice.org v pravém slova smyslu. Spíše se jedná o rozšíření klientské aplikace o možnosti OpenOffice.org.

3.4.3 Rozšíření Component

Tímto typem rozšíření lze implementovat chybějící nebo upravit stávající možnosti nabízené aplikací OpenOffice.org pomocí UNO komponent. Hlavní výhodou je možnost využití nově implementované funkcionality jak samotnou aplikací OpenOffice.org, tak aplikací využívajících některého z ostatních typů rozšíření.

3.4.4 Rozšíření Calc Add-In

Jde o nejspecializovaněji zaměřené rozšíření celého kancelářského balíku OpenOffice.org, které umožňuje využití speciální funkcionality tabulkové aplikace Calc. Proto také toto rozšíření funguje pouze jako součást této aplikace a je samostatně nespustitelné.

3.4.5 Rozšíření Add-On

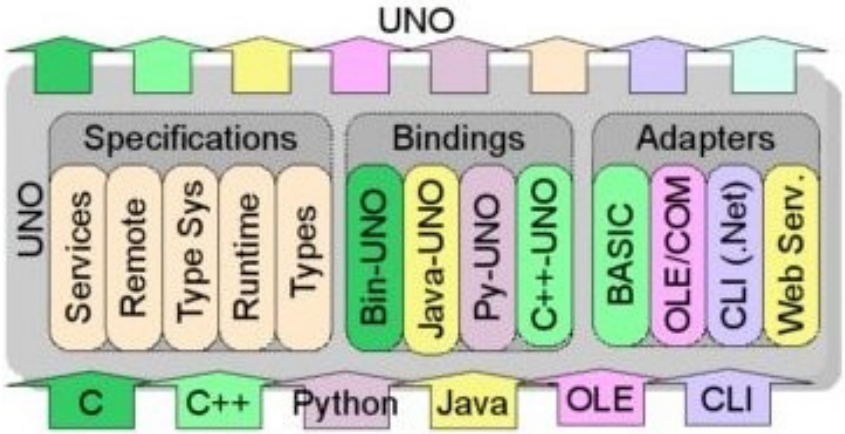
Posledním poskytovaným typem rozšíření je Add-On. Typově je podobné Calc Add-In s tím rozdílem, že lze využít pro všechny aplikace balíku OpenOffice.org. Může být využito pro obecnou funkcionalitu a rozšiřovat celý balík, tedy všechny aplikace. Další možností je zaměřit se na konkrétní specifické funkce jednotlivých aplikací jako například Writer nebo Draw či Base. Jeho možnosti jsou nejideálnější pro vytvoření mojí aplikace. O provázanost se samotným OpenOffice.org se stará třída implementující patřičné rozhraní a pomocí konfiguračního souboru typu xml a jedné registrační třídy sloužící k registraci rozšíření do OpenOffice.org se dá snadno navázat na jakoukoliv aplikaci s daného balíku.

Poté co je rozšíření zaregistrováno se samotná aplikace OpenOffice.org postará o správnou inicializaci celého rozšíření a poskytnutí OpenOffice.org API a UNO pro využití v aplikaci rozšíření. V konfiguračním xml souboru jsou navíc definovány základní nabídky a toolbary rozšíření a v hlavní třídě rozšíření je implementována metoda schopná zachytávat události vyvolané uživatelem v těchto nabídkách. Tím je dokonale vyřešena provázanost s aplikacemi a zdroji OpenOffice.org a na mě již zbývá pouze vytvořit logiku aplikace na navázat ji na tyto *vstupní body*.

3.4.6 OpenOffice.org UNO

UNO je zkratka pro *Universal Network Objects*. Jde o modelovou složku jenž umožňuje komunikaci různých programovacích jazyků se samotnou aplikací OpenOffice.org. Tato komunikace je řešena

podobně jako síťová komunikace jenom v prostředí komunikace mezi jednotlivými procesy. UNO podporuje v současnosti programovací jazyky uvedené na *Ilustrace 3*, ale pro podporu jiného programovacího jazyka stačí pouze binární implementace dané podmnožiny funkcionality UNO v daném programovacím jazyku.



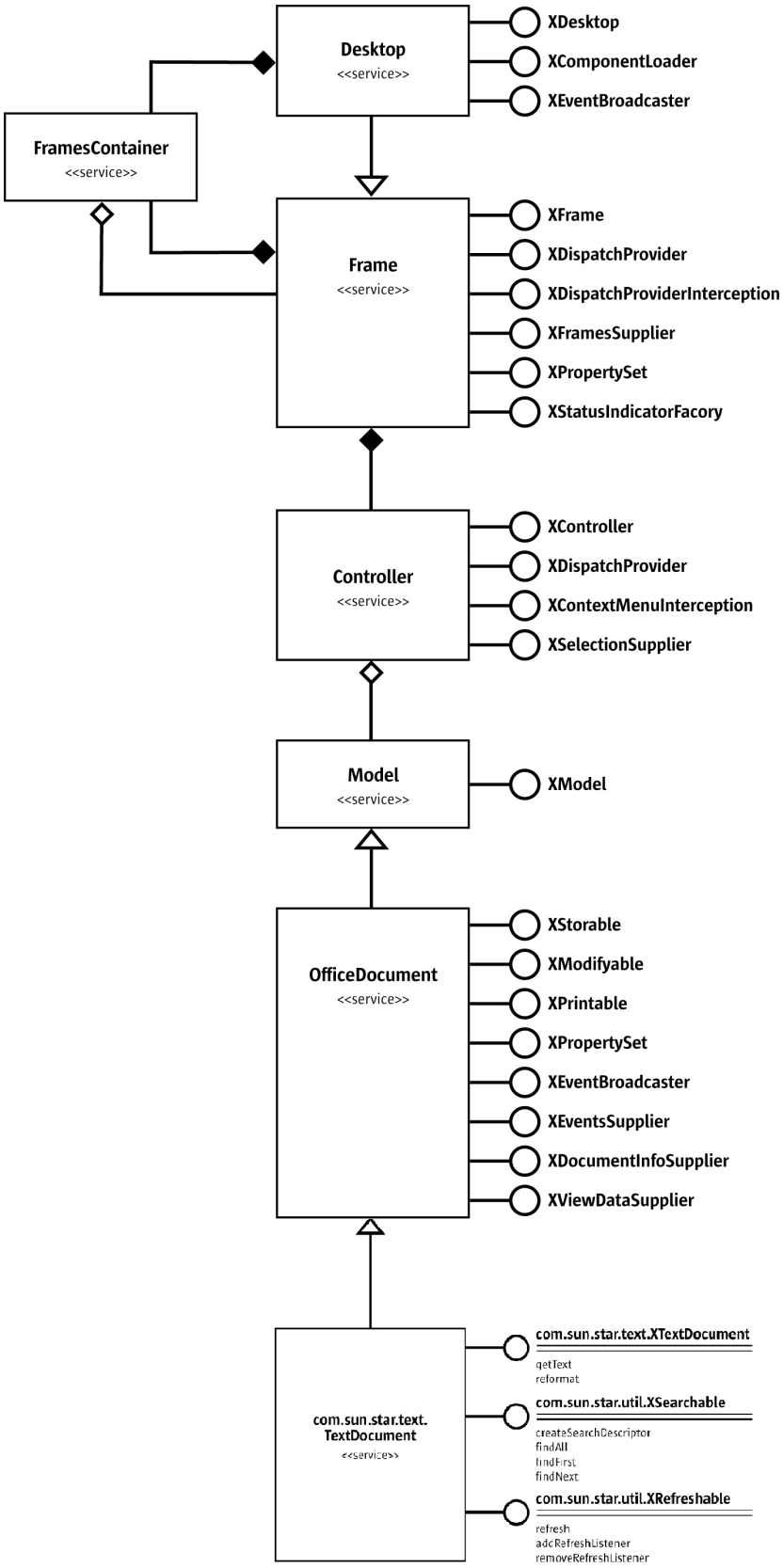
Ilustrace 3: UNO podporované programovací jazyky
Zdrojem Ilustrace 3 je [4]

Pomocí UNO komunikace je aplikace OpenOffice.org poskytovat svou funkcionalitu ostatním aplikacím. UNO existuje i jako aplikace URE (UNO Runtime Environment), která je volně dostupná a samostatně využitelná pro jiné projekty.[5]

Základní spuštění UNO pro přístup k objektům OpenOffice.org je metoda *com.sun.star.comp.helper.Bootstrap.bootstrap* jenž vytvoří instanci rozhraní *com.sun.star.uno.XComponentContext*, který s přístupní samotné objekty OpenOffice.org popsané v následující kapitole.

3.4.7 OpenOffice.org API

Pro využití funkcionalit OpenOffice.org aplikací existuje OpenOffice.org API. Komunikace s OpenOffice.org se provádí pomocí UNO, které má funkci jaké si továrny služeb a rozhraní. Samotné API se skládá z jednotlivých logických objektů. Tyto objekty jsou službami, které poskytují určitou funkcionalitu. Služba může poskytovat rozdílné druhy rozhraní nebo jiné služby. Pro snadnější orientaci a využití mají služby definované rozhraní poskytující konkrétní funkcionalitu. Konkrétní implementace není až tak důležitá, protože žádnou instanci nevytváříme přímo, ale vždy si o ni „řekneme“ UNO, která nám instanci požadované služby nebo rozhraní vytvoří.



Ilustrace 4: Základní Objekty a jejich rozhraní OpenOffice.org
Zdrojem Ilustrace 4 je [6] a [7]

Pro lepší představu uvedu příklad. Pomocí aplikace chci získat text obsažený v aktuálně otevřeném dokumentu. Jako první získám základní rozhraní *com.sun.star.uno.XComponentContext*, které mi poskytne základní rozhraní schopné vytvářet komponenty (taková továrna) *com.sun.star.lang.XMultiComponentFactory*. Pomocí tohoto rozhraní si vytvořím instanci služby Desktop. Se službou samotnou nemohu nic udělat, musím tedy získat její rozhraní *com.sun.star.frame.XDesktop*. Toto rozhraní obsahuje metodu jenž mi poskytne aktuální *com.sun.star.frame.XFrame*. Jde o rámec aktuálně používaného dokumentu. Pokud by je více současně otevřených dokumentů dokáži zde získat jednotlivé jejich rámce a pracovat s nimi.[8]

Rámec mi poskytne přístup k *com.sun.star.frame.XController*, pomocí kterého získám rozhraní *com.sun.star.frame.XModel* aktuálního dokumentu. Služba tohoto rozhraní tedy služba Model obsahuje služby OfficeDocument a *com.sun.star.text.TextDocument*, která má rozhraní *com.sun.star.textXTextDocument*. Pomocí metody tohoto rozhraní získám rozhraní *com.sun.star.text.XText*, které má metodu getString, pomocí které získám text obsažený v dokumentu. [9]

Tento na první pohled složitý postup má určité výhody. Množství rozhraní poskytovaných různými službami, které jsem použil při získávání textu dokumentu lze použít pro nejrůznější úpravy dokumentu samotného. Důležitou vlastností je, že v celém příkladu jsem vytvořil pouze jedinou instanci a to instanci služby Desktop. Je důležité si uvědomit, že všechny dále použité rozhraní jsou získané právě z této instance a tedy přímo jí ovlivňují.

Například rozhraní *com.sun.star.text.XText* mi poskytuje rozhraní *com.sun.star.text.XTextCursor* pomocí, kterého se mohu pohybovat v textu. XTextCursor mi poskytuje rozhraní *com.sun.star.text.XSentenseCursor* a *com.sun.star.text.XWordCursor*. Pomocí XSenteceCursor se posunu na druhou větu v textu a použitím XWordCursor se posunu na třetí slovo v této větě. I když instance těchto rozhraní použiji nezávisle na sobě tyto rozhraní ovlivňují společný objekt. [10]

3.4.8 OpenOffice.org podpora RDF a metadat vázaných na text

OpenOffice.org od verze 3.2 podporuje RDF metadata v OpenDocument verze 1.2. V OpenOffice.org API vznikl *com.sun.star.rdf* balík obsahující služby a rozhraní implementující základní funkce potřebné pro práci s metadaty. Tyto funkce pracují s jakýmikoliv RDF metadaty v dokumentu nejsou tedy omezeny na metadata vázaná na text, ale lze je k tomuto účelu samozřejmě využít.

OpenOffice.org ukládá všechna metadata do jednoho repozitory. S tímto repozitory se dá pracovat pomocí rozhraní *com.sun.star.rdf.XDocumentRepozitory*. OpenOffice.org dokáže pracovat jak s tvrzeními (získanými s repozitory pomocí metod rozhraní *XDocumentRepozitory*), tak s jednotlivými RDF Grafy určenými jedinečnou URI adresou pomocí rozhraní *com.sun.start.rdf.XNamedGraph*. Do těchto grafů se dají přidávat a odebírat tvrzení (tedy trojice).

Všechny objekty dokumentu schopné obsahovat nějaké metadata vázané na text mají rozhraní *com.sun.start.rdf.XMetadatable*, v současnosti se jedná především o objekty odstavce, záložky, nebo speciálního objektu <*text:meta*> určeného přímo pro metadata vázaná na text. Pro vložení takovýchto metadata musíme vytvořit novou instanci objektu *com.sun.star.text.InContentMetadata*, a tento objekt přidat do contentu pomocí *com.sun.star.text.XText* rozhraní a jeho metody insertTextContent. V parametru této metody je rozhraní *com.sun.star.text.XTextRange*, které vymezuje text na který budou metadata vázána.[11]

3.4.9 Vývojové Prostředí

Jedním z možných vývojových prostředí vhodných pro implementaci rozšíření OpenOffice.org je Netbeans. Umožňuje vyvíjet aplikace ve více programovacích jazycích. Navíc jako jednu z jeho největších předností považuji jeho oficiální podporu ze strany komunity vývojářů OpenOffice.org.

Existuje modul pro Netbeans, který umožňuje vytváření přímo aplikací všech typů rozšíření OpenOffice.org. Programovacím jazykem těchto rozšíření v daném modulu je Java.

Modul nabízí navíc možnost automatického sestavení kompletního rozšíření do standardní podoby, v které je možné rozšíření šířit a přidat do jakéhokoliv OpenOffice.org. Asi největší výhodou celého modulu je provázání vývojového prostředí Netbeans s aplikací OpenOffice.org, čímž poskytuje možnost plnohodnotného debugování vytvářené aplikace za běhu přímo v OpenOffice.org samotném.

4 Návrh a implementace aplikace

V této kapitole se zabývám návrhem samotné aplikace, její funkcionalitou, logickým uspořádáním s ohledem na možnost snadného budoucího rozšíření o další funkcionalitu. Na konci kapitoly jsou řešeny problémy vzniklé při samotné implementaci aplikace.

Na začátek návrhu je dobré shrnout co již přímo nebo nepřímo vyplývá z předchozího textu. Při získávání informací o aplikaci OpenOffice.org a jejích možnostech týkajících se funkcionality metadat jsem došel k závěru, že nejlepší možností je implementovat moji aplikaci jako OpenOffice.org rozšíření typu Add-On a jako implementační jazyk tohoto rozšíření jsem zvolil programovací jazyk JAVA. Výhodou toho programovacího jazyka je podpora plně objektově zaměřeného programování.

Ze zadání vyplývá, že aplikace by měla splnit následující funkčnost:

1. Přidání nových metadat vázaných na text do dokumentu
2. Pozdější úpravu neboli editaci těchto metadat vázaných na text
3. Zobrazení těchto metadat vázaných na text uživateli
4. Odstranění aplikací přidaných metadat vázaných na text

Pro zjednodušení práce uživatele jsem se rozhodl bod 2 a 3 sjednotit v jeden, protože si myslím, že je zbytečné uživateli nejprve zobrazit metadata obsažená v dokumentu a až následně přes nějakou nabídku nebo tlačítko mu umožnit v podstatě stejném okně tyto metadata editovat a ukládat.

4.1 Add-On aplikace

Vstupním bodem mojí aplikace bude hlavní třída Add-On rozšíření, která se postará o vytvoření vstupních bodů (více *3.4.5Rozšíření Add-On)* do mé aplikace z OpenOffice.org a provede inicializaci mých hlavních tříd, které předá přístupové body do OpenOffice.org. Jako další tato třída zachytává interakce uživatele s definovanými vstupními body a předá tyto požadavky dál mé aplikaci, která na tyto interakce reaguje provedením příslušné akce.

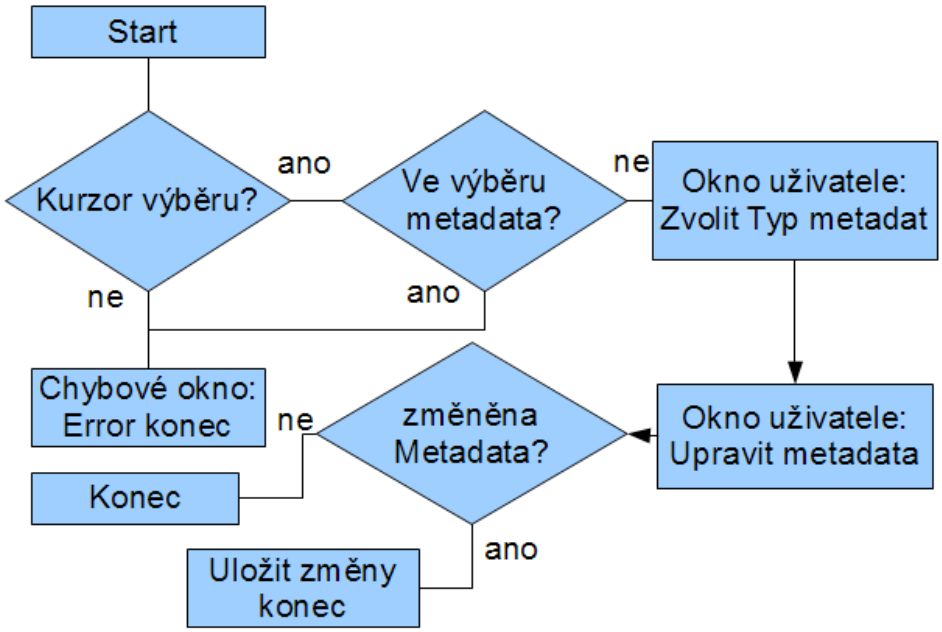
4.2 Životní cyklus aplikace

Protože o samotné spuštění a naslouchání na vstupních bodech se stará samotné rozšíření, stačí, když má aplikace bude navazovat své akce přímo na tyto vstupní body. Protože teoreticky může OpenOffice.org v průběhu času vytvořit více instancí rozšíření je dobré, aby základní třídy byli singleton.

4.2.1 Vstupní bod *přidání* nových metadat vázaných na text

Přidání metadat do dokumentu se na první pohled jeví jako jednoduchá úloha, která se skládá z jednotlivých kroků. Prvním krokem je získání informací o fyzickém textu ke kterému uživatel chce, aby se nová metadata vztahovaly. Těmi nejdůležitějšími informacemi jsou přesná pozice textu v dokumentu, rozsah neboli velikost vybraného textu samotného a existence jeho návaznosti na nějaká metadata.

Vyvstává důležitá otázka, jakým způsobem dokáže uživatel tyto informace poskytnout a jakým způsobem je nejlépe získat? Řešením je použití kurzoru, který má možnost uživatel ovlivnit a označit ním zvolený text před samotným spuštěním akce. Druhým krokem je získat od uživatele informaci o tom jaký typ metadat chce k danému textu přidat. To aplikace provede pomocí dialogového okna v němž uživatel typ metadat zvolí. Třetím krokem je provázání samotného textu a metadat zvoleného typu přidaného do RDF grafu. Následujícím krokem je umožnění uživateli vložit samotná metadata. Tento krok je stejný jako Editace metadat a je popsán níže v *4.2.2Vstupní bod editace metadat*



Ilustrace 5: Cyklus přidání nových metadat

Ilustrace 5 názorně popisuje cyklus přidání nových metadat vázaných na text. Nejdříve se ověří jestli kurzor obsahuje text, tedy je-li vybrán alespoň jeden znak. Pokud ne znamená to, že kurzor se pouze nachází někde v textu, ale žádný neoznačuje. Jde o přidání metadat vázaných na text, a proto je vhodné, aby text na který tyto metadata vážeme nebyla prázdná množina, ale alespoň slovu nebo znak. Z tohoto důvodu je tento stav ošetřen chybovou hláškou uživateli. Pokud kurzor označuje nějaký text jedná se tedy o kurzor výběru, je důležité ověřit, že se k nějakému znaku v tomto výběru nenachází přidružená metadata. Pokud by i jediný znak textu měl přidruženo více metadat nejednalo by se o neřešitelnou situaci, ale jde o určitou komplikaci, pro kterou existuje pouze malá skupina případů pro kterou existuje možnost takovéto vlastnosti opravdu využít.

Reálně by mohli nastat dva případy. Prvním a jednodušším je, že překrývající se metadata jsou ve vzájemném vztahu, kdy dané metadata jsou určitou podmnožinou dalších označených metadata. Druhým příkladem je možnost, že metadata jsou nezávislá a v takovém okamžiku by šlo o křížení objektů v xml dokumentu.

V současnosti vztahují-li se k výběru již nějaká metadata tuto skutečnost aplikace pouze oznámí uživateli a zabrání vytvoření další metadat pro daný výběr, ale bylo by možné zde aplikaci upravit a rozšířit, aby podporovala více typů metadat vztahujících se ke společnému textu obsaženém v dokumentu.

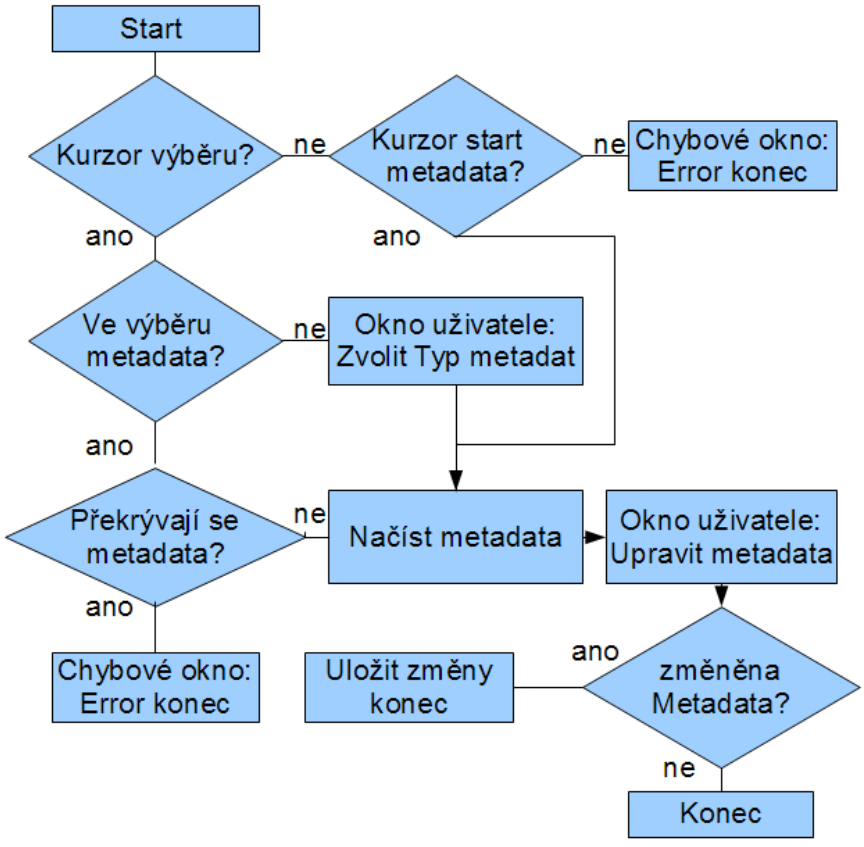
Pokud výběr neobsahuje žádná metadata je uživateli zobrazeno dialogové okno, ve kterém je požádán o zvolení typu metadat, které se mají k danému vybranému textu v dokumentu vztahovat. Poté co uživatel tento výběr provede následuje vytvoření patřičného typu metadat v RDF grafu a jeho provázání s vybraným textem. Následujícím krokem je zobrazení editačního okna řízeného daným typem metadat. Následný postup je shodný s postupem editace metadat popsaným níže. Pokud

uživatel editační okno zruší, zůstanou uložena pouze metadata informujících o typu metadat pro daný označený text.

4.2.2 Vstupní bod *editace* metadat

Tato úloha umožňuje zobrazení uložených metadat vztahujících se k nějakému textu v dokumentu a jejich editace uživatelem. Prvním krokem je zvolení metadat, které chce uživatel zobrazit. Tuto volbu uživatel provede kurzorem, kterým označí text, ke kterému se vztahují nějaká metadata. Pro zjednodušení lze akceptovat i kurzor, který se pouze nachází v textu s návazností na metadata.

Druhým krokem je zjištění typu metadat z již konkrétních metadat, které se označenému textu vztahují. Tato informace bude uložena společně s metadaty. Tvrzení obsahující tuto informaci bude mít podobu trojice: **typ metadat - http//www.w3.org/1999/02/22-rdf-syntax-ns#type – URI oblasti vázaných metadat**. Podle tohoto typu se zobrazí uživateli patřičné editační okno v kterém uvidí uložená metadata. Tyto metadata může uživatel v tomto editačním okně libovolně měnit a následné změny uložit nebo zrušit. Prosté smazání hodnoty a uložení této změny rovná se odstranění těchto metadat z dokumentu. Tímto způsobem lze smazat jakákoliv metadata uživatelem vložená do dokumentu, ale nelze takto provést smazání návaznosti textu na zvolený typ metadat. K tomuto účelu slouží úloha odstranění metadat popsaná v následující kapitole.



Ilustrace 6: Cyklus editace metadat

Ilustrace 6 popisuje cyklus zobrazeni a editace metadat. V první řadě se ověří jestli kurzor jenž má označovat text vztahující se editovaným metadatům je kurzor výběru nebo pouze kurzor v textu. Pokud se jedná o kurzor výběru jsou nalezeny všechny metadata vztahující se k takto vybranému textu. Je-li nalezeno vícero typů metadat je to oznámeno uživateli. Zde by se dala aplikace rozšířit

například o dialogové okno umožňující uživateli vybrat, která metadata chce ve skutečnosti editovat. Nejsou-li nalezena žádná metadata vztahující se k danému výběru je zavolána úloha přidání metadat.

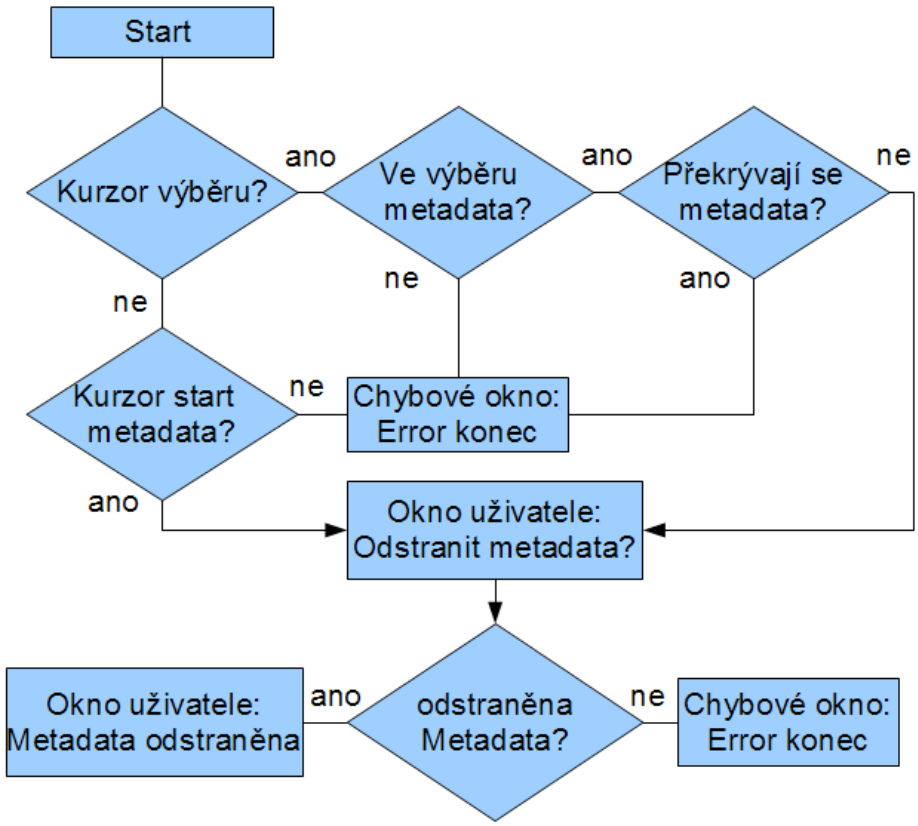
Pokud je jednoznačně vybrán typ metadat je provedeno jejich načtení do editačního okna. Stejným způsobem jsou nalezena metadata pokud se jedná pouze o kurzor do textu.

Následuje zobrazení editačního okna uživateli, kde může provést úpravu hodnot. Změny může a nemusí uložit do dokumentu. Zvolením možnosti uložit změny se provede porovnání nových hodnot s aktuálně uloženými v RDF grafu a všechny změněné hodnoty se postupně ukládají, když dojde k chybě ukládání určité hodnoty z nějakého důvodu, například špatně zadaná URI je to oznámeno informačním oknem uživateli.

4.2.3 Vstupní bod *odstranění* metadat

V předchozí kapitole je zmíněno, že k odstranění metadat stačí prosté smazání jejich hodnot. Ovšem pokud chceme odstranit celou skupinu metadat vztahující se k nějakému textu musíme použít tuto funkci.

V prvním kroku se provede zvolení metadat k odstranění. Metoda získání konkrétní skupiny metadat je v principu stejná jako při editaci metadat. V druhém kroku se odstraní všechna uživatelem uložená metadata z dané skupiny. V posledním kroku se odstraní vazba typu metadat k danému textu.



Ilustrace 7: Cyklus odstranění metadat

Ilustrace 7 popisuje cyklus odstranění metadat z dokumentu. Nejdříve se nalezne odpovídající skupina metadat vybraná uživatelem podle kurzoru. Podmínky tohoto výběru jsou ošetřeny stejně jako předchozího bodu editace metadat. Je-li skupina metadat nalezena, je uživateli zobrazeno dialogové okno pro potvrzení uživatelem, že metadata mají být opravdu odstraněna. Toto okna je zde z důvodu ochrany proti nechtěnému odstraní důležitých metadat. Jakmile uživatel potvrdí své rozhodnutí odstranit metadata, jsou postupně odstraněna nejprve uživatelem zadaná a poté typ skupiny metadat a jeho provázání na text v dokumentu.

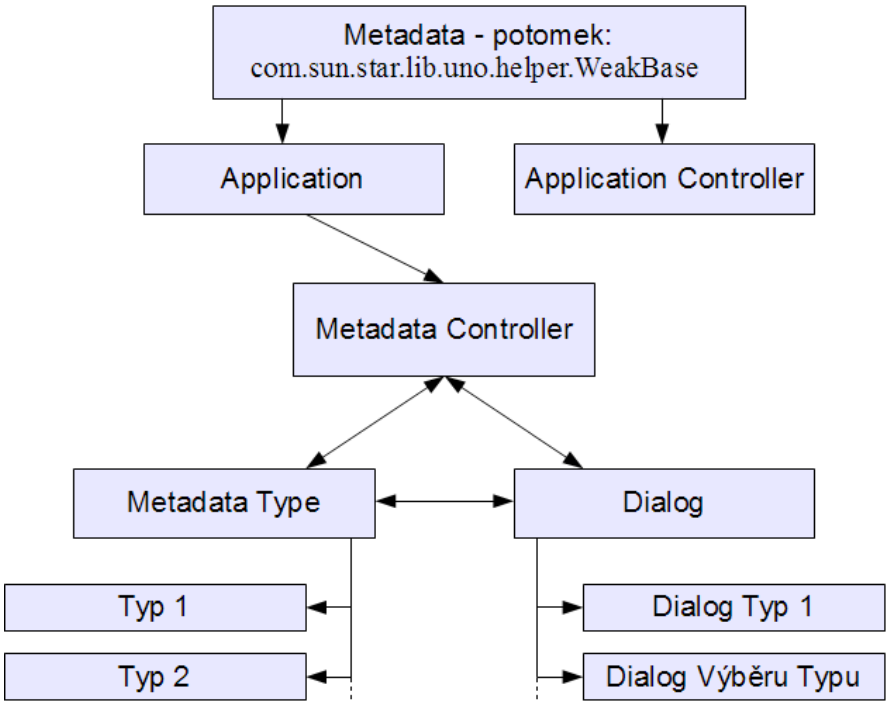
Zde byla aplikace rozšířena, pokud se uživatelská metadata nepodaří všechny odstranit. Je o této skutečnosti uživatel informován, metadata obsahující typ skupiny a provázanost na text v dokumentu se úmyslně neodstraňují, aby mohl uživatel editovat zbylá metadata, která se nepodařilo smazat. Celá tato změna má za následek, že pokud se nepodaří nějaká metadata smazat, má uživatel možnost si je zobrazit v editačním okně a popřípadě je smazat ručně, a poté znovu vyvolat úlohu odstranit metadata, která odstraní zbylé vazby.

4.3 Objektový model

Pro přehlednější a efektivnější programování samotné aplikace jsem si vytvořil objektový model (více *.Ilustrace 8)* na kterém je znázorněno základní hierarchické rozdělení jednotlivých tříd, jejichž konkrétnější informace jsou uvedeny níže.

Základní třídou a hlavním vstupním bodem aplikace je třída Metadata, která je potomkem OpenOffice.org třídy *com.sun.star.lib.uno.helper.WeakBase*. Toto je nezbytné a dané vlastnostmi rozšíření typu Add-On. Zároveň tato třída vytvoří instance tříd Application a ApplicationController, které jsou hlavními třídami mojí aplikace, protože instancí třídy Metadata může být vytvořeno více jsou třídy Application a ApplicationController navrženy jako singleton aby nedocházelo ke kolizím při přístupu ke stejným metadatům ve stejném čase.

Třída Application vytváří třídu MetadataController, která pracuje s metadaty a podle typu úlohy a daných metadat vytváří pomocné třídy odvozené od Abstraktních tříd MetadataType a Dialog. Třídy Typ 1 a Typ 2 jsou pouze pro názornou demonstrací a ve skutečné aplikaci jsou nahrazeny třídami které implementují konkrétní typy metadat. Třídy odvozené od abstraktní třídy Dialog představují různé dialogové okna pro komunikaci s uživatelem.



Ilustrace 8: Objektový model

4.3.1 Třída Metadata

Tato třída slouží jako hlavní bod aplikace, zde se vytvářejí vstupní body a jejich následná obsluha. OpenOffice.org přímo komunikuje s touto třídou a získává od ní nezbytná data o aplikaci rozšířeni například jaké metody implementuje, na jaké akce reaguje.

4.3.2 Třída Application

Tato třída byla navržena jako základní třída celé aplikace, ve smyslu logické struktury. Obsahuje pouze takové metody, jaké vstupní body má celá aplikace. V současnosti jsou to přidání, editace a odebrání metadat. Pro splnění těchto úkolů je využívána třída MetadataController. Rozšíření aplikace o další obecně jakoukoliv funkcionalitu je tak možné pouhým přidáním další metody do této třídy.

4.3.3 Třída ApplicationController

Jedna z nejdůležitějších tříd aplikace. Vytváří ji už třída Metadata. Tato třída má za úkol zprostředkovat služby a rozhraní OpenOffice.org. Instance této třídy je předávána do každé třídy aplikace již v konstruktoru a žádná třída aplikace se bez nástrojů této třídy neobejde.

4.3.4 Třída MetadataController

Jde o třídu, která spravuje Metadata repozitory. Jsou zde základní metody umožňující přiřazení konkrétních skupin metadat ke konkretní části textu v dokumentu včetně s prostředkování uživateli výběr typu , a naopak odstranění těchto vazeb. Dále metody, které podle typu skupiny metadat vytvoří konkrétního potomka třídy MetadataType a předá mu metadata, z kterými vytvořená třída dále pracuje.

4.3.5 Třída MetadataType

V této třídě se provádí operace s uživatelskými metadaty, uložení, odstranění hodnot. Protože pro každý typ skupiny metadat se bude tato třída lišit v počtu a typu hodnot s kterými pracuje, je základní funkcionalita třídy vytvořena v Abstraktní třídě a pro každý typ metadat s kterým aplikace umí pracovat je vytvořen potom této třídy, který implementuje metody, v kterých se jednotlivé typy liší. Ke každému potomkovy této třídy patří protějšek v podobně potomka třídy Dialog, který zprostředkovává interakci s uživatelem pro daný typ metadat.

4.3.6 Třída Dialog

Jde o abstraktní třídu, která byla rozdělena na abstraktní část a odvozené třídy implementující rozdíly pomocí stejné logiky jako třída MetadataType. Abstraktní třída implementuje základní metody pro tvorbu dialogového okna, různé editační prvky tlačítka atd. Odvozená třída těchto metod využije pro vytvoření konkrétního okna ze specifickým rozložením prvků. Dále potomek této třídy může implementovat různá rozhraní, a tím zachytávat a reagovat na různé události vyvolané interakcí mezi uživatelem a některým z prvků v okně.

4.4 Vzhled a ovládání

Při návrhu ovládání pokud jde o rozšíření existující aplikace jako v mém případě je důležité dodržet základní logické a grafické uspořádání vzhledu jednotlivých dialogových oken. V mém případě mi

k tomu pomohl dokument[12] s pravidly pro rozložení prvků dialogového okna přímo od OpenOffice.org.

Dialogové okno může být definováno v jednotkách pixelů(px) nebo ve speciální jednotce Map AppFont(ma), kterou OpenOffice.org definuje takto: jeden Map AppFont(ma) bod je roven jedné osmině výšky a čtvrtině šířky průměrného znaku systémového (uvažuje se aktuálním operačním systémem používaného) písma. V dalších odstavcích této kapitoly jsou uvedeny všechny rozměry v této jednotce (ma), pokud není přímo uvedena jiná jednotka.

Rozměry jednotlivých prvků v dialogovém okně mají uvedeny předefinované hodnoty šířky nebo výšky, někdy oboje, tak, aby odpovídaly vzhledu zbytku aplikace. Mezery mezi těmito funkčními prvky mají také definovány určité velikosti. Brát na tato doporučení zřetel při návrhu vzhledu nemohu než doporučit. Každá skupina prvků okna podle jejich významu (tlačítka, editovací pole, prostý text, atd.) mají jiné pro jejich ideální funkcionalitu vhodné rozměry. Vzájemné mezery mezi těmito prvky, které kompenzují tyto rozdíly tak, aby výsledné dialogové okno působilo uhlazeným a vyváženým dojmem se dopočítávají celkem obtížně, ale právě při využití předefinovaných hodnot jsem zjistil, jak perfektně jsou tyto hodnoty nastaveny a celkově sladěny.

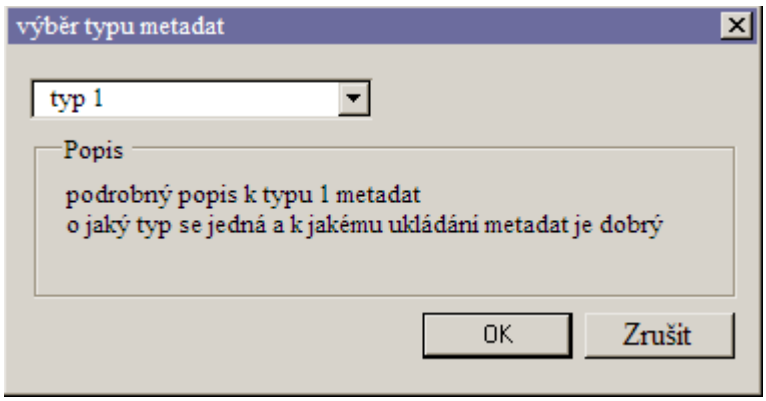
Šířka	Výška	Grafický prvek
50	14	Tlačítka
Dle potřeby	12	Editační, combo a list Boxy
Dle potřeby	8	Radio a Check Boxy, Text
6 nebo 4 (a kombinace obojího)	6 nebo 4 (a kombinace obojího)	Mezery mezi grafickým prvky

Tabulka 1: Předefinované hodnoty grafických prvků dialogového okna

Startovním bodem mé aplikace je grafický prvek tzv. Ovládací Panel neboli Toolbar (více *Ilustrace 9)*. Odtud uživatel přes funkční tlačítka ovládá funkce rozšíření, jako přidání nových metadat, editace (slouží také pro zobrazení uložených) metadat, odstranění metadat. Všechny tyto funkce jsou spjaty k textu označeného v dokumentu uživatelem pomocí kursoru výběru. Při editaci nebo odstranění metadata je dostačující kurzor umístit do textu ke kterému se vztahují nějaká metadata. Tento způsob doporučuji, protože je pro uživatele rychlejší a pohodlnější a nenastává problém z označením více vzájemně nezávislých metadat.

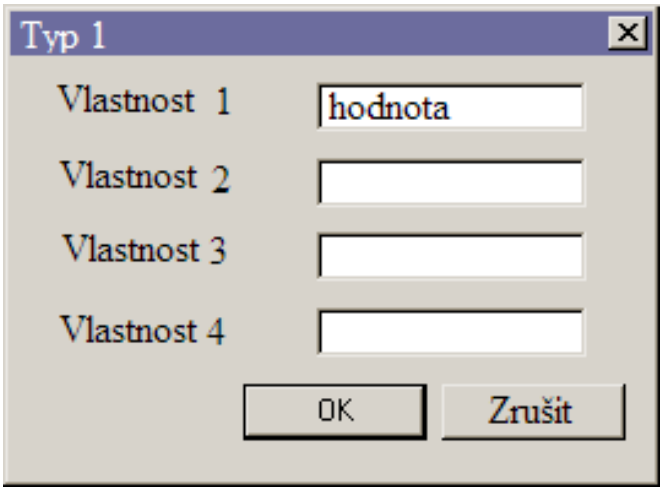
Ilustrace 9: Ovládací panel

Akce vyvolaná tlačítkem Přidat nová metadata zobrazí dialogové okno (více *Ilustrace 10)* pro výběr jednoho z aplikací podporovaných typů metadat. Při této akci mohou nastat chyby, týkající se pozice uživatelem nastaveného kurzoru. Na tyto chyby je uživatel upozorněn zprávou v informačním okně (více *Ilustrace 13)* a je požádán o napravení této chyby. Poté následuje editační okno zvoleného typu metadat a postup je odtud schodný jako při akci editace daného typu metadat.



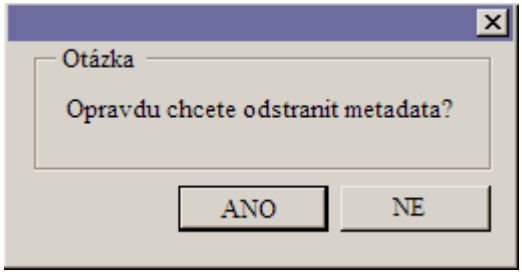
Ilustrace 10: Výběr typu Metadat

Akce vyvolaná tlačítkem Editovat metadata zobrazí dialogové okno (více *Ilustrace 11)* pro editování daného typu metadat, v kterém se zobrazí uložené metadata pro daný objekt, vztahující se k uživatelem zvolenému textu pomocí kurzoru výběru. Uživatel může editovat jednotlivé hodnoty a po ukončení editačního okna pomocí tlačítka OK dojde k uložení změn hodnot provedených uživatelem. Při této akci mohou nastat chyby při ukládání uživatelských hodnot například nevalidní URI. Taková hodnota se neuloží a uživatel bude na tuto skutečnost upozorněn informačním oknem (více *Ilustrace 13)*.



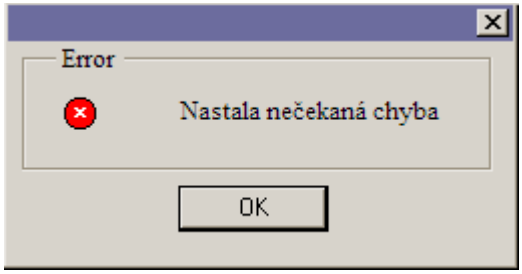
Ilustrace 11: Editační okno pro metadata typu 1

Akce vyvolaná tlačítkem Odstranit metadata zobrazí dialogové okno (více *Ilustrace 12)* pro potvrzení celé akce zabraňující nechtěnému stisku tlačítka. Po odsouhlasení akce se provede odstranění všech metadat k danému objektu určeného uživatelem pomocí kurzoru výběru. Nepovede-li se odstranění všech metadat, je o tom uživatel informován a je mu nabídnuto zobrazení editačního okna z metadaty, které se nepodařilo odstranit.



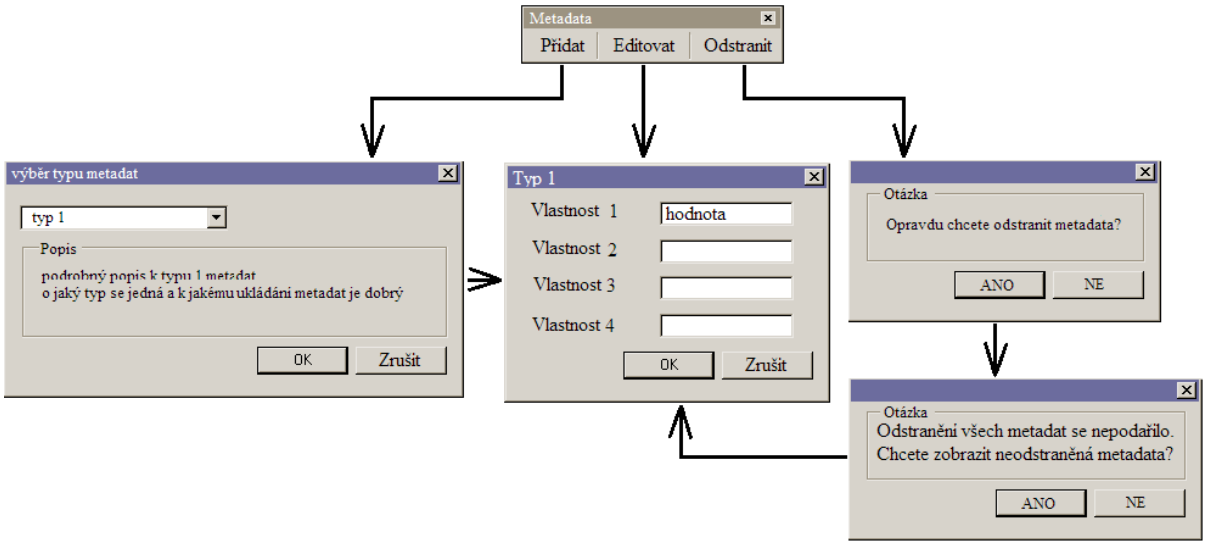
Ilustrace 12: Odstranění metadat

Pro informaci uživatele o různých stavech a chybách aplikace slouží informační a chybové okno doplněné o detail nastalé chyby (více *Ilustrace 13)*.



Ilustrace 13: Chybové okno

Nejdůležitějším oknem aplikace je editační okno, které je jedinečné pro každý typ metadat podporovaných aplikací, protože toto editační okno slouží zároveň i jako informační okno zobrazující uložená metadata, lze se k němu dostat (proklikat) více způsoby. Vše přehledně demonstruje následující obrázek (více *Ilustrace 14)*



Ilustrace 14: Kompozice oken aplikace

4.5 Realizace implementace aplikace

Aplikaci jsem realizoval v programovacím jazyku Java. Celou aplikaci jsem pro přehlednost rozdělil do tří balíků (package). Základním balíkem je *vut.openoffice.metadata* obsahující třídy implementující hlavní logiku aplikace. Dalším balíkem je *vut.openoffice.metadata.type* obsahující třídy implementující práci s metadaty v OpenOffice.org. Posledním balíkem je *vut.openoffice.metadata.gui* obsahující třídy implementující dialogová okna mojí aplikace. Pro popis konkrétních metod daných tříd je na přiloženém disku uložena generovaná programová dokumentace.

4.5.1 Nastalé a řešené problémy při implementaci

Největším problém, na který jsem při implementaci narazil byla špatná funkčnost metody *com.sun.star.rdf.XDocumentMetadataAccess.addMetadataFile*. Tato metoda má vytvořit prázdný soubor RDF/XML s definovanými klasifikačními schématy. Informace o těchto schématech jsou uložena v souboru manifest.rdf více v příkladu v *3.3.2Uživatelská RDF Metadata v OpenDocument verze 1.2*. Ve skutečnosti metoda vytvoří soubor RDF/XML, ale již nevloží metadata o tomto souboru do manifest.rdf. Naštěstí metadata z tohoto souboru jsou programově dostupná a tento problém je vyřešen dodatečným vložením správných metadat o RDF/XML souboru do metadat souboru manifest.rdf. Tato konkrétní úprava je implementována v metodě *vut.openoffice.metadata.type.IBPAType.createMetadataFile* a po opravě originální OpenOffice.org metody bude úprava odstraněna.

Další překážkou při implementaci byla neexistence metody, která by byla schopná nějakým způsobem odlišit metadata vázaná na text od všech ostatních metadat v dokumentu. Tuto překážku jsem vyřešil pomocí znalosti formátu OpenDocument a způsobu ukládání metadat, vztahujících se k textu.

Text textového dokumentu je uložen v souboru content.xml a obsahuje-li nějaká metadata vázaná na text, je takový blok textu identifikován pomocí jednoznačného id. Jedinečná URI definovaná pro takovýto blok se skládá s IRI, a relativní cesty v balíku dokumentu. V této relativní cestě je zahrnut název souboru tedy content.xml a id bloku. Příklad metadat vázaných na text je v *3.3.2Uživatelská RDF Metadata v OpenDocument verze 1.2*. Řešením v mém programu je načtení všech metadat dokumentu a následné testování URI jednotlivých objektů z dané trojice RDF metadat na přítomnost relativní cesty do content.xml k bloku s id. Tento způsob není zcela ideální, ale do doby vydání oficiální funkce OpenOffice.org řešící tento problém, je plně dostačující pro funkci aplikace.

4.6 Srovnání s podobnými produkty

V současnosti aplikací, která asi nejvíce využívá možností RDF metadata v OpenDocument v 1.2 je kancelářský balík KOffice. KOffice je programován pro multiplatformní grafické prostředí KDE. Vývojáři tohoto balíku aplikací se snaží systematicky začlenit využití metadat do svých aplikací.

Tak jako můj program, nabízejí uživateli využít metadata vázaná na text pro uložení informací o osobě a události. KOffice rozšiřují tuto nabídku o uložení informací o přesné geografické poloze. Systematičnost KOffice v používání metadat se projevuje vzájemnou provázaností těchto metadat mezi jednotlivými aplikacemi tohoto kancelářského balíku. Například možností přetažení kontaktu na osobu uloženou v textovém dokumentu do adresáře nebo možností nastavit si jaké údaje o dané osobě se mají zobrazovat přímo v textu. Nicméně v současnosti jsou všechny tyto funkce zatím ve stádiu vývoje nebo závěrečného testování

4.7 Budoucnost a další vývoj

V budoucnosti budou metadata využívána v čím dál větší míře. Vývoj informačních technologii se zaměřuje na způsoby práce s dokumenty, automatickými metodami bez zásahu člověka a metadata jsou formátem jenž již nyní umožňuje být pro tyto účely využit.

Má aplikace demonstruje některé možnosti, ale ne zdaleka všechny, jak využít metadata v OpenDocumentu. Na to jsem při návrhu a implementaci pamatoval. Mou aplikaci lze snadno rozšířit. Některé možnosti pro rozšíření se nabízejí již dnes.

1. Podpora pro jazykové mutace a překlad aplikace do více jazyků.
2. Snadná rozšiřitelnost aplikace o další typy metadat vázaných na text (stačí pouze vytvořit třídu pro dané klasifikační schéma a třídu pro dialogové okno)
3. Možnost doprogramování jiné funkcionality využívající metadata

5 Závěr

Vytvořil jsem aplikaci, rozšiřující funkcionalitu kancelářského balíku OpenOffice.org o možnosti, umožňující uživateli uložit metadata typické pro osobu nebo událost do textového dokumentu formátu OpenDocument v 1.2. Tato metadata může uživatel opětovně zobrazit, upravit nebo odstranit. Pro metadata jsou použita rozšířená klasifikační schémata [1] a [3], což usnadňuje aplikacím třetích stran vlastní použití těchto metadat. Postupně jsem tak splnil jednotlivé body zadání.

První bod: seznámení se s potřebnými formáty a technologiemi, které jsou popsány v kapitole *3Analýza*. Nejvíce práce a času mi trvalo nastudování pokročilé práce programování rozšíření pro kancelářský balík OpenOffice.org.

Druhý bod zadání: navržení aplikace dané funkcionality jsem splnil v kapitole *4Návrh a implementace aplikace*. Při návrhu jsem využil všech možností objektového návrhu, jako jsou abstraktní třídy a dědičnost.

Třetí bod: tedy implementace samotné aplikace je uložen na přiloženém disku. Rozšíření je plně funkční a může být kdykoliv použito. K tomuto bodu patří kapitola *4.5Realizace implementace aplikace*.

Ke čtvrtému bodu: omezení implementace vůči možnostem modelu metadat v OpenDocument verze 1.2 se částečně vztahuje kapitola *4.5.1Nastalé a řešené problémy při implementaci*, která popisuje současné nedostatky OpenOffice.org pro práci s metadaty vázanými na text. Největší omezení mé aplikace je ovšem v tom, že využívá pouze metadata vázaná na text. Formát OpenDocument v.1.2 je vybudován na modelech metadat, a proto je možné pomocí metadat ovlivnit jakoukoliv část balíku dokumentu. Například přidávat další soubory nebo ukládat metadata o jakékoliv části balíku dokumentu. Tyto možnosti metadat nebyli předmětem této práce, ale samotnou aplikaci je možné o tuto funkčnost rozšířit.

Literatura

[1] Dan Brickley, Libby Miller.: FOAF Vocabulary Specification 0.97, 2009-12-15,[online], [cit. 2010-05-08]. Dostupný z WWW: <http://xmlns.com/foaf/spec/#term_Person>

[2] Dan Brickley, Libby Miller.: FOAF Vocabulary Specification 0.97, . , 2009-12-15 ,[online], [cit. 2010-05-08]. Dostupný z WWW: <http://xmlns.com/foaf/spec/#term_Organization>

[3] Dan Connolly, Libby Miller.: RDF Calendar - an application of the Resource Description Framework to iCalendar Data, 2005,[online], [cit. 2010-05-08]. Dostupný z WWW: <http://www.w3.org/TR/2005/NOTE-rdfcal-20050929/>

[4] OpenOffice.org community.: Uno-Arc.jpg, [online], [cit. 2010-05-08]. Dostupný z WWW: <http://wiki.services.openoffice.org/wiki/Image:Uno-Arc.jpg>

[5] Kay Ramme.: Uno Project, [online], [cit. 2010-05-08]. Dostupný z WWW: <http://udk.openoffice.org/>

[6] OpenOffice.org community.: Framework.png, [online], [cit. 2010-05-08]. Dostupný z WWW: <http://wiki.services.openoffice.org/wiki/File:Framework.png>

[7] OpenOffice.org community.: TextDocumentWithMethods.png, [online], [cit. 2010-05-08]. Dostupný z WWW: <http://api.openoffice.org/docs/DevelopersGuide/FirstSteps/TextDocumentWithMethods.png>

[8] OpenOffice.org community.: Using the Desktop,[online], [cit. 2010-05-08]. Dostupný z WWW: <http://wiki.services.openoffice.org/wiki/Documentation/DevGuide/OfficeDev/Using_the_Desktop>

[9] OpenOffice.org community.: Using Services,[online],[cit. 2010-05-08]. Dostupný z WWW: <http://api.openoffice.org/docs/DevelopersGuide/FirstSteps/FirstSteps.xhtml#1_5_2_Using_Services>

[10] OpenOffice.org community.: OpenOffice.org Api First Steps,[online], [cit. 2010-05-08]. Dostupný z WWW: <http://api.openoffice.org/docs/DevelopersGuide/FirstSteps/FirstSteps.xhtml>

[11] OpenOffice.org community.: RDF metadata,[online],[cit. 2010-05-08]. Dostupný z WWW: <http://wiki.services.openoffice.org/wiki/Documentation/DevGuide/OfficeDev/RDF_metadata>

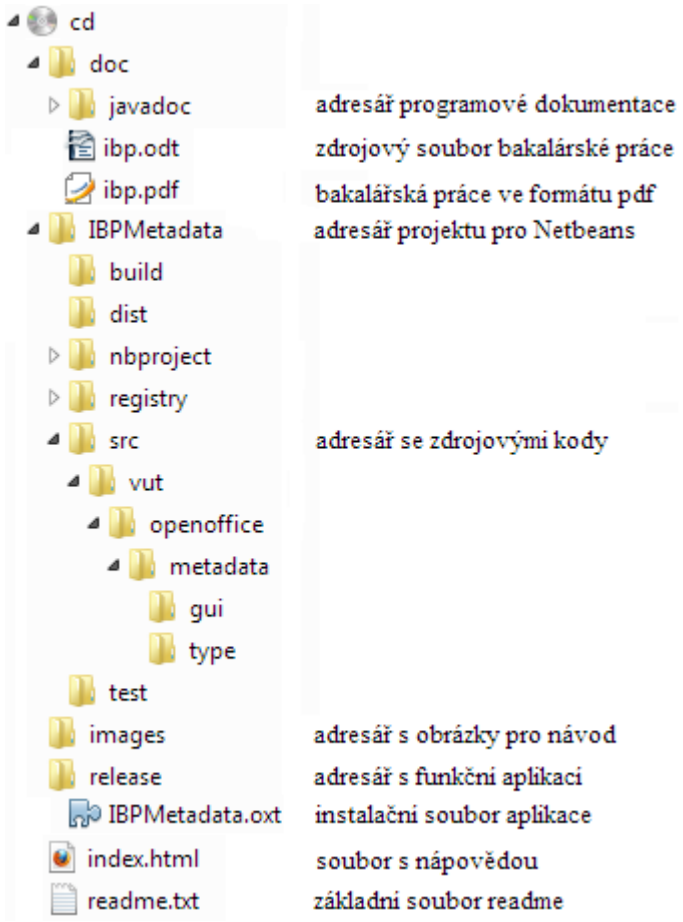
[12] Christian Jansen.: Dialog Specification and Guidelines - Visual Design, . Sun Microsystems, 2001,[online],[cit. 2010-05-08]. Dostupný z WWW: <http://ui.openoffice.org/knowledge/DialogSpecificationandGuidelines.odt>

Seznam Ilustrací

Seznam příloh

Příloha A

Obsah CD



V adresáři **doc** je uložena bakalářská práce v souboru **ibp.pdf** a adresář javadoc obsahující programovou dokumentaci ve formě html souborů.

Adresář **IBPMetadata** je zdrojovým adresářem projektu rozšíření pro vývojové prostředí Netbeans.

Adresář **images** obsahuje obrázky použíté v nápovědě, kterou je dokument index.html.

V adresáři **release** je soubor instalační soubor rozšíření **IBPMetadata.oxt**.