VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

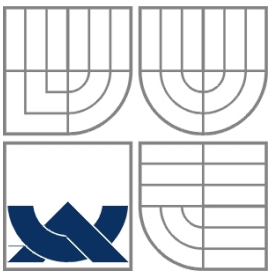
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

INTERAKTIVNÍ VIZUALIZACE XML

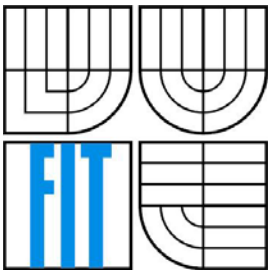
BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE DANIEL KUBÍČEK
AUTHOR

BRNO 2007



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

INTERAKTIVNÍ VIZUALIZACE XML

INTERACTIVE VISUALIZATION OF XML

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

DANIEL KUBÍČEK

VEDOUCÍ PRÁCE
SUPERVISOR

ING. PETR CHMELAŘ

BRNO 2007

Zadání bakalářské práce/5726/2006/xkubic17

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2006/2007

Zadání bakalářské práce

Řešitel: **Kubíček Daniel**
Obor: Informační technologie
Téma: **Interaktivní vizualizace XML**
Kategorie: Databáze

Pokyny:

1. Seznamte se s problematikou XML a jeho transformacemi.
2. Identifikujte a pokuste se vyřešit problémy interaktivního zobrazení XML dat včetně jejich definice.
3. Vytvořte grafický nástroj pro zobrazení dokumentů XML.
4. Zhodnoťte vlastnosti a případné vylepšení nástroje.

Literatura:

- Bradley, N. *XML - kompletní průvodce.* Grada Publishing, Praha 2000, 536 stran. ISBN 80-7169-949-7.
- Quin, L. W3C. *Extensible Markup Language (XML).* 2006. Dostupný z: http://www.w3.org/XML/.
- Oracle Corporation. *Oracle Technology Network.* 2006. Dostupný z: http://www.oracle.com/technology/.

Při obhajobě semestrální části projektu je požadováno:

- Body 1 a 2.

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

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

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

Vedoucí: **Chmelař Petr, Ing.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2006
Datum odevzdání: 15. května 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

L. S.

doc. Ing. Jaroslav Zendulka, CSc.
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. Pan

Jméno a příjmení: **Daniel Kubíček**
Id studenta: 88936
Bytem: Ečerova 956/5, 635 00 Brno
Narozen: 12. 04. 1984, Brno
(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta informačních technologií
se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....
(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
bakalářská práce

Název VŠKP: Interaktivní vizualizace XML
Vedoucí/školitel VŠKP: Chmelař Petr, Ing.
Ústav: Ústav informačních systémů
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:
tištěné formě počet exemplářů: 1
elektronické formě počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
   - ☐ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

.....	.....
Nabyvatel	Autor

Abstrakt

Práce se zabývá problémem zobrazování XML. V první části je ve stručnosti popsán jazyk XML a jeho základní charakterizace. Stručně shrnuty možnosti využití XML, jeho výhody, syntaxe XML a popsána logická struktura. Práce také částečně popisuje i problematiku načítání XML v Javě. Dále se zabývá problémem vizualizace XML, kde jsou shrnuty v krátkosti vlastnosti, výhody a nevýhody dostupných XML vizualizačních nástrojů.

Kapitola Implementace popisuje výběr vhodného programovacího jazyka a vývojového prostředí. Stručně popisuje jednotlivé dostupné modelovací nástroje se zdůvodněním, proč nemohly být použity.

Vytvořená aplikace obsahuje dva typy zobrazení XML elementů, základní a agregační typ. Byla tak naprogramována aplikace umožňující jednoduché zobrazení XML. Tento program je ve fázi praktického a využitelného vizualizačního nástroje a je možné jej dále rozšiřovat a upravovat.

Klíčová slova

XML, vizualizace, strukturovaná data, Java.

Abstract

This bachelor's thesis is engaged in problem of visualization XML. In the first part, there is described XML language in brief and his basic characterization. There is resumed the possibility of XML usage, his advantage, XML syntax and logical structure. Bachelor's thesis also describes the problematic of reading XML in Java. The thesis is engaged in problems of XML visualization. There are summarized XML properties, advantages and disadvantages of available visualization tools in short.

The Implementation chapter describes choosing of suitable programming language and development system. It briefly defines available modeling tools with individual reasons why they couldn't be used.

The developed application contains two types of displaying XML elements, the basic type and the aggregation type. The program output is supposed to be practical and usable visualization tool, although many extensions can be made.

Keywords

XML, visualization, structured data, Java.

Citace

KUBÍČEK, Daniel. *Interaktivní vizualizace XML*, bakalářská práce, Brno, FIT VUT v Brně, 2007.

Interaktivní vizualizace XML

Prohlášení

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

.....
Jméno Příjmení
Datum

Poděkování

Děkuji Ing. Petru Chmelařovi za odborné vedení během mé bakalářské práce. Zejména nastínění pokročilého využití XML.

Obsah

Úvod

V době rozmachu internetu a rozvoje komunikačních technologií vznikl formát pro publikování a výměnu dokumentů. Tento formát se jmenuje XML - eXtensible Markup Language nebo-li česky rozšířitelný značkovací jazyk.

Díky spojení výborných vlastností značkovacích jazyků a rozšiřitelnosti, jazyk XML rychle pronikl do mnoha oblastí sdílení informací. Ať už do samotné oblasti publikovaní informací, jako formát konfiguračních souborů aplikací nebo jako datový výstup databází.

Ale právě kvůli tak velikému rozsahu oblasti použití, vznikají velké a nepřehledné XML dokumenty. Tyto dokumenty je třeba vhodně zobrazovat nebo-li interpretovat z něj požadované informace. Na internetu se objevuje velké množství nástrojů pro zobrazování XML dokumentů. V podstatě všechny z nich ale zobrazují pouze stromovou strukturu, tedy pro zobrazování velkých dokumentů jsou nevhodné. Existují i nástroje, které nabízí alternativní zobrazování XML dokumentů ale většina těchto nástrojů nabízí zobrazování jen konkrétních XML dokumentů s předem známou strukturou.

Univerzální zobrazování je celkem náročná problematika, kterou se zabývá velká část vědecké obce. Přesto stále mnoho problémů není vyřešených. Největším nedostatkem zůstává samotné zobrazování. Jak vlastně data univerzálně zobrazit, aby byla jednoduše čitelná uživateli pro jakýkoliv vzorek dat?

Univerzálně zobrazit data přináší do této problematiky umělou inteligenci. Implementace umělé inteligence je ale velmi náročná a drahá. Proto se místo umělé inteligence stále využívá ještě interaktivita uživatele. Tedy v případě, že stroj se není schopen rozhodnout, nabídne všechny možnosti a je na uživateli, kterou zvolí.

Právě tímto interaktivním přístupem k zobrazení XML dat se zabývá tato bakalářská práce. Je v ní nástin jazyka XML a jeho přidružené technologie. Přináší pohled na možnosti zobrazení strukturovaných dat a zabývá se i vyhledáváním pro tento účel vhodných aplikací. V neposlední řadě přináší i řešení vlastního návrhu v podobě aplikace.

1 Shrnutí současného stavu

1.1 XML

1.1.1 Úvod do XML

XML, tedy rozšiřitelný značkovací jazyk (Extensible Markup Language), je velmi účinný univerzální formát pro tvorbu, správu a výměnu dokumentů. Zejména je ideálním formátem pro správu strukturovaného textu určeného pro šíření mezi různými druhy médií.

Značkování obecně znamená označení údajů, tak abychom jim přiřadili určitý význam. Strukturu dat v XML souborech je tedy možno přímo definovat a pomocí připravené šablony vyplnit daty.

Dokumenty XML se skládají z logické a fyzické struktury. Logická struktura dělí samotný dokument na elementy. Fyzická struktura umožňuje samostatně uložit a pojmenovat části dokumentu a následně je využít na několika místech (např. obrázky) [1].

XML podporuje vytváření elementů s vlastními jmény. Tedy vytváří popisnou podstatu místo aby definovali, jak se mají tyto elementy zobrazit (jak je tomu u jazyka HTML). Díky tomuto zobecněnému značkování se informace v XML dokumentech stávají samopopisujícími.

Vyčerpávající popis jazyka XML lze nalézt na stránkách *World Wide Web Consortium* (W3C) [7].

1.1.2 Typy XML dokumentů

XML dokumenty lze rozdělit dle jejich struktury do svou pomyslných skupin, které se mohou u mnohých dokumentů prolínat – a také se prolínají (např. email uložený pomocí XML). W3C tyto dvě skupiny nijak nedefinuje. Rozdělení skupin vyšlo z dlouholeté práce s XML.

1.1.2.1 Datově orientované XML dokumenty

Datově orientované XML dokumenty které jsou známé pod mezinárodním označením *data centric documents*, se využívají pro přenos dat mezi aplikacemi. Tyto dokumenty se dělí na malé logické jednotky, jejichž hodnoty jsou na úrovni hodnot typu PCDATA a atributů. Celková struktura dokumentů bývá pravidelná, ale nezáleží na pořadí elementů, pouze se vyžaduje striktní validita [6].

Příklad 1. Ukázka struktury datově orientovaného XML dokumentu.

```
<?xml version="1.0" encoding="windows-1250" ?>
  <objednavka id="35498834">
    <zakaznik id="654">
      <jmeno>Daniel</jmeno>
```

```
      <prijmeni>Kubíček</prijmeni>
      <ulice>Bezejmenná 523</ulice>
      <mesto>Brno</mesto>
    </zakaznik>
    <datum_dodani>15-05-2007</datum_dodani>
    <zbozi>
    <polozka kod="20171945" pocet="2">
      <cena mena="Kc">999</cena>
    </polozka>
    <polozka kod="68768449" pocet="5">
      <cena mena="Kc">58</cena>
    </polozka>
  </zbozi>
</objednavka>
```

Do těchto dokumentů jsou nejčastěji ukládány faktury, objednávky, ceníky nebo různá statistická či technická data. Jednoduše lze pomocí těchto XML dokumentů uchovávat tabulky relačních databází včetně informací o vzájemných relacích. Na příkladu 1 je uveden typický datově orientovaný dokument XML objednávky [6].

V této bakalářské práci jsem se zaměřil zvláště na vizualizaci právě datově orientovaných XML dokumentů, a to z důvodu usnadnění uživatelům čitelnost dat těchto dokumentů. Zejména dat generovaných z relačních databází plných obsahujících různé statistické údaje

1.1.2.2 Dokumentově orientované XML dokumenty

Struktura těchto dokumentů, odborně nazývaných *document-centric documen,* bývá nepravidelná a obsahuje delší části textu. Hlavním rozdílem oproti datově orientovaným dokumentům je nutnost zachovávat pořadí elementů [6].

Tyto dokumenty většinou vytváří lidé. Typickým příkladem mohou být knihy nebo časopisy uložené do XML např. pomocí programu DTD DocBook. Na příkladu 2 je zobrazen úryvek dokumentově orientovaného dokumentu vytvořeného pomocí DTD DocBook [5].

Příklad 2. Ukázka struktury dokumentově orientovaného XML dokumentu.

```
<?xml version='1.0' encoding='windows-1250'?>
<!DOCTYPE book PUBLIC '-//OASIS//DTD DocBook XML V4.2//EN'
          'http://www.oasis-open.org/docbook/xml/4.2/docbookx.dtd'>
<book lang="cs">
  <bookinfo>
    <title>První pokusná kniha</title>
    <author>
      <firstname>Jiří</firstname>
      <surname>Kosek</surname>
    </author>
  </bookinfo>
  <preface>
    <title>Úvod</title>
    <para>Odstavec textu.</para>
    <para>Druhý odstavec</para>
  </preface>
  <chapter>
    <title>První kapitola</title>
```

```
    <para>Text první kapitoly</para>
    <para>Další odstavec</para>
  </chapter>
  <chapter>
    <title>Druhá kapitola</title>
    <para>Text druhé kapitoly</para>
    <para>Další odstavec</para>
  </chapter>
  <appendix>
    <title>První příloha</title>
    <para>Text přílohy</para>
    <para>Další odstavec</para>
  </appendix>
</book>
```

1.1.3 Využití XML

1.1.3.1 XML jako prostředek pro výměnu dat (Aplikace XML)

V případě výměny dat mezi dvěma aplikacemi se XML nabízí jako vhodný formát. Řada aplikací pro sdílení či výměnu dat využívá právě technologii založenou na formátu XML.

Jako výměnný formát se uplatňuje především na poli relačních databázových systémů. Během přenosu mezi databázemi slouží XML jako vhodný obal dat. Díky strukturovanému a zároveň opakovatelnému přístupu se XML pasuje do ideálního řešení ukládání souvisejících datových polí definovaných tabulkami se vztahem 1:N.

Různé významné společnosti jako Microsoft, Netscape či W3C vyvíjí vlastní standarty pro přenos metadat. Většina z nich využívá XML, které rozšíří o významnou úroveň značkování.

1.1.3.2 XML jako prostředek pro publikovaní dokumentů

Jak již bylo řečeno XML je určeno k ukládání strukturovaných či částečně strukturovaných dokumentů, tedy dokumentů jako jsou návody, katalogy, časopisy zprávy a dalších.

Každý typický dokument lze zabalit do značek XML. Na obrázku 2 lze vidět příklad knihy „zabalené“ do formátu XML.

Velkou předností XML je vytváření a správa metadat (informací o informacích). Na obrázku 3 lze vidět příklad XML knihy a z něj vytažená metadata. Těmito metadaty jsou typicky obsah, rejstřík či datum vytvoření knihy. Díky XML jsou tyto metadata oddělena od samotných částí textu a v případě publikovaní nemusí být nikdy vidět. Naopak pro účely vyhledávání či klasifikace jsou stále k dispozici.

Díky strukturovanému formátu XML může dojít k vysoké úrovní automatizace na poli publikovaní XML. Jeden a tentýž dokument XML lze pomocí přidružených technologií lehce publikovat na papír, CD-ROM či Internet, vždy v požadovaném formátu. Jedná se o technologie jako XSL, XPath, XLink apod.

1.1.4 Výhody XML

1.1.4.1 XML jako standardní formát pro výměnu a sdílení informací

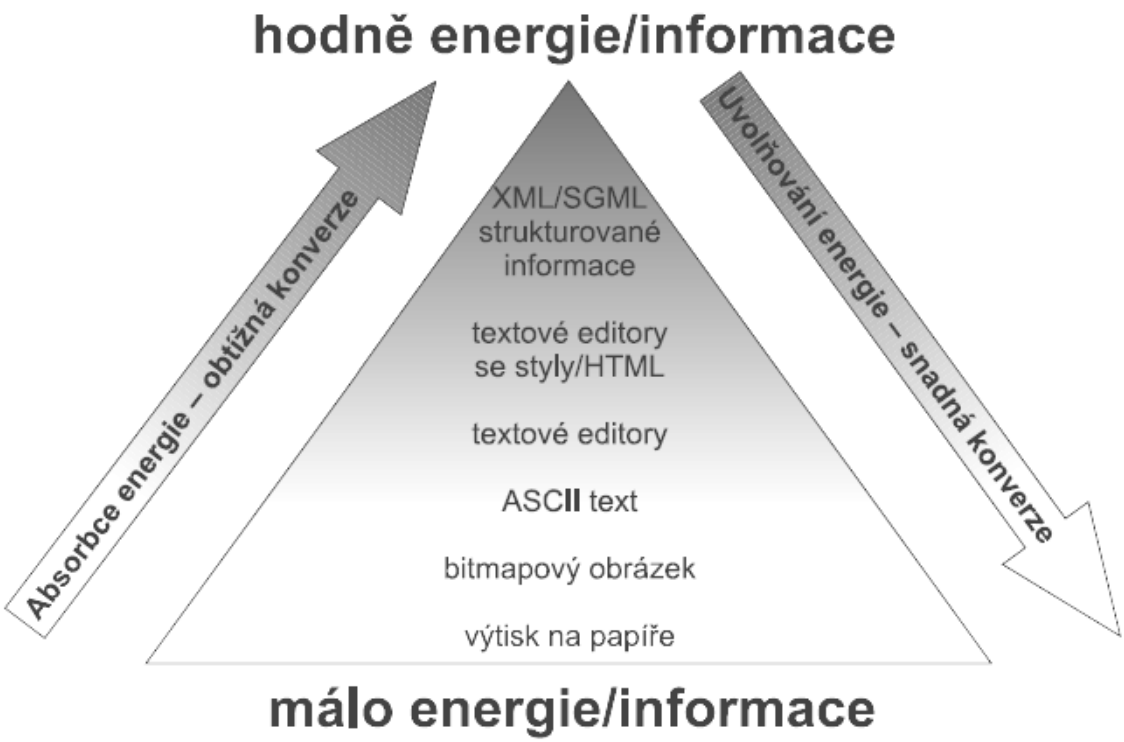
XML je standardní formát pro výměnu a sdílení informací, není majetkem žádné společnosti ani organizace. Není svázán s žádným softwarem či hardwarem. Díky celosvětové podpoře existuje mnoho programů, pomocí nichž jde XML dokumenty prohlížet, editovat a kontrolovat jejich syntaxi apod. Některé z těchto programů jsou zdarma jako samotné XML.

1.1.4.2 Mezinárodní podpora

Jako první na světě myslí tento formát od samého začátku na různé kódování jazyků. Jako znaková sada je využívána až 32bitová znaková sada ISO 10646, která dokáže pojmout všechny znaky dnes používaných jazyků. Díky tomu lze pomocí XML vytvářet a sdílet dokumenty v mnoha jazycích [3].

1.1.4.3 Vysoký informační obsah

Oproti prezenčnímu značkování XML dokumenty uchovávají mnohem více informací, které jsou zároveň přesněji definované. Toho lze zejména využít při vyhledávání.



Obrázek 1: XML dokumenty v sobě mají nejvíce informace, kterou mohou automaticky zpracovávat i počítače [3].

1.1.4.4 Snadná konverze do dalších formátů

XML samo o sobě nemá žádné prostředky pro konverzi do jiných formátů. Ale díky strukturovanosti dat v XML není problém pro jiné jazyky tuto konverzi provést. Nejznámějším stylovacím jazykem je asi jazyk kaskádových stylů (CSS), který slouží pro formátování a prezentaci dokumentu. Dalším jazykem může být XSL, který před samotným formátováním dokáže dokument úplně přetransformovat. Pro transformaci slouží XSLT standard.

1.1.4.5 Automatická kontrola struktury dokumentu

XML je možno využít také jako metajazyk pro vytváření vlastních značkovacích jazyků. Samotnou strukturu XML dokumentu, tedy jaké elementy a atributy může dokument obsahovat, lze definovat pomocí schémat. Tyto schémata zajišťují, že je XML dokument v požadovaném tvaru. V jednom dokumentu můžeme dokonce používat najednou nezávisle několik druhů sad značek díky tzv. jmenným prostorům.

1.1.4.6 Možnost vytváření odkazů a hypertextu

Hypertextové odkazy jsou pochopitelně nedílnou součástí XML. Oproti odkazům z HTML však nabízí mnohem více možností. Můžeme vytvářet klasické odkazy uvnitř dokumentu nebo mezi dokumenty také různé vícesměnné odkazy, které spojují několik dokumentů dohromady. A díky možnosti ukládání odkazů zcela mimo dokument lze jednoduše psát různé anotace či komentáře.

K tvorbě odkazů se dnes využívá technologie Xlink, kdy se jednotlivé odkazy určují podle URL. Dále pak XPointer k určování částí dokumentu a XPath, který umožňuje adresovat jednotlivé části XML dokumentu

1.1.5 Syntaxe XML

Jazyk XML je velmi jednoduchý. Jeho datový model se skládá z logické a fyzické struktury. Tato logická struktura podléhá mnohým omezením, které je třeba ve validních XML dokumentech dodržet. Každý XML dokument se skládá z elementů, jež jsou do sebe navzájem vnořené. Element může mít atributy. Každý XML dokument musí být obsažen v jednom elementu, který se nazývá kořenový element [2].

1.1.5.1 Element

Elementy se v dokumentech XML vyznačují pomocí značek tzv. *tagů*. Převážně je každý element vyznačen počátečním a koncovým tagem.

Příklad 3. Zápis elementu.

```
<element>hodnota elementu</element>
```

Název elementu je vždy uveden mezi znaky „<“ a „>“. Pro odlišení počátečního a ukončovacího tagu se před ukončovací tag vkládá znak „/“. Každému počátečnímu tagu musí odpovídat jeden koncový tag. Toto neplatí jen v případě, že element neobsahuje žádnou hodnotu. V takovém případě je možné zapsat tag dvěma způsoby, buď klasicky pomocí počátečního a koncového tagu nebo zkrácenou formou, u které se koncový znak „/“ vkládá za název elementu.

Příklad 4. Zkrácená forma tagu.

```
<element />
```

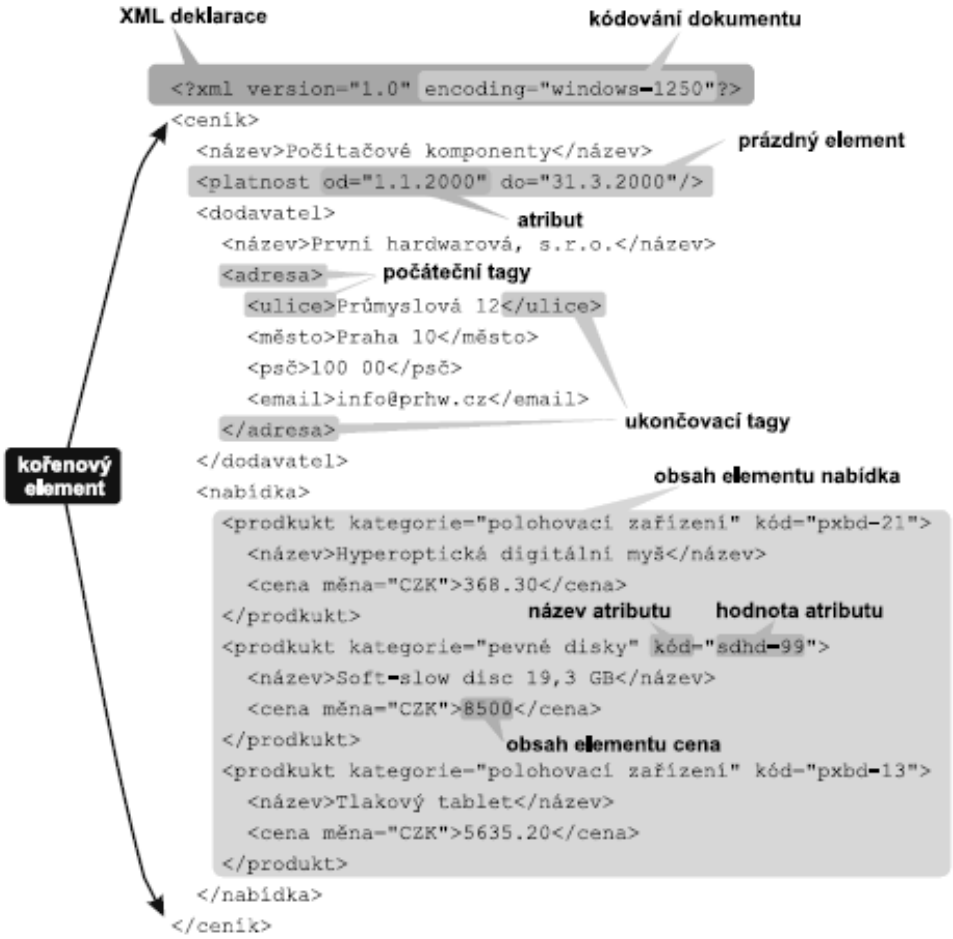
Elementy se ovšem nesmějí v XML dokumentu křížit

Příklad 5. Nesprávné křížení tagů.

```
<element>hodnota elementu
      <subelement>hodnota subelementu
      </element>
</subelement>
```

1.1.5.2 Atributy

Každý element může obsahovat i atributy. Názvy atributů se zapisují vždy za jméno elementu a jejich hodnota do uvozovek. Mezi jméno atributu a jeho hodnotou je vždy znak „=“. Pokud se v hodnotě atributu vyskytuje znak uvozovky je třeba jej zapsat pomocí únikové hodnoty " Příklad atributu elementu, lze vidět na obrázku 2.



Obrázek 2: Základní komponenty XML dokumentů [3].

1.1.6 Logická struktura

Významnou vlastností XML značkování je vytváření šablon dokumentů. Díky těmto šablonám lze kontrolovat výskyt a umístění elementů a atributů. Existují dvě skupiny těchto šablon. Tou starší je DTD (Document Type Definition) a novější XML Schema, které se snaží DTD nahradit.

1.1.6.1 DTD

Document Type Definition nebo také česky Definice typu dokumentu je zděděný jazyk od SGML. Jedná se o velmi silný nástroj, který obsahuje sadu pravidel pro definici struktury dokumentu.

DTD zavádí do XML formální pravidla struktury dokumentu. Definuje elementy, které se mohou či musí v dokumentu nacházet, a kde mohou být elementy použity v závislosti k ostatním elementům. Dále určuje jména a typ atributů elementu a zda je atribut povinný.

DTD není u XML dokumentů povinné, ale jeho přítomnost usnadňuje práci s XML dokumenty a zajišťuje, že formát XML odpovídá požadovanému formátu.

1.1.6.2 XML Schema

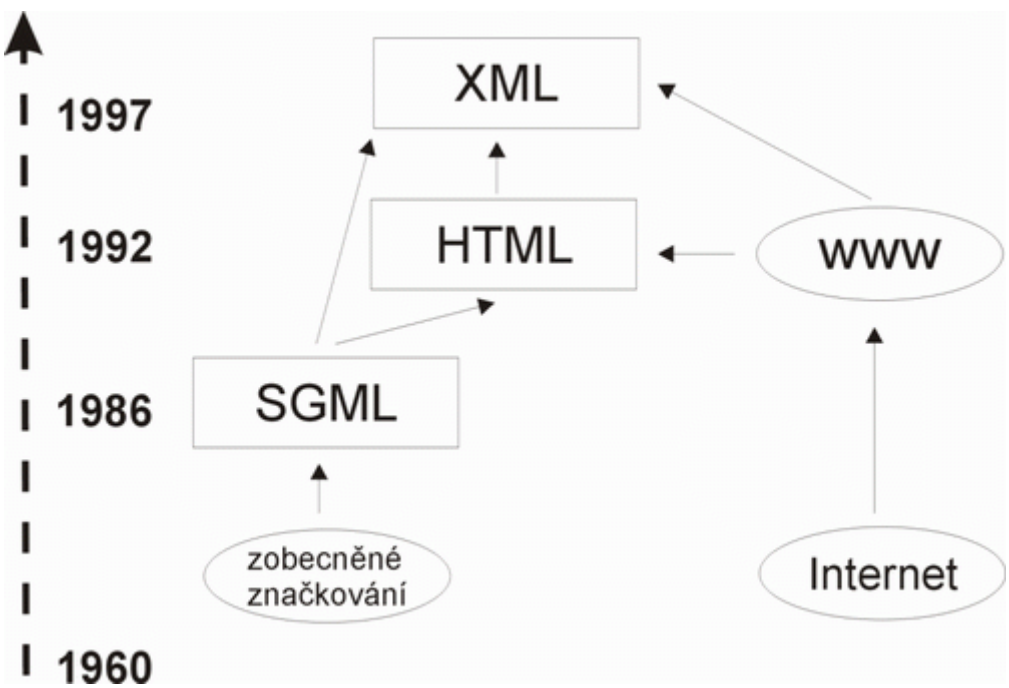
XML Schema vzniklo s nástupem využití XML na poli databází. A právě svět databází má větší požadavky na různé definice typů dat než jaké nabízí DTD. Kvůli validaci databázově orientovaných XML dokumentů vznikl jazyk pro definici schématu dokumentu.

Úplnou specifikaci XML Schema lze nalézt na stránkách konsorcia W3C XML Schema [15]

1.1.7 Historie značkovacích jazyků

Historie značkovacích jazyků sahá až k počátkům 80.let, kdy firma IBM řešila problém ukládání velkého množství právních dokumentů. Výsledkem snažení o vytvoření programově nezávislého formátu byl obecný značkovací jazyk GML (General Markup Language), za jehož vznikem stojí Charles Goldfarb, Edward Mosher a Raymond Lorie [3] . Díky tomu, že se tento formát značkovaní osvědčil, vznikl v roce 1986 na jeho základě oficiální značkovací jazyk SGML (Standard Generalized Markup Language).[4]

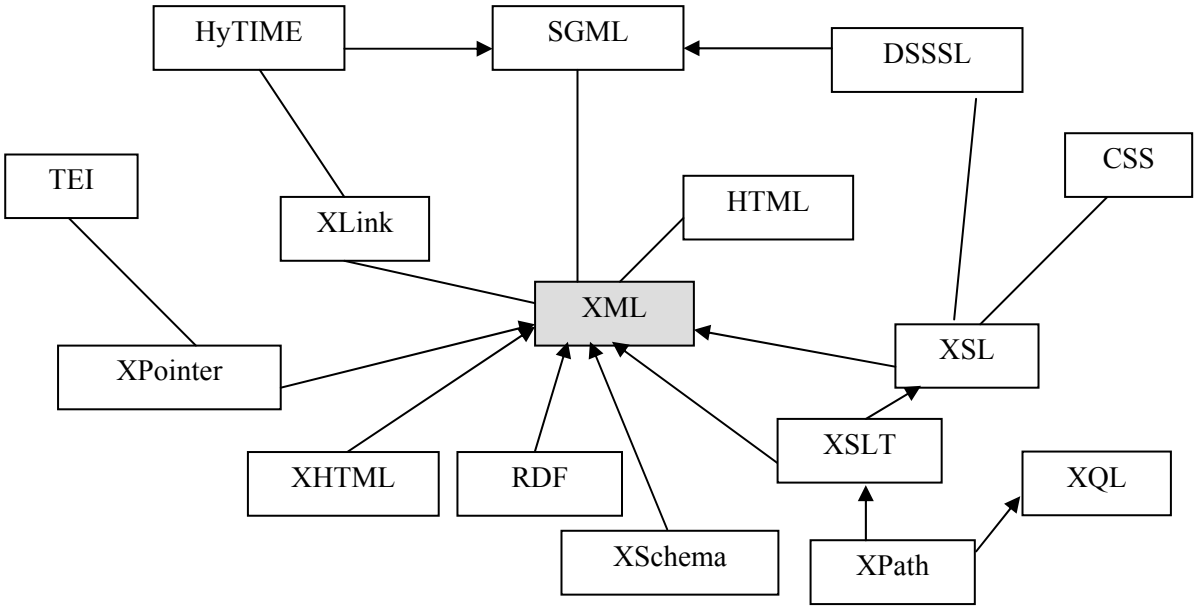
V roce 1991 s nástupem internetu a služby www vzniklo HTML pro účely zobrazování dokumentů na internetu. Bohužel HTML kvůli pevně definovaným značkám ztrácelo flexibilitu a tedy nevyhovovalo požadavkům vývojářů, proto vzniklo XML. Oficiálně bylo XML 1.0 představeno v roce 1998 po dvouletém intenzivním vývoji speciálně utvořeným týmem konsorcia W3C [6].



Obrázek 3: Schéma historie XML [1].

1.2 Rodina XML

Na poli sdílení informací a dat není formát XML zcela osamocen. Po jeho vzniku se zcela logicky objevily další technologie, které práci z XML usnadňují. Jedná se zejména o různé vyhledávací, klasifikační a prezentační nástroje.



Obrázek 4: Schéma rodiny XML.

1.2.1 Formátování dokumentů

Jak již bylo napsáno XML slouží k ukládání strukturovaných dokumentů. Tyto dokumenty je ale také třeba vhodným způsobem prezentovat. K tomu slouží různé formátovací nástroje.

Z pohledu vývojářů webu lze využít pro prezentaci XML kaskádové styly CSS. Avšak CSS úplně nepatří přímo do rodiny XML, jelikož syntaxe CSS je zcela odlišná od XML. Navíc CSS slouží spíše ke stylování HTML. Jako pokročilý nástroj pro prezentaci XML tedy není příliš vhodný

Vhodným standardem, který také spadá pod správu W3C je XSL (Extensible Stylesheet Language). XSL se zabývá formátováním a transformacemi XML dokumentů. Obsahuje tři komponenty. XPath, XSL-FO, a XSLT.

1.2.1.1 XPath

XPath je jazyk, který slouží k navigaci v XML dokumentech. Pomocí XPath lze vyjádřit relativní cestu z jednoho elementu ke druhému. Pomocí XPath lze vybírat jednotlivé elementy a atributy a pracovat s jejich hodnotami.

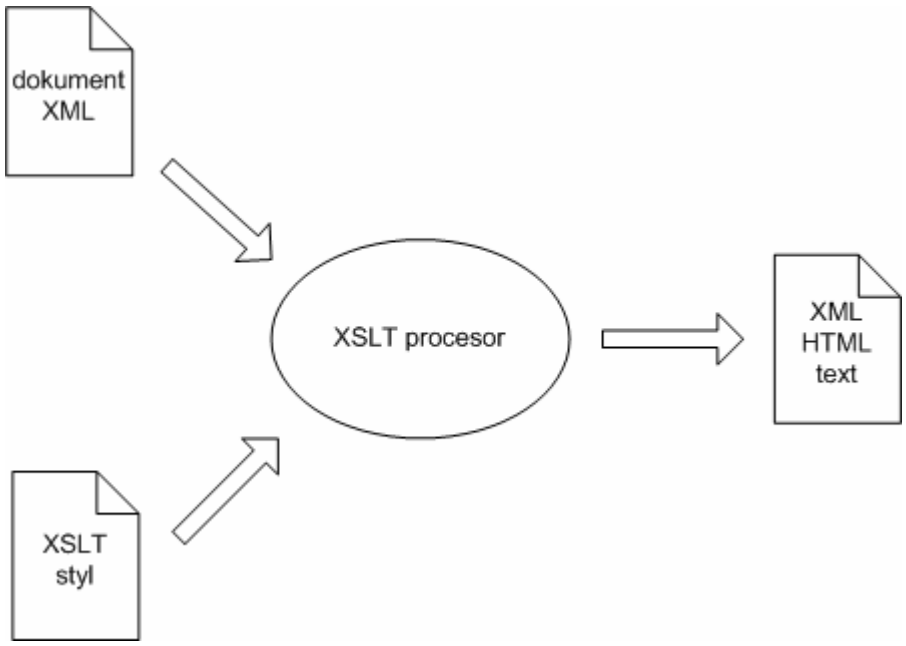
Podrobný popis XPath naleznete v jeho specifikaci XML Path Language (XPath) Version 1.0 [14].

1.2.1.2 XSLT

Standart XSLT slouží k transformaci XML dokumentů do jiného XML dokumentu, případně HTML nebo textového dokumentu.Využívá k tomu transformačních pravidel, které jsou uvedeny v dokumentu XSLT. Tento soubor stylů je identifikován jmenným prostorem http://www.w3.org/1999/XSL/Transform [13].

Samotnou transformaci provádí XSLT procesor. Ten dostane na vstup XML dokument a XSLT šablonu. Výsledkem je dokument určený k prezentaci. XSLT procesorů existuje celá řada a každý nabízí různé přednosti.

Podrobný popis XSLT naleznete v jeho specifikaci XSL Transformations (XSLT) Version 1.0 [13].



Obrázek 5: Schéma XSL transformace[1].

1.3 XML a Java

Velké množství volně dostupných nástrojů pro práci s XML je napsána v Javě. Je tedy pochopitelné, že Java je ideální programovací jazyk pro práci s XML.

V této podkapitole jsem se zaměřil na problematiku načítání XML dokumentů v Javě. Existují dva přístupy (SAX a DOM).

1.3.1 SAX

SAX (Simple API for XML) je standardní API pro zpracování dat XML.

Dokument XML se načítá proudově, přičemž se dokument rozdělí na jednotlivé části (počáteční a koncové elementy, jejich hodnoty, atributy apod.). Poté se volají jednotlivé události ohlašující nalezení konkrétní části.

1.3.2 DOM

Document Object Model – objektový model dokumentů byl vytvořen konsorciem W3C. Jeho hlavním účelem bylo určit, jakým způsobem mají webové prohlížeče přistupovat k XML a HTML dokumentům. Standard platný pro XML i HTML definuje Document Object Model (DOM) Level 1 Specification [14].

DOM načítá najednou celé XML dokumenty a v paměti je ukládá do stromové struktury. Tato technologie se nazývá *grove (Graph Representation Of property ValuEs)* [17].

Oproti SAX je DOM náročnější na paměť a je o něco pomalejší. Na druhou stranu nabízí efektivnější přístup k jednotlivým elementům XML dokumentu.

Na příkladu 6 je ukázka načítání XML dokumentů pomocí DOM v této bakalářské práci.

Příklad 6. Načtení xml dokumentu pomocí DOM.

```
import javax.xml.parsers.SAXParser;
import javax.xml.parsers.SAXParserFactory;
import javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMResult;
import javax.xml.transform.sax.SAXSource;
import org.w3c.dom.Document;
import org.xml.sax.InputSource;
import org.xml.sax.XMLReader;
public class DOMUtil{
    public static Document createDocument(InputSource is) throws
Exception{
        SAXParserFactory saxFactory = SAXParserFactory.newInstance();
        SAXParser parser = saxFactory.newSAXParser();
        XMLReader reader = new XMLTrimFilter(parser.getXMLReader());
        TransformerFactory factory = TransformerFactory.newInstance();
        Transformer transformer = factory.newTransformer();
        transformer.setOutputProperty(OutputKeys.INDENT, "no");
        DOMResult result = new DOMResult();
        transformer.transform(new SAXSource(reader, is), result);
        return (Document)result.getNode();
}}
```

1.3.3 JAXP

JAXP (Java API for XML Processing) je rozhraní, které umožňuje práci s XML dokumenty v rámci programovacího jazyka Java . Umožňuje jak přístup pomocí SAX tak pomocí DOM.

Součástí rozhraní JAXP je parser Crimson a procesor Xalan, které slouží k zpracování XML dokumentu. Oba programy jsou napsány v Javě. Architektura JAXPu umožňuje použít i jiný parser a procesor (samozřejmě pokud podporuje příslušné standardy tzn. SAX a DOM).

2 Otevřené problémy

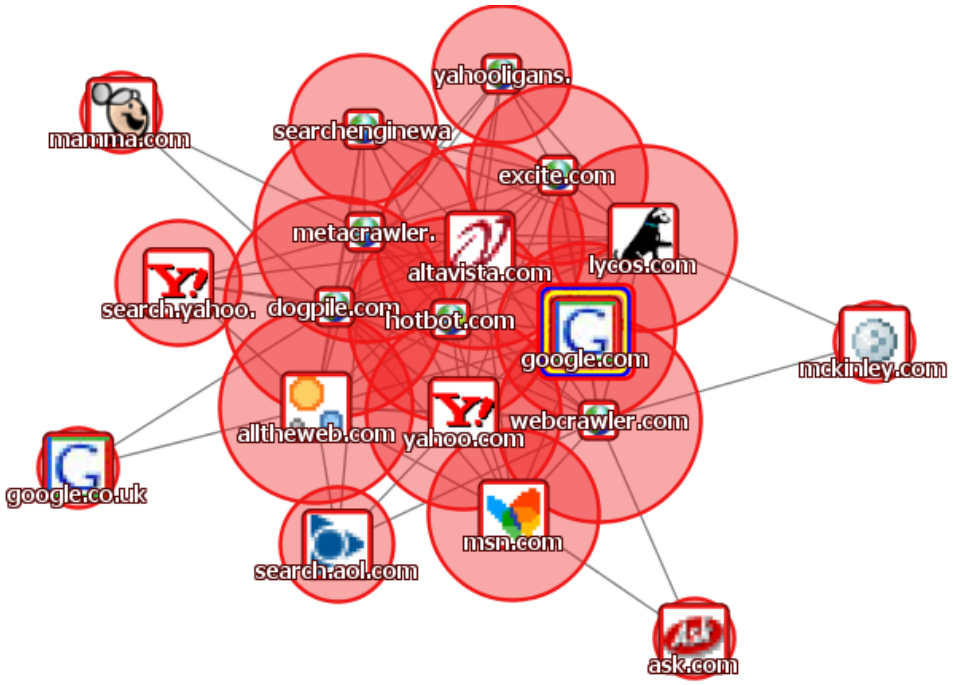
2.1 Vizualizace strukturovaných dat

Snaha o porozumění struktury dat i samotnému obsahu strukturovaných dat vede člověka k jejich vizualizaci. Vizualizaci interpretuje ve 2D nebo 3D scénách, protože v těch se pohybuje a těm nejlépe rozumí. 2D modely dat jsou jednodušší než modely 3D. Navíc 2D modely v dnešní době stále přinášejí lepší výsledky, hlavně kvůli velké složitosti 3D modelů a ještě ne úplné vyspělosti nástrojů pro 3D modelování.

Existuje několik volně dostupných vizualizačních nástrojů, které nabízí různorodý pohled na data. U většiny z těchto nástrojů pokračuje vývoj a využívají se i u komerčních projektů. Bohužel žádný nástroj neumožňuje plně interaktivní zobrazení XML dokumentů včetně jeho definic.

2.1.1 TouchGraph

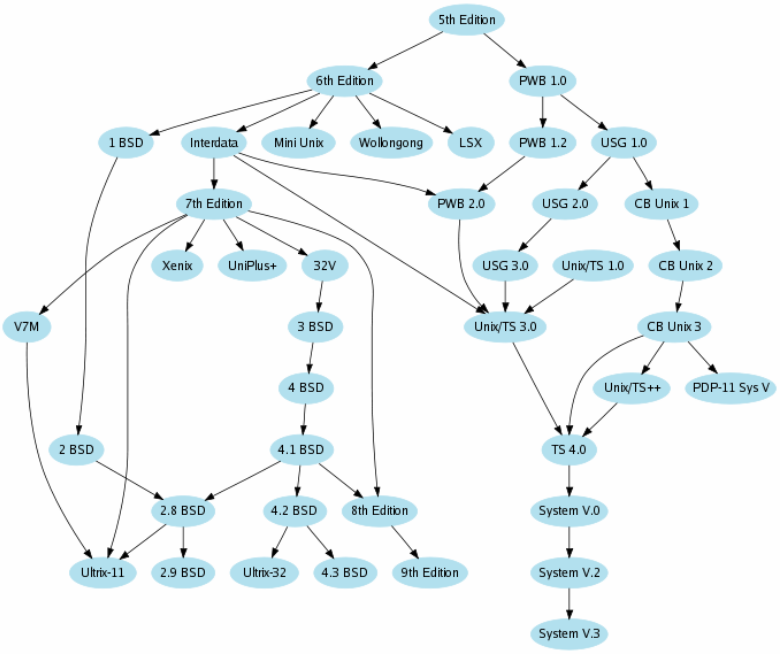
TouchGraph je velmi zdařilý open source nástroj pro vizualizaci. Využívá techniky spring-layout a focus-context pro zobrazení grafů. Nejznámějším projektem tohoto nástroje je Google Browser. Ten v podstatě zobrazuje výsledek vyhledávání odkazů podobných stránek na googlu do grafové podoby. Na obrázku je vidět struktura vyhledávání podobných webových stránek www.google.com [9].



Obrázek 6: Ukázka grafického výstupu nástroje Gogole Browser [10].

2.1.2 GraphViz

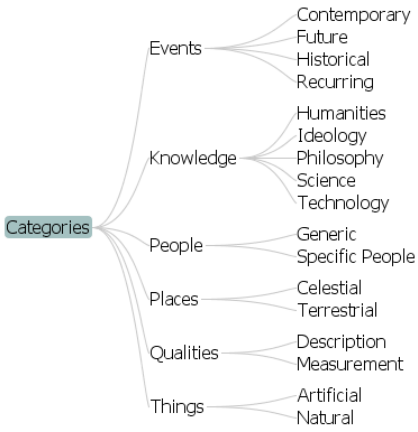
Dalším ze zdařilých open source projektů je GraphViz, který také prezentuje strukturovaná data pomocí diagramů a sítí. Jeho uplatnění je zejména v oblasti kreslení softwarového inženýrství a designu databází a počítačových sítí. Pro ukládání dat využívá XML nebo GXL a výstupem aplikace je obrázek, SVG grafika, PDF nebo PS dokument. Na obrázku je uveden příklad grafu z GraphViz [10].



Obrázek 7: Ukázka grafického výstupu aplikace GraphViz.

2.1.3 Prefuse

Prefuse je sada nástrojů pro 2D vizualizaci, založená na jazyce Java. Zaměřuje se na zobrazování strukturovaných dat do stromů grafů a tabulek. Využívá knihovnu Java 2D a je lehce integrovatelný do aplikací založených na Java Swing či Java Applet [11].



Obrázek 8: Ukázka zobrazení stromu pomocí Prefuse.

2.1.4 HyperGraph

Open source projekt HyperGraph slouží především k zobrazování grafů a stromů založených na hyperbolické geometrii. Výhodou tohoto projektu je, že dokáže zobrazit velké množství uzlů ve stromu a zároveň má uživatel možnost sledovat v danou chvíli jen určitou část stromu. HyperGraph je Java Applet, který načítá soubory XML z nichž konstruuje stromovou strukturu. Na obrázku je vidět jednoduché XML zobrazené pomocí HyperGraphu. Nevýhodou tohoto projektu je, stejně jako většiny zobrazovacích programů, že není určen k zobrazení obecného XML, ale podléhá přesnému formátu [12].

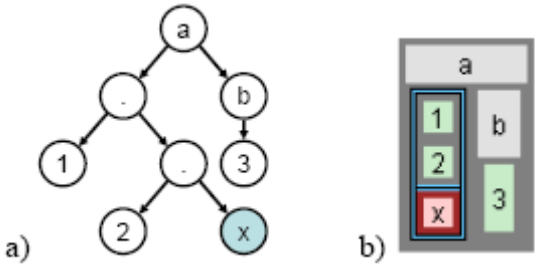
Příklad 7. XML vstup pro Hypergraph.

```
<?xml version="1.0" encoding="windows-1250"?>
<!DOCTYPE GraphXML SYSTEM "GraphXML.dtd">
<GraphXML>
<graph id="example">
<node name="1"><label>rodič</label></node>
<node name="2"><label>syn</label></node>
<node name="3"><label>dcera</label></node>
<edge source="1" target="2">
<style><line linewidth="6" colour="#0000FF"/></style>
</edge>
<edge source="1" target="3">
<style><line linewidth="6" colour="#FF0000"/></style>
</edge>
</graph>
</GraphXML>
```

Obrázek 9: Grafické znázornění příkladu 7 pomocí Hypergraphu.

2.1.5 Tree Rewriting

Zajímavý pohled na zobrazování přinášejí pánové J. Jelínek a P. Slavík v článku XML Visualization Using Tree Rewriting. Grafickou reprezentaci stromu podle jejich návrhu lze vidět na obrázku 10 [16].



Obrázek 10: Grafická reprezentace stromu.

2.2 Interaktivní vizualizace XML

Pojmem interaktivní vizualizace XML je velmi obecný. Co si pod ním vlastně představit? Stačí se podívat na předešlou pod kapitolu, kde jsou různé možnosti řešení vizualizace XML. Mnohé z těchto variant řešení jsou interaktivní – tedy zobrazována na základě podnětů uživatele.

Cílem však není pouze udělat interaktivní prohlížeč, nýbrž vytvořit program, který dokáže přehledně zobrazit XML dokument včetně jeho definic a využít data jako metadata pro další elementy XML dokumentu. A právě k tomuto je třeba využít interaktivity uživatele, který definuje jak zobrazovat jednotlivé elementy XML. Mé řešení nabízí dva typy pohledů na XML elementy. Jedná základní typ a agregační typ.

Základní typ představuje jeden element XML dokumentu. Tento element obsahuje atributy, subelementy případně vlastní hodnotu. Agregační typ shlukuje elementy XML dokumentu stejného jména a entity k sobě. V tomto elementu představují další subelementy XML metadata daného XML elementu. Podrobně se těmito typy zabývám v kapitole 3.3.1.

3 Implementace

Kapitola implementace se zabývá postupem tvorby aplikace. Je v ní zahrnut vývoj, implementační řešení problému interaktivního zobrazení XML dokumentů, použité nástroje k vytvoření aplikace a samotný popis vytvořené aplikace. Součástí popisu jsou i náhledy na aplikaci.

3.1 Vývoj

Tato kapitola se zabývá vývojem samotné aplikace pro vizualizaci XML dokumentů. Jsou zde popsány problémy, se kterými jsem se potýkal a jejich řešení.

3.1.1 Výběr programovacího jazyku a příslušné vývojové prostředí.

Pro samotnou implementaci programu bakalářské práce jsem vybral programovací jazyk Java 1.5.0 JDK. Jedná se o velmi rozšířený objektově orientovaný programovací jazyk. Jeho hlavní výhodou je přenositelnost mezi platformami a různými zařízeními. Jedna aplikace tedy může fungovat na mobilních telefonech či jiných přenositelných zařízeních a zároveň na desktopových počítačích. Tato přenositelnost je díky tomu, že Java je jazyk interpretovaný. To znamená, že pro spuštění programu je třeba mít interpret Javy tzv. virtuální stroj Javy – Java Virtual Machina (JVM). Syntaxe jazyka Java je podobná syntaxi C a C++. Jedná se o zjednodušenou verzi [19].

Velkou výhodu pro tuto bakalářkou práci přináší Java v oblasti práce s dokumenty XML. Integrované rozhraní JAXP, popsané v kapitole 1.3 Java a XML, nabízí komfortní práci programátora s dokumenty XML. API dokumentaci platformy Java se nachází na Java 2 Platform Standard Edition 5.0 API Specification [8].

Pro jazyk Java existuje několik vývojových prostředí, které usnadňují práci programátora. Mezi nejznámější patří Netbeans IDE, Oracle JDeveloper, Borland JBuilder či Eclipse. Každé nabízí řadu nástrojů usnadňující vývoj aplikace. Já jsem si nakonec vybral Oracle JDeveloper zejména kvůli předpokládanému využití některých knihoven Oracle pro samotnou vizualizaci XML.

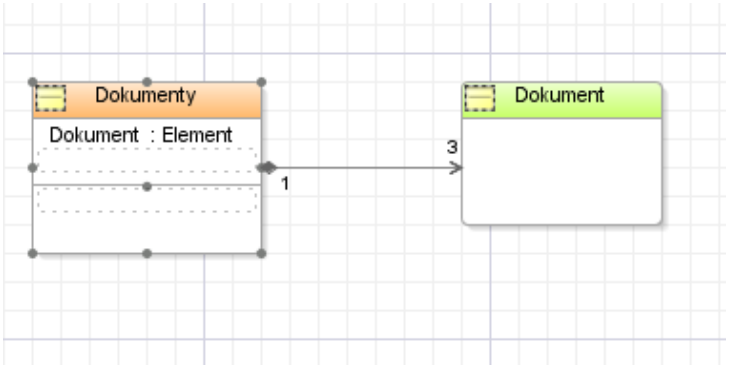
3.1.2 Problémy s prostředím pro vizualizace

Už od samého počátku bakalářské práce jsem měl v úmyslu použít pro samotnou vizualizaci některé z již hotových a vyzkoušených prostředí. Na internetu je k dispozici takových prostředí několik. Po podrobnějším prostudování se ukázalo, že žádné z nich nesplnilo mé očekávání.

3.1.2.1 Oracle Class diagram

Prvním a nejvážnějším adeptem pro vizualizace XML se zdál být Class diagram od společnosti Oracle. V prostředí JDeveloperu se jevilo jako ideální řešení. Nabízí všechny potřebné funkce modelování. Obsahuje různé druhy přechodů, včetně agregačních. Tento nástroj mne zaujal díky svému grafickému výstupu.

Ač na první pohled tento produkt vypadá jako volně dostupný, ukázalo se, že tomu tak není. Zdrojové kódy a programovou dokumentaci jsem se snažil získat u zákaznické podpory firmy Oracle. Tam mě odkázali na hlavní stránky vývojového prostředí JDevepoler, kde se zdrojové kódy ani dokumentace nenachází.



Obrázek 11: Ukázka Oracle Class diagram.

3.1.2.2 Visual Paradigm

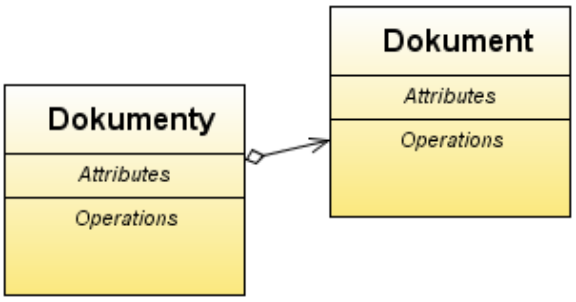
Jako další pěkně vypadající modelovací nástroj byl Visual Paradigm. Nabízel velké množství typů modelování. Od modelování diagramu tříd až po modelování databáze. Řekl bych že na poli modelování patří tento produkt k absolutní špičce.

Bohužel jedná se o komerční produkt. V Community Edition je sice možné Visual Paradigm využívat pro modelování nekomerčních projektů. Ale není možné jej využít při vývoji vlastních nástrojů, což je případ této bakalářské práce.

3.1.2.3 Netbeans UML Modeling beta

Netbeans UML Modeling beta je velmi podobný modelovací nástroj jako Oracle Class Diagram. Podle mého názoru není graficky tak dobře zvládnutý. Na první pohled využívá jednodušší grafický výstup. Stejně jako Oracle Class Diagram umí modelovat celou řadu diagramů.

Jeho obrovskou výhodou je to, že je open source a tedy ho lze využít pro vlastní projekty. Velkou nevýhodou se ukázalo být absence API dokumentace, kterou se mi nepodařilo sehnat ani od vývojářů Netbeans UML Modeling. Kvůli tomuto problému jsem nemohl použít ani tento nástroj ve své bakalářské práci.

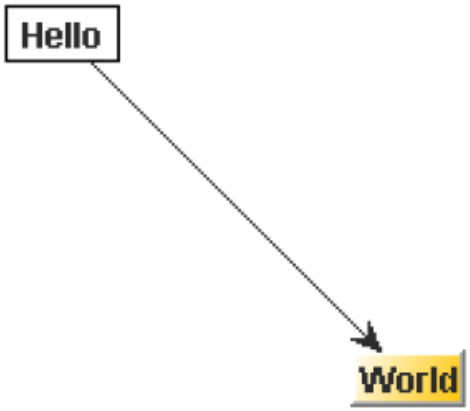


Obrázek 12: Ukázka Netbeans UML Modeling.

3.1.2.4 JGraph

JGpraph je open source nástroj pro tvorbu grafů a diagramů v Javě. Je plně kompatibilní s prostředím Java Swing, a proto využití ve vlastních produktech není příliš složité. Výhodou je podpora antialiasingu tzn. vyhlazování hran, zoomování grafu, podpora vrstev a seskupování.

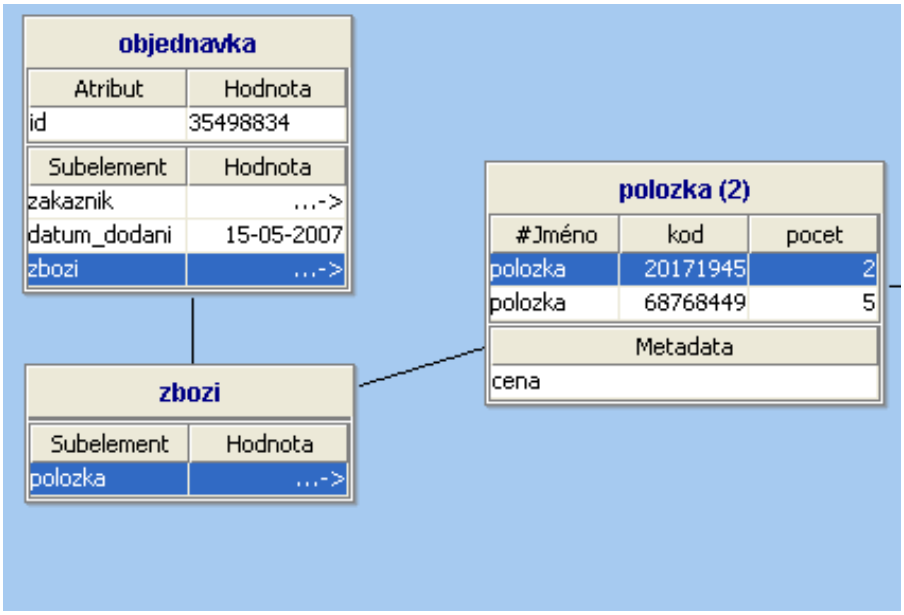
Přestože jsem studoval dokumentaci tohoto nástroje poměrně dlouho a byl téměř přesvědčený, že tento nástroj použiji, nestalo se tak. Narazil jsem na problém, kdy komponenty Java Swing umístěné do elementu diagramu nebyly aktivní a tedy nešlo jednoduše využívat jejich interaktivní vlastnosti. Po diskusi s vývojářem tohoto nástroje, jsem dospěl k závěru, že JGpraph není vhodné řešení pro tuto bakalářskou práci.



Obrázek 13: Ukázka JGraph.

3.1.2.5 Vlastní řešení vizualizace

Nakonec se mi nepodařil nalézt žádný vizualizační nástroj, který by splnil mé požadavky. Buď u nástroje chyběla příslušná dokumentace nebo se jednalo o komerční produkt nebo jsem nepřišel na způsob jak efektivně využít nástroj v tomto projektu. Proto jsem se rozhodl jej naprogramovat sám. Bylo jasné, že se mi nepodaří vytvořit prostředí, které by mohlo konkurovat prostředím, jejichž vývoj trval několik let. Přesto jsem nakonec vytvořil slušný základ pro další rozvoj. Podrobnějším popisem se zabývají další podkapitoly Implementace.



Obrázek 14: Ukázka vlastního řešení.

3.2 Implementační nástroje

V této kapitole jsou popsány všechny nástroje, které jsem použil pro vytvoření aplikace. Jedná se o programovací jazyk, vývojové prostředí, knihovny a nástroje pro vytvoření grafického uživatelského prostředí.

3.2.1 Programovací jazyk a vývojové prostředí

Implementace programu byla vytvořena ve vývojovém prostředí Oracle JDeveloper. Je naprogramována pomocí programovacího jazyka Java 1.5.0 JDK. Podrobněji se programovacím jazykem Java zabývá kapitola 3.1.1.

3.2.2 Implementace grafického uživatelského prostředí

Grafické uživatelské prostředí (GUI) je naprogramováno pomocí balíku Java Swing. Je to knihovna určená k tvorbě vizuálních komponent (např. okna, tlačítka, dialogy apod.). Vytvářené prostředí pomocí této knihovny je zcela nezávislé na platformě stejně jako Java samotná.

Pro vytvoření vzhledu GUI ve stylu operačního systému Windows XP je využita knihovna look and feel looks-1.3.2.jar. Samotná technologie look and feel umožňuje nezávislé grafické uživatelské prostředí zobrazit ve stylu požadovaného operačního systému.

Program je zasazen do java komponenty JFrame, která představuje hlavní okno programu. V tomto okně je vytvořeno menu pomocí JMenu. Lišta tlačítek (toolbar) je vytvořen pomocí komponenty JToolbar. Samotná zobrazovací plocha je rozdělena na dvě části. Levá část je vytvořena pomocí JTree a zobrazuje se v ní stromová struktura otevřeného XML dokumentu. Pravá část je

vytvořena pomocí XmladDiagramPane, což je třída odvozená od třídy JDesktopPane. Tato třída slouží k obsluze JInternalFrame, pomocí nichž jsou vizualizovány XML elementy. Všechny tyto komponenty jsou součástí balíku Java Swing a nabízejí všechny základní funkce pro grafickou vizualizaci aplikace.

3.2.3 Knihovny

Pro načítání a práci s XML dokumenty jsem využil JAXP (Java API for XML Processing). Tento nástroj umožňuje parsovat XML dokumenty jak pomocí SAX 2.0 tak pomocí DOM level 2. Těmito technologiemi se zabývám v kapitole 1.3.

Pro seřazování dat v tabulce byla použita volně dostupná třída TableSorter [18]. Tato třída při správné implementaci umožňuje uživateli seřadit dle abecedy data v tabulce.

Pro look and feel aplikace jak bylo napsáno v předchozí podkapitole, je využita knihovna looks-1.3.2.jar.

3.3 Vlastní implementace

Základní vlastností programu je interaktivně zobrazovat XML. Po otevření aplikace program umožňuje otevřít XML dokument a dále ho dle uvážení uživatele interaktivně zobrazit. Do komponenty JTree se načte stromová struktura XML dokumentu. Na komponentu XmladDiagramPane se XML dokument vykreslí interaktivně. Projektová dokumentace se nachází v příloze 2.

3.3.1 Implementace interaktivního zobrazení XML dokumentu

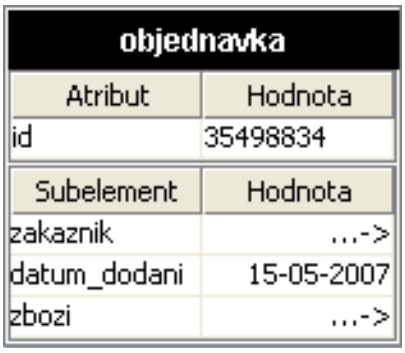
Celý XML dokument je načten do paměti jako struktura DOM strom. Na zobrazovací plochu je vykreslen kořenový element XML dokumentu. Tento element lze dále otevírat a zobrazovat jeho subelementy.

Aplikace obsahuje dva typy zobrazení XML elementů.

3.3.1.1 Základní typ

Základní typ představuje jeden element XML dokumentu. Je tvořen dvěma tabulkami. První tabulka, tabulka atributů, obsahuje atributy daného XML elementu. Druhá tabulka, tabulka elementů, obsahuje XML subelementy příslušného XML elementu, případně jejich hodnotu. Hodnota XML subelementu se v tabulce zobrazí jen pokud se jedná o konečný XML element (nemá již žádné další XML subelementy) a zároveň se nachází v dané entitě XML dokumentu jen jednou. V opačném případě se na místo hodnoty zobrazí znaky „...->“ což znázorňuje, že pro podrobnější zobrazení je nutné XML

subelement otevřít. Na obrázku 15 je možno vidět základní typ zobrazení kořenového elementu XML dokumentu z příkladu 1.



objednavka	
Atribut	Hodnota
id	35498834
Subelement	Hodnota
zakaznik	...->
datum_dodani	15-05-2007
zbozi	...->

Obrázek 15: Ukázka zobrazeni XML elementu v základním typu.

3.3.1.2 Agregační typ

Agregační typ představuje shluk XML elementů stejného jména v jedné entitě XML dokumentu.

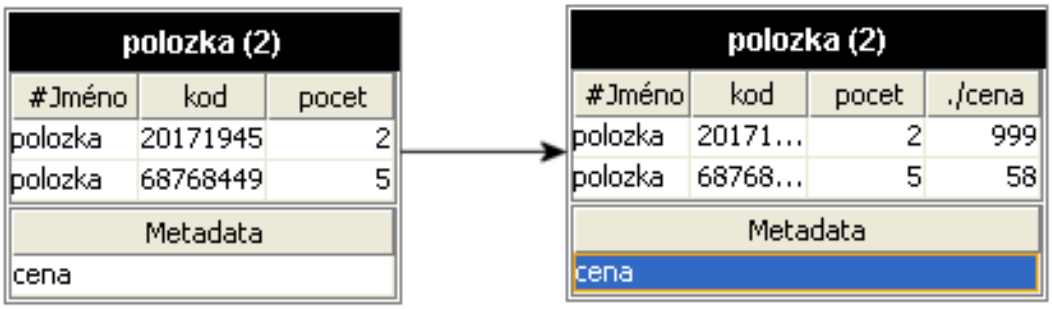
Obsahuje také dvě tabulky. Tabulka agregačních dat, obsahuje všechny XML elementy, který daný shluk obsahuje. Každý řádek tabulky představuje jeden XML element, případně v jednotlivých sloupcích atributy těchto XML elementů. Pokud některý element obsahuje konečnou hodnotu je vepsán do sloupce #text. Název XML elementu je napsán ve sloupci #Jméno. Ke konfliktu jména atributu u názvů sloupce #text a #Jméno nemůže dojít, jelikož specifikace XML nedovoluje u názvu atributu jako první znak použít #. Jednotlivé elementy lze dále otvírat do základního typu.

polozka (2)		
#Jméno	kod	pocet
polozka	20171945	2
polozka	68768449	5
Metadata		
cena		

Obrázek 16: Ukázka agregačního typu.

Druhá tabulka, tabulka metadat, obsahuje názvy všech XML subelementů, které představují v tomto případě metadata. Při dvojkliku na některý z řádků tabulky metadat, se tyto metadata vloží do tabulky agregačních dat a vypíšou se příslušné hodnoty k jednotlivým XML subelementům.

Na obrázku 17 je zobrazen interaktivní přesun metadat mezi data.

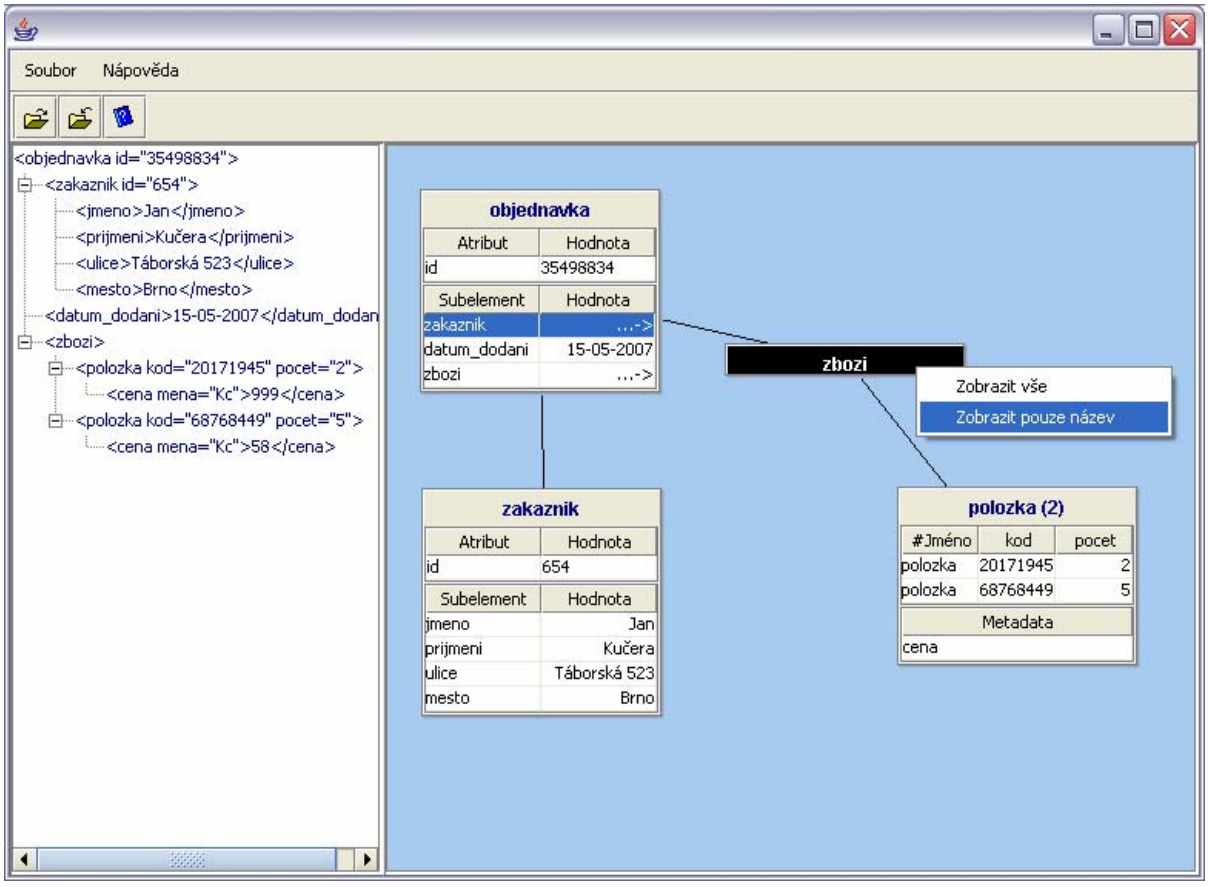


Obrázek 17: Ukázka aplikace metadat.

3.3.2 Možnosti zobrazení

Kromě plného zobrazení jednotlivých elementů, program umožňuje zobrazovat jen jména jednotlivých elementů. To se hodí zejména v případě velkých XML dokumentů, kdy chceme šetřit místo na zobrazovací ploše. Možnosti zobrazení lze dosáhnout vyvoláním popup menu, kliknutím pravého tlačítka myší na název typu elementu. Během držení levého tlačítka myší nad názvem XML elementu lze tímto XML elementem libovolně pohybovat po zobrazovací ploše.

Programová dokumentace se nachází v příloze 1.



Obrázek 18: Celkový pohled na aplikaci.

4 Možnosti rozšíření a pohled na další vývoj softwaru.

Tento program je ve fázi praktického a využitelného vizualizačního nástroje, ale je možné jej dále rozšiřovat a upravovat. Možností pro rozšíření se nabízí celá řada a na vývoji tohoto programu bych dále rád pracoval.

Velké rozšíření nabízí přímo samotné grafické uživatelské prostředí. Zatím neobsahuje, dnes již běžně používané, uživatelsky přívětivé technologie drag and drop (uchop a pusť), kterou lze využít pro otvírání XML subelementů. Dále by vhodným vylepšením bylo zdokonalení manipulace se samotnými elementy například označit více elementů najednou a pracovat s nimi apod.

Dalším vylepšením, které program zatím postrádá je možnost uložení otevřeného XML dokumentu v rámci aplikace, aby se uživatel mohl k zobrazovanému XML dokumentu vracet.

Možnosti rozšíření nabízí i samotná vizualizace. Agregační typ lze rozšířit o zobrazení matematických funkcí na základě typu hodnot v XML dokumentech například by šel počítat průměr nebo medián hodnot XML elementů a další.

Efektivněji pracovat s metadaty a nabízet více pohledů.

5 Závěr

Téma Interaktivní vizualizace jsem si vybral, protože jsem chtěl proniknout podrobněji do technologie XML. Lákalo mne se podílet na vývoji nástroje, který umožňuje alternativní pohled na XML dokumenty.

Během bakalářské práce jsem se podrobně seznámil z jazykem XML a technologiemi, které pracují s tímto jazykem. Potvrdilo se mi, že XML je velmi silný nástroj pro uchovávání strukturovaných dokumentů a dat.

Na základě poznatků o XML jsem vytvořil program, který interaktivně zobrazuje libovolné XML. Mimo klasického stromového zobrazení nabízí pohled, ve kterém elementy stejného jména a entity shlukuje do sebe. Umožňuje tedy přehledně zobrazit především XML dokumenty, které obsahují velké množství elementů stejného jména, ale různých atributů či hodnot.

Z volně dostupných modelovacích prostředí, které by šlo použít i pro vizualizaci XML dokumentů se až při jejich podrobnějším studiu neukázal žádný jako příliš vhodný pro další vlastní programování a ukázalo se, že nejvhodnějším řešením je vytvoření vlastního programu. Podařilo se mi naprogramovat aplikaci umožňující jednoduché zobrazení XML. Tento program je ve fázi praktického a využitelného vizualizačního nástroje a je možné jej dále rozšiřovat a upravovat.

Literatura

[1] BRADLEY, Neil. *XML : kompletní průvodce*. Brázda Jiří. Praha : Grada Publishing, 2000. 540 s. ISBN 80-7169-949-7.

[2] SKONNARD, Aaron, GUDGIN, Martin. *XML pohotová referenční příručka : Referenční příručka programátora ke XML, XPath, XSLT, XML Schema, SOAP a dalším*. Bittnerová Rút Lucie. Praha : Grada Publishing, 2006. 344 s. ISBN 80-247-0972-4.

[3] KOSEK, Jiří. *XML pro každého*. Praha : Grada Publishing, 2000. 164 s. ISBN 80-7169-860-1.

[4] KOSEK, Jiří. *XML : staronový formát* [online]. c1999 [cit. 2007-03-15]. Dostupný z WWW: <http://www.kosek.cz/clanky/xml/xml-historie.html>.

[5] KOSEK, Jiří. DocBook : Finální řešení pro vaši dokumentaci. *Softwarové noviny* [online]. 2002, č. 6 [cit. 2007-04-25]. Dostupný z WWW: <http://docbook.cz/clanky/uvod.html>

[6] TAUCHMAN, Vít. *Zpracování strukturovaných dokumentů*. [s.l.], 2006. 39 s. Diplomová práce.

[7] *W3C World Wide Web consortium* [online]. c1994-2007 [cit. 2007-04-20]. Dostupný z WWW: <http://www.w3.org/>.

[8] *JavaTM 2 Platform Standard Edition 5.0 : API Specification* [online]. c2004 [cit. 2007-04-23]. Dostupný z WWW: <http://java.sun.com/j2se/1.5.0/docs/api/>.

[9] *TouchGraph : Google Browser* [online]. c2007 [cit. 2007-05-01]. Dostupný z WWW: <http://www.touchgraph.com/TGGoogleBrowser.html>.

[10] *Graphviz* [online]. [2007] [cit. 2007-05-01]. Dostupný z WWW: <http://www.graphviz.org/>.

[11] *Prefuse : information visualization toolkit* [online]. [2006-] , on Thursday Apr 05, 2007 at 11:30 AM [cit. 2007-05-01]. Dostupný z WWW: <http://www.prefuse.org/>.

[12] *Hypergraph* [online]. [2003-] [cit. 2007-05-01]. Dostupný z WWW: <http://hypergraph.sourceforge.net/>.

[13] CLARK, James. *XSL Transformations (XSLT) Version 1.0.* [online]. 1999 [cit. 2007-05-03]. Dostupný z WWW: <http://www.w3.org/TR/xslt>.

[14] CLARK, James, DEROSE, Steve. *XML Path Language (XPath) Version 1.0* [online]. 1999 [cit. 2007-05-03]. Dostupný z WWW: <http://www.w3.org/TR/xpath>.

[15] SPERBERG-MCQUEEN, Michael, THOMPSON, Henry. *XML Schema* [online]. 2000 [cit. 2007-05-02]. Dostupný z WWW: <http://www.w3.org/XML/Schema>.

[16] JELÍNEK, Josef, SLAVÍK, Pavel. XML Visualization Using Tree Rewriting. *Proceedings of the 20th spring conference on Computer graphics 2004*. 2004, s. 65-72. Dostupný z WWW: <http://delivery.acm.org/10.1145/1040000/1037219/p65-jelinek.pdf?key1=1037219&key2=7855648711&coll=&dl=&CFID=15151515&CFTOKEN=61846188>.

[17] *Document Object Model* [online]. 2006- [cit. 2007-05-05]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/DOM>.

[18] *How to Use Tables* [online]. c1995-2007 [cit. 2007-05-05]. Dostupný z WWW: <http://java.sun.com/docs/books/tutorial/uiswing/components/table.html>.

[19] *Wikipedie: Otevřená encyklopedie: Java[on-line]*. c2007 [cit. 2007-05-10]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/Java>.

Seznam příloh

6 Přílohy

6.1 Příloha 1 – Programová dokumentace

6.1.1 Spuštění programu

Program se spouští pomocí dávkového souboru start_windows.bat v operačním systému Microsoft Windows nebo start_linux v operačních systémech Linux.

6.1.2 Ovládání programu

6.1.2.1 Lišta tlačítek (toolbar)

 Otevře dialogové okno pro otevření XML dokumentu (Ctrl+N).

 Zavře XML dokument (Ctrl+F4).

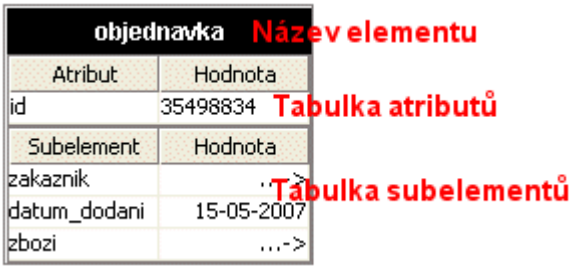
 Otevře tuto nápovědu.

6.1.2.2 Vizualizace XML

Základní typ elementu

Základní typ elementu představuje jeden konkrétní XML element. Obsahuje tabulku atributů a tabulku subelementů.

Každý subelement lze dvojklikem myši otevřít.



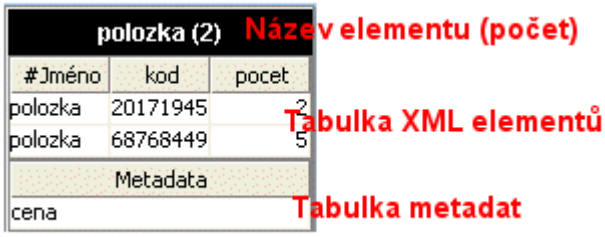
Agregační typ elementu

Agregační typ elementu shlukuje do sebe XML elementy stejného jména a entity. Obsahuje tabulku XML elementů a tabulku metadat.

Každý řádek tabulky představuje jednotlivý XML element.

Každý XML element lze dvojklikem myši otevřít.

Po dvojkliku na řádek tabulky metadat se konkrétní metadata vloží do tabulky XML elementů jako nový sloupec.



6.2 Příloha 2 – Projektová dokumentace

Součásti projektové dokumentace je ER diagram přiložený na samostatném listu. Na ER diagramu nejsou zobrazeny všechny funkce daných tříd. Zobrazeny jsou pouze hlavní funkce programu.

6.3 Příloha 3 – CD

Tato příloha obsahuje datový nosič CD, na kterém je celá práce v elektronické podobě.