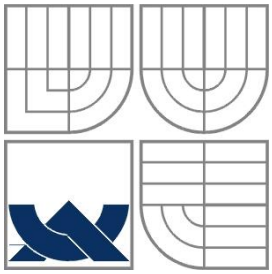
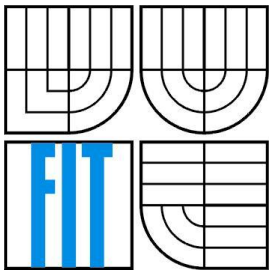


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



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

Ladislav Pospěch

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. Petr Chmelař

BRNO 2008

Zadání bakalářské práce/6869/2007/xpospe01

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2007/2008

Zadání bakalářské práce

Řešitel: **Pospěch Ladislav**
Obor: Informační technologie
Téma: **Interaktivní vizualizace XML**
Kategorie: Uživatelská rozhraní

Pokyny:

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

Literatura:

- Quin, L. W3C. *Extensible Markup Language (XML)*. 2006. Dostupný z: http://www.w3.org/XML/.
- Troelsen, A. *C# a .NET 2.0 profesionálně*. Brno : Zoner Press, 2006. 1197 s. ISBN 80-86815-42-0.
- Jelinek, J. - Slavik, P. XML visualization using tree rewriting. *Proceedings of the 20th spring conference on Computer graphics*. ACM, 2004. pp. 65-72. ISBN:1-58113-967-5.

Při obhajobě semestrální části projektu je požadováno:

- 1. a 2. bod zadání.

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

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

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

Vedoucí: **Chmelař Petr, Ing.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2007
Datum odevzdání: 14. května 2008

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: **Ladislav Pospěch**
Id studenta: 84177
Bytem: Zašová 395, 756 51 Zašová
Narozen: 18. 03. 1985, Olomouc
(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

Cílem této bakalářské práce bylo řešení problémů s vizualizací obecných XML dokumentů a vytvoření aplikace schopné interaktivně zobrazit obsah jakéhokoli takového dokumentu. Tato práce staví na standardech vytvořených konsorciem W3C pro definici a zpracování dokumentů XML a příbuzných technologií, jako je jazyk XPath.

Výsledem této práce je nová aplikace pro vizualizaci XML dokumentů v co nejjednodušší a nejucelenější podobě pro potřeby široké uživatelské obce. Vizualizační aplikace byla naprogramována v jazyce C# v prostředí Microsoft .NET.

Klíčová slova

XML, vizualizace XML, tabulky v XML, XML dokument, zobrazení XML, C#, .NET

Abstract

The goal of this bachelor thesis was solving problems connected with visualization of common XML documents and creating application, which can show contents of common document in an interactive way. This thesis builds on standards created by W3C consortium to make definition and manipulation with XML and related technologies like XPath language.

The result of entire work is brand new application for visualization XML documents in the most user-friendly and complex way to fulfill needs of users. Visualization application was programmed in C# language in Microsoft .NET Framework.

Key words

XML, XML visualization, XML in tables, XML document, viewing of XML, C#, .NET

Citace

Pospěch, Ladislav: Interaktivní vizualizace XML. Brno, 2008, bakalářská práce, FIT VUT v Brně.

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.

.....
Ladislav Pospěch
10. května 2008

Poděkování

Chtěl bych poděkovat vedoucímu své bakalářské práce Ing. Petru Chmelařovi za jeho cenné rady a předávání zkušeností v oblasti zpracování XML a souvisejících technologií.

Obsah

1 Úvod

Výsledkem činnosti počítačů a jejich uživatelů jsou data. Ať už je to obrázek nebo upravená fotografie, textový dokument se slohovou prací nebo účetní tabulka zachycující finanční toky podniku. Ve výčtu příkladů bych mohl jít ještě dále, důležité je, že takřka všechny tyto úkony produkují nějaká data. To s sebou nese několik problémů, z nichž nejdůležitější je tento: Data je třeba uchovat a následně je znova přečíst. Pro různá data existují různé formáty pro jejich uložení. Každá aplikace má možnost ukládat svá data do svého jedinečného a pro jiné aplikace nečitelného formátu. To samozřejmě vede k obrovskému množství formátů, které jsou navzájem nesourodé, a pro každý potřebujeme jeho „domovskou" aplikaci, abychom jej otevřeli. Nebylo by výhodné zredukovat toto množství formátů na co nejmenší počet, aby je mohly aplikace navzájem sdílet?

Jazyk XML (eXtensible Markup Language) se právě o takový přístup snaží. Byl navržen jako univerzální formát pro uložení hierarchických a tabulkových dat. Ač může každý dokument na první pohled vypadat jinak, jeho koncept je vždy stejný a proto čitelný pro všechny aplikace, které jej číst chtějí. Pokud programátor chce, aby jeho aplikace četla XML dokument z nějakého zdroje, stačí mu, aby si daný dokument otevřel a v aplikaci počítal s jeho rozestavením XML značek.

Takto řečeno vše vypadá jednoduše. Problém nastává, chceme-li vytvořit aplikaci, která si poradí s jakýmkoliv XML dokumentem. Formát dokumentů je sice stejný, ale jazyk XML je rozšiřitelný a jeho obsah je předem neodhadnutelný. Víme pouze, že v dokumentu jsou značky, kterými je dokument formátován a textová data, která jsou značkami uzavřena. Pokud jsme dokument předem neviděli, nevíme, jak se značky jmenují, kolik jich je a jak jsou navzájem uspořádány.

Řešení tohoto problému je cílem této bakalářské práce. Navrhnout a implementovat tokovou aplikaci, která dokáže zobrazit obsah XML dokumentu tak, aby si jej mohl přečíst běžný uživatel, který neví nic o značkovacích jazycích, jmenných prostorech ani znakových sadách. Práce se zabývá primárně datově orientovanými dokumenty, pro které je ideální tabulkové zobrazení. Na rozdíl od nich jsou textově orientované dokumenty pro tabulkové zobrazení zcela nepoužitelné a jejich podpora by znamenala vytvoření další aplikace podobného rozsahu jako výsledná aplikace této práce.

1.1 Struktura práce

V úvodu práce se nachází krátká předmluva představující základní problematiku programového zpracování XML dokumentů. Teoretická část je věnována základnímu popisu použitých technologií. První část praktického oddílu práce je věnována koncepci aplikace pro vizualizaci XML dokumentů s uvedením do základních problémů týkajících se návrhu. Druhou část praktického oddílu pak tvoří popis samotné implementace vizualizačního programu. V závěru jsou zhodnoceny výsledky celé práce a možnosti jejího rozšíření. V poslední části se nachází uživatelská příručka vytvořené aplikace a výpis obsahu přiloženého CD.

2 Teorie

Tato kapitola popisuje základy použitých technologií a základní problémy zobrazování obecného XML dokumentu. V úvodní části se také zmíníme o historii a základech jazyka XML včetně souvisejících technologií XML schématu a jazyka XPath. Dále nastíníme základy technologie Microsoft .NET Framework a jazyka C#.

2.1 Úvod do XML

2.1.1 Historie

Jazyk XML – Extensible Markup Language – tedy „rozšířitelný značkovací jazyk“, má své kořeny již v šedesátých letech dvacátého století. Cesta jeho vývoje byla dlouhá a vývoj probíhá i dnes. První konkrétní předchůdce XML spatřil světlo světa v roce 1986. Byl to jazyk SGML (Standard Generalized Markup Language) a v tomto roce jej firma IBM certifikovala jako normu ISO 8879. SGML byl vyvinut jako univerzální formát pro ukládání různých typů dokumentů a jejich následné čtení. Nicméně jeho flexibilita na úkor jednoduchosti způsobila jeho neatraktivnost a bylo třeba jej nahradit. První masivně rozšířená odnož SGML je jazyk HTML používaný dodnes jako formát webových stránek. Ani tento jazyk nebyl a není bez problémů. Již v jeho počátcích byl rozšiřován výrobci internetových prohlížečů, z nichž každý přidával jazyku jinou funkcionalitu. Zároveň ale jednotlivé prohlížeče nepodporovaly rozšíření ostatních prohlížečů. Proto bylo třeba zavést nějaký všeobecně uznávaný standard. O to se snažilo konsorcium W3C zavedením HTML verzí 2.0, 3.2 a 4.0. Ani to však nezabránilo výrobcům prohlížečů dále rozšiřovat jazyk oproti standardu a působit tak potíže zejména tvůrcům webů, kteří museli tuto různorodost brát v potaz. Konsorcium W3C následně přišlo s myšlenkou nového, flexibilnějšího nástupce HTML. Tím je jazyk XML, jehož finální verze byla zveřejněna v únoru 1998. [2]

2.1.2 Základy jazyka

Základem jazyka XML jsou značky (tagy). Podobně jako značky jazyka HTML (,
, <table> atd), jsou i značky jazyka XML uzavřeny v lomených závorkách <>. Na rozdíl od jazyka HTML, který rozeznává předepsanou sadu značek, v XML si může tvůrce dokumentu definovat značky sám.

Pomocí značek jsou v XML dokumentu definovány takzvané elementy. Element může mít libovolný počet atributů, jak si ukážeme níže. Každý element je tvořen počáteční a

koncovou značkou (<element></element>), případně pokud nemá element žádný obsah, jen atributy, může být zapsán jen jednou značkou (<element atribut=”atr”/>). Obsahem elementu může být text nebo další elementy. Takové elementy pak nazýváme potomky. Jeden element může mít libovolný počet potomků. Každý potomek má pouze jednoho rodiče.

První značka v dokumentu XML je povinná a obsahuje použitou verzi jazyka XML, případně kódování znaků. Vypadá například takto:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

Vidíme značku s názvem `xml` a dva její atributy. První udává verzi jazyka XML (`version="1.0"`) a druhý kódování znakové sady (`encoding="ISO-8859-1"`). V této značce také vidíme dva znaky `?` (otazník), které obklopují vnitřní část značky. Tyto znaky mají speciální význam právě pro tuto povinnou první značku.

Dalším povinným prvkem XML dokumentu je kořenový element. V tom je uzavřeno celé tělo dokumentu, a všechny ostatní elementy v dokumentu jsou jeho přímými či nepřímými potomky. V následujícím výpisu vidíme příklad XML dokumentu:

```
<?xml version="1.0" encoding="ISO-8859-2"?>
<kontakt xmlns="mujnamespace">
  <data id="1">
    <nazev>Lekarna</nazev>
      <adresa>
        <ulice>
          <Nazev lang="CZ">Zahnuta</Nazev>
          <CP>32</CP>
        </ulice>
      <mesto>Brno</mesto>
      <psc>603 00</psc>
    </adresa>
    <osoba>Pavel Apatykar</kosoba>
    <popis>kompletni prodej zdravotnickeho vybaveni a
leku</popis>
    <email> zijtezdrave.kupujteunas@lekarny.cz</email>
  </data>
</kontakt>
```

Na tomto příkladu je jasně vidět struktura dokumentu XML. Vidíme kořenový element s názvem „kontakt“ a jeho atribut s názvem „xmlns“. Tento atribut označuje primární jmenný prostor dokumentu, o kterém si povíme v praktické části práce. Dále vidíme element „data“, jež si můžeme představit jako jeden řádek tabulky. Sloupci tabulky jsou pak potomci elementu „data“. Každý potomek může mít také své potomky atd. Každá značka má také svojí koncovou značku (až na první řádek dokumentu, který je speciální) a celý dokument uzavírá koncová značka kořenového elementu „kontakt“.

2.1.3 XML Schéma

Již od počátků formátu XML bylo jasné, že vzhledem k pojetí a rozšiřitelnosti jazyka bude potřeba nějakého prostředku, díky kterému by bylo možno zkontrolovat správnost struktury dokumentu.

Původním formátem pro popis struktury XML dokumentu byl popisovací jazyk DTD (Document Type Definition). Tento měl jednu obrovskou nevýhodu: On samotný měl svůj vlastní formát. Chtěl-li programátor implementovat kontrolu XML dokumentů, musel počítat s dalším formátem navíc. Podle DTD bylo možné zkontrolovat správnost dokumentů po stránce rozložení elementů a atributů, nebylo však možné definovat datové typy uložených dat.

Nápravu neduhů DTD sjednalo opět konsorcium W3C, které navrhlo jazyk XSD (XML Schema Definition) a od března roku 2001 jej ustanovilo oficiálním doporučením a v říjnu 2004 publikovalo jeho revidovanou verzi.

Formát XSD schématu plně odpovídá specifikaci XML dokumentu. Z toho vyplývá, že XSD je vlastně XML dokument popisující jiný XML dokument. Toto sjednocení formátů vedlo k velkému zjednodušení kontroly dokumentů, navíc přibyla možnost uložit samotné schéma přímo do dokumentu a kontrolovat tak datovou část oproti schematické části, a to vše ve formátu XML. Mezi další výhody XSD paří možnost definovat datové typy jednotlivých elementů a jejich referenční vlastnosti, například definováním maximálního či minimálního počtu elementů jednoho jména v dané úrovni dokumentu nebo jejich vynucenou přítomnost. Více na [15].

2.1.4 XPath

Jazyk XPath je speciálním adresným a dotazovacím jazykem pro zpracování XML dokumentů. XPath dokáže najít uzly (elementy a atributy) v dokumentu XML pomocí jejich relativní nebo absolutní pozice v dokumentu. Jeho syntaxe může připomínat například adresářovou strukturu na pevném disku počítače. Jako výsledek XPath dotazu se ve většině případů očekává uzel nebo sada uzlů dokumentu. Jazyk XPath je velmi propracovaným nástrojem a při zpracování XML velmi usnadňuje orientaci v dokumentu a výběr požadovaných dokumentů. Více na [4].

2.2 Microsoft .NET Framework

Ve světě dnešních počítačů je stále drtivá většina těch, které běží pod operačním systémem Microsoft Windows. Technologie firmy Microsoft díky tomu mají dostatečně velké pole působnosti a byla by škoda tohoto potenciálu nevyužít.

2.2.1 Úvod

Microsoft .NET Framework je prostředí pro běh aplikací vyvinuté firmou Microsoft. Toto prostředí plní funkci prostředníka mezi aplikacemi a samotnou platformou operačního systému (Windows). Jeho výhodou je množství programovacích jazyků, které mohou díky jednotnému prostředí navzájem spolupracovat. Například jazyk ASP.NET je určen pro tvorbu internetových aplikací a dynamických stránek. Protože je součástí .NET Framework, můžeme tento jazyk zkombinovat s použitím jiných .NET jazyků, například C#. Z dalších programovacích prostředku pro .NET jmenujme například jazyky Visual Basic, Visual C++, J# a nejnovější přírůstek do rodiny jazyků F#.

Microsoft .NET Framework byl uveden na trh v roce 2002 ve verzi 1.0. Od té doby se vývoj prostředí nezastavil. Nejnovější verze – 3.5 – byla představena v roce 2007.

2.2.2 Vývojové prostředí

Vývojovým prostředím pro Microsoft .NET je Visual Studio, v poslední verzi s číslovkou 2008. Toto prostředí je vyvinuto přímo pro .NET firmou Microsoft, a proto nemá smysl používat jiných prostředků pro vývoj.

2.2.3 Jazyk C#

Pro implementaci mé bakalářské práce jsem zvolil jazyk C#. Důvodů je hned několik. Jazyk C# je přímo vytvořen pro platformu .NET a jako takový má pro spolupráci s ní nejlepší dispozice. Jak vyplývá i z jeho názvu, je jazyk postaven na základech jazyka C/C++, což usnadňuje programátorům ovládnutí jazyka, jelikož jeho základy již dobře znají z jeho předchůdců. Jazyk C# čerpá také z jazyka Java. C# je vysoce úrovňový objektově orientovaný programovací jazyk a podporuje dědičnost. Z jazyka Java čerpá aspekty objektového programování a z jazyka C je odvozena jeho syntaxe.

Jazyk C# byl společně s platformou .NET představen v roce 2002. Tehdy ve verzi 1.0. Dodnes se jazyk stejně jako platforma .NET vyvíjí a v roce 2007 byla představena zatím jeho poslední verze 3.0. Více na [5].

2.3 Zpracování XML v prostředí .NET Framework

XML je jako technologie pro výměnu dat úzce a úplně integrována do prostředí .NET Framework. Toto prostředí má pro zpracování XML dokumentů několik knihoven, jejichž hlavní a nadřazenou knihovnou je `system.xml.dll`. V této knihovně, respektive jmenném prostoru `system.xml`, nalezneme prostředky potřebné pro veškerou práci s XML daty. Je zde implementována například podpora pro stromové zpracování XML (DOM – Document Object Model) nebo pro událostmi řízené zpracování (SAX – Simple API for XML). Tato označení se sice v prostředí .NET Framework nepoužívají, ale podle typu zpracování je můžeme k těmto modelům připodobnit. Rozsah podpory zpracování XML je úplný a zaštiťuje všechny potřebné prostředky. Všechny standardy týkající se XML, které jsou implementovány v .NET Framework vidíme v tabulce 1. Více v [9].

Tabulka 1 – Standardy W3C, podporované v .NET Framework [9]

Standard	Odkaz
XML 1.0	http://www.w3.org/TR/1998/REC-xml-19980210
Jmenné prostory XML	http://www.w3.org/TR/REC-xml-names
XML Schema	http://www.w3.org/TR/xmlschema-2
DOM Level 1 a Level 2 Core	http://www.w3.org/TR/DOM-Level-2
XPath	http://www.w3.org/TR/xpath
XSLT	http://www.w3.org/TR/xslt
SOAP 1.1	http://www.w3.org/TR/SOAP

3 Koncepce a návrh aplikace

Cílem implementace je vytvořit grafický nástroj pro interaktivní zobrazení XML dokumentů obecného typu. Velký důraz je kladen na jednoduchost a uživatelskou přívětivost. Požadavky jsou následující:

- Zobrazení jakéhokoliv obecného XML dokumentu
- Uživatelská přívětivost aplikace
- Možnost exportu dokumentu do jiných formátů (HTML, Excel)

Program není nástrojem pro tvorbu nových nebo editaci existujících XML dokumentů. Jeho hlavní činností je zobrazit XML dokument a jeho data tak, aby byl srozumitelný běžnému uživateli.

Program je určen k zobrazování datově orientovaných dokumentů, nejlépe tabulkových dat. Jeho použití je však univerzální a je schopen zobrazit také textově orientované dokumenty, ovšem opět v tabulkové reprezentaci, což nemusí na uživatele působit příjemně.

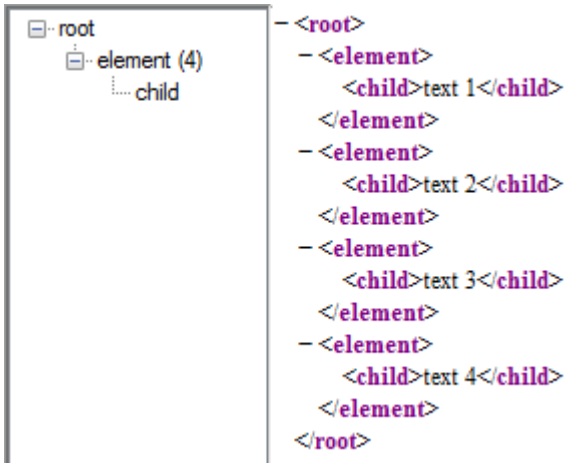
3.1 Zobrazení XML dokumentu

Prvním úskalím při řešení vizualizace XML dokumentů je forma jejich zobrazení v aplikaci. Bylo třeba navrhnout uživatelské rozhraní tak, aby zobrazovalo všechny potřebné informace a přitom si zachovalo jednoduchost a snadnou orientaci.

3.1.1 Zjednodušené stromové zobrazení dokumentu

Jako základ při zobrazení XML dokumentu jsem zvolil stromové zobrazení v levé části aplikačního okna. Umístění toho to prvku je standardní a většina uživatelů je na něj zvyklá z jiných aplikací (například Průzkumník Windows). Když uvážíme, že dokument XML je stromovou strukturou reprezentován, je tento krok logický. Zobrazením celého obsahu XML, nebo jen všech jeho uzlů, by se stal program nepřehledným. Proto bylo třeba zobrazit strom dokumentu ve zjednodušené formě. Musíme brát v potaz, že mnoho potomků jednoho uzlu je stejného jména a stejného typu. Takovéto uzly nemusíme ve stromu zobrazovat všechny. Stačí

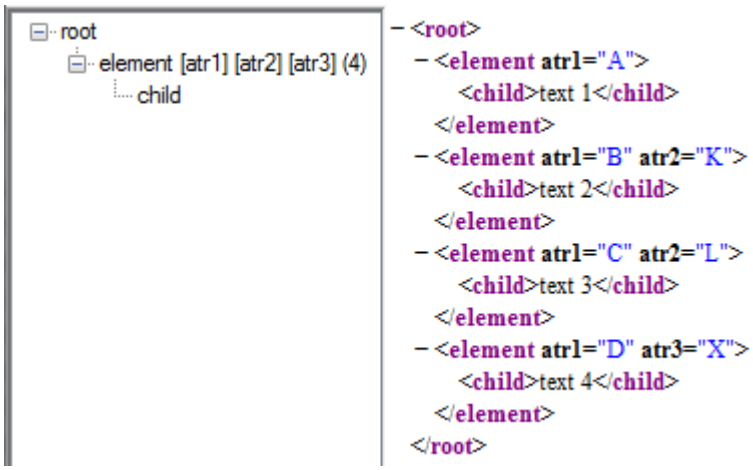
jeden záznam s upozorněním, že stejných nebo podobných (mohou se lišit svými atributy) uzlů je více. Síla tohoto přístupu je vidět na obrázku 1, kde vlevo vidíme příklad XML dokumentu a vpravo jeho stromovou reprezentaci v programu. Strom zůstává přehledný a uživatel má o dokumentu jasnou představu. Zobrazení zjednodušeného stromu můžeme připodobnit ke schématu XML dokumentu, kde také nenajdeme žádné názvy elementů, pouze jejich strukturu, kterou je dokument tvořen.



Obrázek 1 – zjednodušené zobrazení stromu dokumentu

Na obrázku 1 vidíme ve stromovém zobrazení za názvem uzlu „element“ v závorce číslo 4, což znamená, že uzel „root“ má čtyři potomky jména „element“. Právě ve zobrazení uzlů, jichž je více v jedné úrovni dokumentu, tkví síla implementované aplikace. Více informací se dozvíme v následující podkapitole.

Pokud mají elementy dokumentu nějaké atributy, ve stromové reprezentaci dokumentu opět nepotřebujeme znát jejich hodnoty. Zobrazení hodnot by naopak strom znepřehlednilo a ten by pak neplnil svůj účel. Má-li element atributy, jsou vypsány za jméno elementu do hranatých závorek, jak vidíme na obrázku 2. U elementů se také může stát, že v jedné skupině stejného jména najdeme různé atributy. V takovém případě jsou ve stromu dokumentu vypsány všechny atributy, které elementy mají. Přesně to vidíme na obrázku 2.



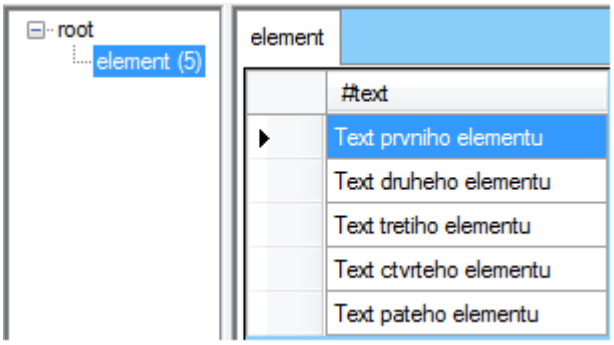
Obrázek 2 – zobrazení atributů ve stromu

Na tomto obrázku vidíme, že ve zdrojovém dokumentu mají sice všechny elementy atribut se jménem „atr1“, ale „atr2“ už mají jen druhý a třetí element. Poslední, čtvrtý element má navíc atribut „atr3“. S podobnou situací se můžeme v XML dokumentu běžně setkat. Proto uživatel ve stromovém zobrazení vidí všechny atributy.

3.1.2 Zobrazení elementů a jejich obsahu

Pro zobrazování obsahu elementu slouží druhá část aplikačního okna. To je tvořeno plochou, na které se při zobrazování elementů vytvářejí záložky (panely). Každý panel obsahuje reprezentaci jednoho elementu, nebo jedné skupiny elementů stejného jména a jejich podelementů.

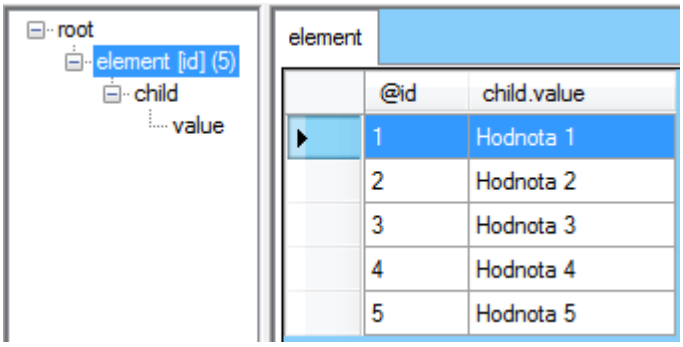
Při zobrazování elementů je opět kladen hlavní důraz na přehlednost a jednoduchost ovládání. Pro výběr jednoho nebo skupiny elementů, které chceme zobrazit, použijeme stromovou reprezentaci dokumentu, která se zobrazí automaticky po jeho načtení. Uživateli stačí, aby poklepal ve stromu dokumentu na element, který chce zobrazit. Ten se automaticky zobrazí v tabulce v pravé části aplikace. Pokud uživatel zvolil element, který je v daném kontextu samostatný (nemá sourozence stejného jména), zobrazená tabulka bude mít pouze jeden řádek. Sloupce tabulky budou tvořit atributy elementu, případně jeho potomci, jejich atributy atd. Zobrazení jednoduchého seznamu elementů vidíme na obrázku 3.



Obrázek 3 - zobrazení jednoduchého seznamu elementů

Obrázek 3 ukazuje, že zobrazené elementy nemají žádné potomky ani atributy, pouze text. Proto nemá sloupec v tabulce nějaké konkrétní jméno, ale je označen rezervovaným názvem `#text`. Dalším příkladem je zobrazení elementů, které mají atributy. Atribut elementu vytvoří při zobrazení v tabulce nový sloupec. Že se jedná o atribut, poznáme podle předpony `@` (zavináč).

Může ale nastat situace, že zvolený element má nějaké potomky. V takovém případě se v tabulce promítne obsah všech elementů, jejich atributů, případně všech jejich potomků a jejich atributů. Přímým potomkům zvoleného elementu je v tabulce vytvořen sloupec označený jejich jménem. V případě, že by i oni měli potomky, je jim vytvořen v tabulce další sloupec. Před názvem každého takového elementu je název jeho rodiče. Názvy jsou odděleny tečkou. Jméno sloupce v tomto případě nemůže být shodné se jménem elementu, který reprezentuje, protože by to bylo matoucí pro uživatele. Nevěděl by, ve které úrovni potomků prvního elementu se vlastně element nachází. Jak taková složitější tabulka vypadá, vidíme na obrázku 4.

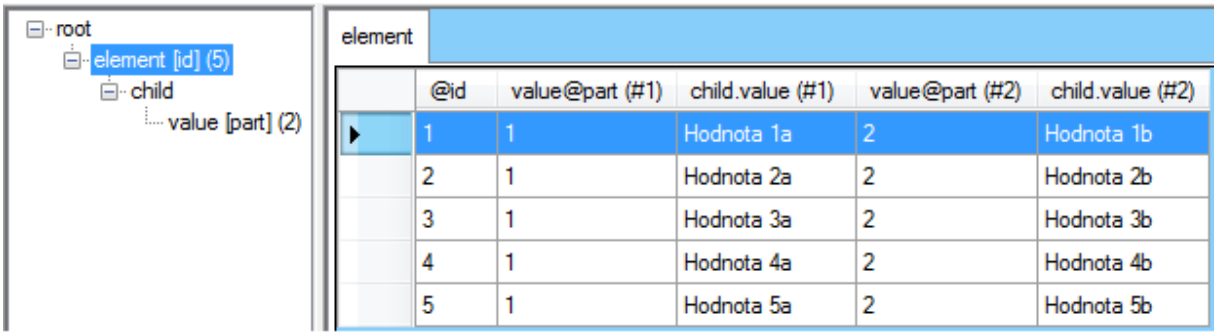


Obrázek 4 - zobrazení složitějšího seznamu elementů

Na obrázku vidíme v tabulce dva sloupce. V prvním jsou hodnoty atributů samotného zobrazovaného elementu a ve druhém, označeném `child.value`, jsou hodnoty elementu `value`, který je potomkem elementu `child`, který je přímým potomkem elementu

`element`. Element `child` nemá žádnou svou vlastní hodnotu, ani atribut. Proto pro něj samotného nemá tabulka žádný sloupec.

Poslední možností, která může v běžném XML dokumentu nastat, je případ, kdy zobrazujeme element, který má více přímých potomků stejného jména. Tato situace je stejně jako předešlé řešena s maximálním ohledem na jednoduchost. Pro každý takový element je v tabulce vytvořen nový sloupec, jehož název podléhá pravidlům popsaným výše. Jediný rozdíl spočívá v tom, že za názvem sloupce je v závorce společně se znakem "#" (mříž) uvedeno pořadové číslo elementu. Jak taková tabulka vypadá, ukazuje obrázek 5.

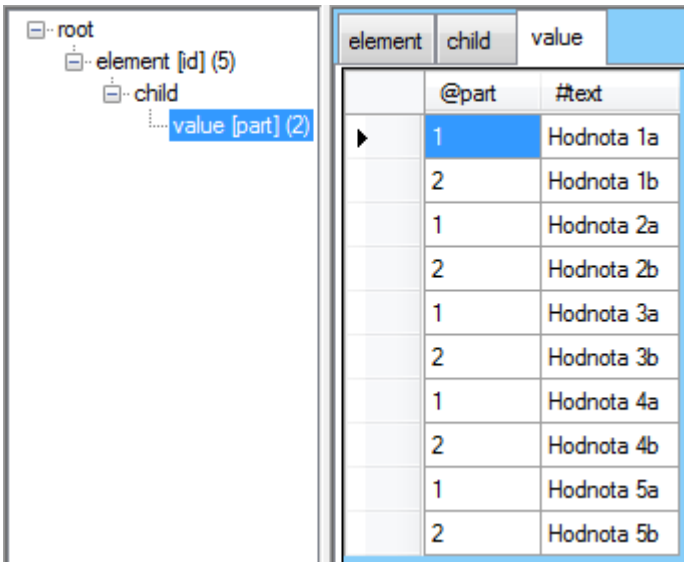


Obrázek 5 - zobrazení vícenásobných potomků

Kombinací všech výše zmíněných pravidel pro zobrazování elementů, jejich atributů a potomků je program schopen zobrazit jakýkoliv element, seznam elementů nebo vybraný podstrom elementů do jedné, přehledné tabulky. V dokumentu také může nastat situace, kdy uživatel chce zobrazit potomky elementu pouze do určité úrovně. Maximální úroveň zobrazených uzlů může uživatel jednoduše nastavit. Tím je zachována požadovaná přehlednost a flexibilita zobrazení.

3.1.3 Zobrazení více elementů najednou

Jak už bylo řečeno výše, pro každý element nebo seznam elementů, který chceme zobrazit, se vytvoří jeden panel se záložkou, na kterém je tabulka s obsahem elementu. V praxi to znamená, že uživatel může mít najednou otevřeno libovolné množství tabulek a díky záložkám bude mít stále přehled, které tabulky jsou otevřeny. V případě potřeby bude moci kteroukoli tabulku přenést do popředí. Část aplikace s několika otevřenými tabulkami vidíme na obrázku 6.



Obrázek 6 - záložky s otevřenými tabulkami

3.2 Ovládání aplikace

Ovládání programu bylo navrženo v duchu celé aplikace, tedy s ohledem na komfort uživatele. Jak už bylo řečeno výše, nejdůležitějším ovládacím prvkem je stromové zobrazení. Z něj je možné zobrazit obsah elementů pouhým poklepáním.

Dalším ovládacím prvkem je aplikační menu se standardním umístěním nahoře. V tomto menu uživatel nalezne klasické příkazy například pro otevření souboru, tisk či zavření aplikace. Další možnosti menu budou popsány v kapitole 7.1 Uživatelská příručka programu XML Dabster.

Vizuálně nejzřetelnějším ovládacím prvkem je ovládací panel. Ten poskytuje nejrychlejší přístup k některým akcím. Použitím tlačítek v ovládacím panelu může uživatel provést tyto operace: otevřít dokument, zkontrolovat dokument podle schématu, vytisknout tabulku, exportovat tabulku do samostatného XML dokumentu, dokumentu Microsoft Excel a stránky HTML. Poslední tlačítko slouží pro zobrazení okna „O programu“.

3.2.1 Kontextová menu

Důležitým prvkem ovládání aplikace jsou kontextová menu. V aplikaci jsou dvě různá kontextová menu, vyvolaná stiskem pravého tlačítka myši nad určitou komponentou.

První kontextové menu je velice jednoduché a slouží pouze pro uzavírání panelů s tabulkou.

Druhé kontextové menu je vyvoláno stiskem pravého tlačítka myši nad hlavičkami sloupců v tabulce hodnot elementů. Toto menu slouží kromě jiného také k mazání, či schovávání sloupců, nebo naopak jejich navracení do tabulky. Díky této funkci má uživatel možnost upravit si zobrazenou tabulku jak pro pohled (schováním některých sloupců), tak i pro export (odstraněním sloupců). S pomocí těchto možností si uživatel zobrazí opravdu jen ta data, která potřebuje, a pracovní plochu mu nezabírají nepotřebná data. Detailně se funkcemi kontextových menu zabývá kapitola 7.1 Uživatelská příručka programu XML Dabster.

4 Implementace programu XML Dabster

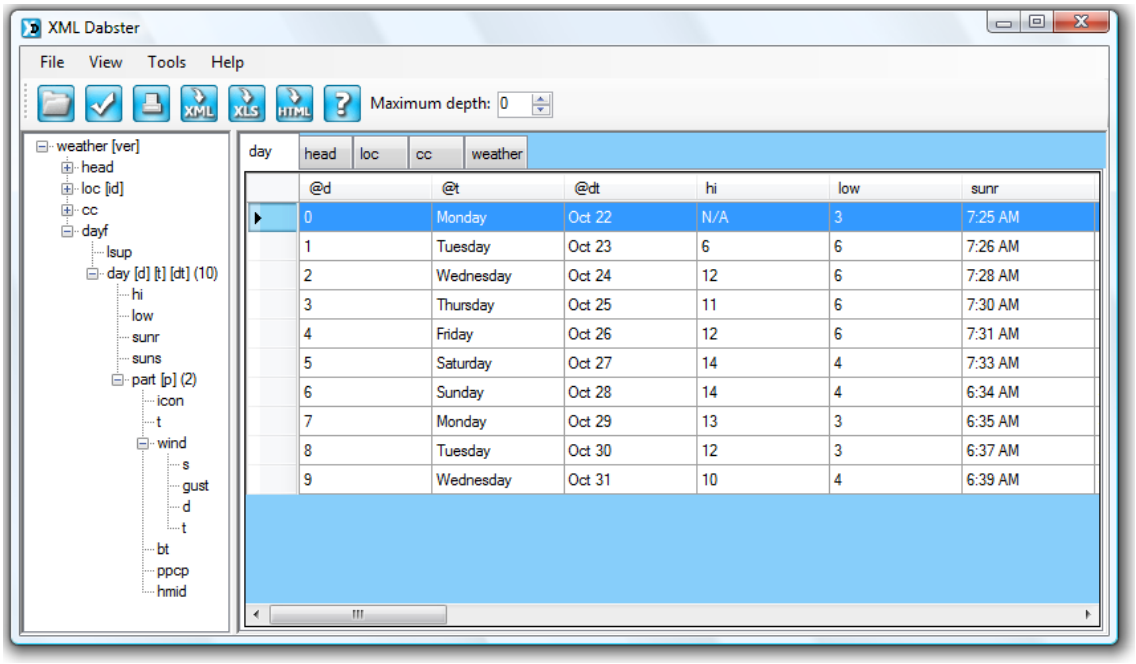
Aplikace XML Dabster je výsledkem zpracování problematiky tématu této práce. Je implementována v jazyce C# spadajícím do rodiny jazyků .NET Framework. Jazyk C# byl v době implementace ve verzi 3.0, běhové prostředí .NET Framework ve verzi 3.5. Detailněji o těchto technologiích v kapitole 2.2 Microsoft .NET Framework.

4.1 Základní zpracování XML dokumentu

Jako základ pro zpracování XML dokumentu je v programu použita třída `XmlDocument` ze jmenného prostoru `System.Xml`. Tato třída odráží vlastnosti modelu DOM, jak jej specifikuje W3C konsorcium. Třída `XmlDocument` uchovává celou strukturu XML dokumentu v paměti. To je sice náročnější na paměť, ale pro účely interaktivní vizualizace je to ideální řešení. Na rozdíl od sekvenčního přístupu k dokumentu, kdy by se musel dokument mnohokrát znova procházet, byla zvolena možnost náhodného přístupu, jak jej umožňuje model DOM a třída `XmlDocument`.

4.2 Základ uživatelského rozhraní

Program XML Dabster je grafickou aplikací. Jeho nedílnou součástí je grafické uživatelské rozhraní (GUI). Jeho základ je tvořen třemi komponentami ze standardní sady prostředí Microsoft .NET. Základní obrazovku programu s otevřeným dokumentem vidíme na obrázku 7.



Obrázek 7 - XML Dabster - zobrazení dokumentu weather.xml

Půjdeme-li shora, první komponentou je aplikační menu. To je reprezentováno komponentou `MenuStrip`. Ta poskytuje dostatečné možnosti a komfort pro tvorbu programového menu.

Další komponentou je panel nástrojů. Použitá komponenta `ToolStrip` je opět jednoduchým a dostačujícím prostředkem pro vytvoření vhodného panelu nástrojů. Aplikace v tomto případě využívá pouze nástrojová tlačítka.

Komponentou zabírající většinu aplikačního okna je `SplitContainer`. Tento prvek rozděluje zbylý prostor aplikačního okna na dvě vzájemně propojené části. `SplitContainer` se skládá ze dvou částí nazývaných `Panel`. První (levý) panel je použit pro umístění stromového zobrazení dokumentů a druhý panel slouží pro zobrazení tabulek elementů. Mezi panely je kolmý rozdělovač neboli `Splitter`. Ten může uživatel libovolně posunout, a tím přerozdělit šířku aplikačního okna mezi oběma panely.

4.3 Stromové zobrazení dokumentu

V pravém panelu `SplitContaineru` je umístěna komponenta `TreeView`, jenž se stará o stromovou reprezentaci otevřeného XML dokumentu. Vytvoření prezentovaného stromu je jednou ze základních činností celé aplikace. Jak bylo řečeno v kapitole 3.1 Zobrazení XML dokumentu, je samotný dokument reprezentován stromově. Pokud bychom načetli celý

dokument a jeho strukturu předali komponentě `TreeView`, stal by se strom nepřehledným a obsahoval by mnoho uživatelsky nezajímavých informací. Proto bylo třeba před zobrazením stromu dokument zpracovat a strom zobrazit v jeho zjednodušené formě. To je provedeno následujícím způsobem:

Program prochází načtený dokument reprezentovaný třídou `XmlDocument`. Jelikož chceme mít strom zjednodušený, algoritmus si vůbec nevšímá hodnot jednotlivých uzlů a zabývá se pouze jejich jmény, pro které vytváří jejich obraz ve stromu. Každý zobrazený element je ve stromu komponenty `TreeView` reprezentován typem `TreeNode`.

Prvním problémem při vytváření stromu je nutnost nevypisovat každý element jedné skupiny, ale označit jej jako skupinu elementů a pro tu vytvořit reprezentaci ve stromu. Algoritmus kontroluje, zda má právě zpracovávaný uzel sourozence stejného jména, a pokud ano, zachycuje tuto eventualitu počtem stejných sourozenců. V případě že zjistí, že žádný další sourozenec stejného jména neexistuje, vytvoří reprezentaci elementu ve stromu s označením počtu elementů stejného jména.

Další problém se vztahuje k atributům skupiny elementů. Ty jsou u elementů nepovinné a v jedné skupině elementů se mohou různit. V takovém případě jsou do stromu za název elementu vyznačeny všechny atributy, které se u elementů vyskytují. K nalezení všech těchto atributů je použit jazyk XPath.

Pokud má element nějaké potomky, algoritmus vytvoří ve stromové reprezentaci dětský uzel podobně jako je tomu v načítaném dokumentu. Pro potomky elementu probíhá stejná procedura se stejnými pravidly jako pro jejich rodiče. Algoritmus pracuje rekurzivně. Tím je zaručena správnost výsledného stromu pro jakoukoliv hloubku zanoření uzlů a jejich potomků.

4.4 Zobrazení tabulek hodnot elementů

Vůbec hlavním cílem aplikace XML Dabster je interaktivní zobrazení hodnot uzlů XML dokumentu. Tuto činnost obstarává v aplikaci několik komponent a o zpracování se stará několik algoritmů.

4.4.1 Hlavní prostředí pro zobrazení tabulek

Hlavní komponentou, která zobrazování tabulek zastřešuje, je komponenta `TabControl`. Ta sama o sobě slouží jen pro sdružování panelů se záložkami, které se v prostředí .NET nazývají `TabPages`. Základem pro zobrazení tabulek je tedy kolekce objektů

TabPage. Panely jsou na své rodičovské komponentě vytvářeny dynamicky, podle toho, kolik tabulek chce mít uživatel otevřených. Panelů může být otevřeno jakékoliv množství a díky víceřadému zobrazení jejich záložek bude pracovní prostor vždy přehledný.

4.4.2 Zobrazení tabulky

Samotné zobrazení tabulky hodnot elementu/ů je realizováno komponentou DataGridView. Ta je implicitní komponentou prostředí .NET pro zobrazování tabulkových dat. Jako taková je prostředím velmi podporována, ať už množstvím metod, či možnostmi pro kooperaci s jinými komponentami a třídami prostředí .NET.

V aplikaci připadá jedna tabulka DataGridView na jeden záložkový panel TabPage. Komponenta DataGridView sama o sobě neudržuje žádná data, je pouze vizuálním prvkem, který zobrazuje data z jiného datového zdroje. Jako datový zdroj je v aplikaci použita třída DataSource. Detaily jejího plnění budou specifikovány v kapitole 4.5 Fyzická reprezentace dokumentu. Pro práci s tabulkou je implementováno několik metod, jež dovolují schovávat nebo odstraňovat její sloupce. Schováním slupce rozumíme jeho odstranění z vizuální reprezentace tabulky. Fyzicky je sloupec stále přítomen. Odstraněním sloupce rozumíme jak jeho odstranění z vizuální reprezentace tabulky, tak z její fyzické reprezentace – v tomto případě jeho odstranění s tabulky ve zdroji dat, tedy DataSetu. Rozdíl mezi těmito dvěma úkony je důležitý zejména pro tisk nebo export tabulky, kdy schované sloupce budou exportovány, zatímco odstraněné nikoliv.

4.5 Fyzická reprezentace dokumentu

Po otevření XML dokumentu je jeho obsah předán instanci třídy XmlDocument, která uchová jeho nezměněný obsah. Abychom byli schopni zobrazit tabulky dat elementů ve formě, kterou potřebujeme, je nutné data přeuspořádat. Pro tento účel slouží instance třídy DataSet, která udržuje kolekci tříd DataTable. Pro každý zobrazovaný element je vytvořena tabulka DataTable ve zdroji dat – DataSetu. Ten je spojen s vizuálními komponentami DataGridView.

Vytvoření datové tabulky ve zdroji dat podléhá stejně jako tvorba stromu dokumentu jistým pravidlům, popsaným v kapitole 3.1 Zobrazení XML dokumentu. Pro výběr všech elementů, které je třeba do tabulky promítnout, posloužil výborně jazyk XPath. Pomocí pozice zvoleného elementu ve stromu dokumentu je vytvořen XPath řetězec, podle kterého je

nalezen jeden nebo skupina odpovídajících elementů. Výsledek dotazu je předán metodě, která plní nově vytvořenou datovou tabulku. Pro každý element či atribut a jejich potomky je vytvořen sloupec v tabulce, který je naplněn daty XML dokumentu. Na konci celého procesu je výsledná tabulka předána nově dynamicky vytvořené komponentě `DataGridView` jako její zdroj dat.

4.6 XPath a jmenné prostory XML

Třída `XmlDocument` poskytuje pro zpracování XPath dotazů metodu `SelectNodes()`, která podle zadaného XPath výrazu vyhledá jeden nebo více elementů. Během implementace použití jazyka XPath pro vyhledávání elementů v dokumentu bylo nutno vyřešit problém se jmennými prostory XML (více na [7]). Pokud vezmeme v potaz obyčejný XML dokument, ve kterém nejsou jmenné prostory (namespaces) definovány, funguje vykonávání dotazů XPath v prostředí .NET bez problémů. Pokud ale otevřeme dokument, ve kterém jsou jmenné prostory definovány, kde kromě primárního jmenného prostrou to mohou být i další jmenné prostory, vyniká při vykonávání XPath dotazů potřeba tyto jmenné prostory definovat. Pro uchování jmenných prostor XML je v prostředí .NET třída `XmlNamespaceManager`. Tato třída uchovává hlavní i vedlejší jmenné prostory a při volání XPath výrazu je nutno ji předat jako druhý parametr. Pokud má načítaný dokument definovány jmenné prostory, je jejich načtení nutností a jednou z prvních operací po otevření dokumentu.

4.7 Export XML dokumentu do jiných formátů

Aplikace XML Dabster nabízí export zobrazené tabulky do tří různých formátů. Tím prvním je samotný formát XML, kdy je vytvořen nový soubor, ve kterém je uložena pouze jedna datová tabulka. Druhým formátem je formát Microsoft Excel a posledním je webová stránka v jazyce HTML.

4.7.1 Export tabulky do samostatného XML dokumentu

Export do samostatného XML dokumentu využívá možností třídy `DataSet` respektive její dceřiné třídy `DataTable`. Ta umožňuje zapsat zvolenou datovou tabulku přímo do XML

souboru na zvolenou cestu. Prostředí .NET poskytuje pro tuto činnost dostatečnou podporu a není nutná další implementace.

4.7.2 Export tabulky do formátu MS Excel

Převod tabulky do formátu Microsoft Excel aplikací XML Dabster není plnohodnotnou konverzí dat do formátu xls. Program využívá vlastností tabulkového procesoru Excel, který je od verze 2003 schopen načíst soubory XML. Tato funkce je velice propracovaná a při otevření takového souboru vytvoří v hlavičkách sloupců i filtry dat. To nám umožňuje nezabývat se plnohodnotnou konverzí a využít možností stávajícího softwaru. Aplikace XML Dabster uloží XML reprezentaci tabulky do souboru s příponou xls, který je implicitně otevírán v programu MS Excel. Exportovanou tabulku otevřenou v programu Excel vidíme na obrázku 8.

	A	B	C	D	E	F	G	H	I	J	K
1	@d	@t	@dt	hi	low	sunr	suns	part@p (#1)	part.icon (#1)	part.t (#1)	wind.s (#1)
2	0	Monday	Oct 22	N/A	3	7:25 AM	5:51 PM	d	44	N/A	N/A
3	1	Tuesday	Oct 23	6	6	7:26 AM	5:49 PM	d	12	Rain	27
4	2	Wednesday	Oct 24	12	6	7:28 AM	5:47 PM	d	11	Light Rain	14
5	3	Thursday	Oct 25	11	6	7:30 AM	5:46 PM	d	11	Light Rain	10
6	4	Friday	Oct 26	12	6	7:31 AM	5:44 PM	d	11	PM Light Rain	16
7	5	Saturday	Oct 27	14	4	7:33 AM	5:42 PM	d	9	Drizzle	11
8	6	Sunday	Oct 28	14	4	6:34 AM	4:40 PM	d	30	Partly Cloudy	13
9	7	Monday	Oct 29	13	3	6:35 AM	4:38 PM	d	30	Partly Cloudy	18
10	8	Tuesday	Oct 30	12	3	6:37 AM	4:36 PM	d	11	Light Rain	13
11	9	Wednesday	Oct 31	10	4	6:39 AM	4:35 PM	d	11	Light Rain	11

Obrázek 8 - část exportované tabulky v programu MS Excel 2007

4.7.3 Export tabulky do webové stránky HTML

Funkce převodu tabulky do podoby webové stránky v jazyce HTML je v programu XML Dabster nejpropracovanější exportní funkcí. Vytvářená webová stránka má předem danou hlavičku a některé hodnoty, jako je styl tabulky, typ písma a zakončení stránky. Dynamicky generovanou část tvoří kód, který je ve výsledném souboru mezi značkami `<table>` a `</table>`, které označují začátek a konec tabulky. Kromě toho je na stránce generován už jen nadpis stránky a nadpis tabulky. Při generování obsahu tabulky jsou postupně tvořeny řádky uzavřené do značek `<tr>` a `</tr>`. Mezi těmi jsou pak generovány obsahy jednotlivých sloupců tabulky, uzavřené do značek `<td>` a `</td>`.

V tomto ohledu by bylo vhodné rozšířit export o možnosti nastavení vzhledu výsledné stránky. Tomuto tématu je věnována kapitola 5.2 Možnosti dalšího vývoje.

Exportovanou tabulku zobrazenou internetovým prohlížečem vidíme na obrázku 9.

weather.dayf.day

@d	@t	@dt	hi	low	sunr	suns	part@p (#1)	part.icon (#1)	part.t (#1)	wind.s (#1)	wind.gust (#1)	wind.d (#1)
0	Monday	Oct 22	N/A	3	7:25 AM	5:51 PM	d	44	N/A	N/A	N/A	N/A
1	Tuesday	Oct 23	6	6	7:26 AM	5:49 PM	d	12	Rain	27	N/A	29
2	Wednesday	Oct 24	12	6	7:28 AM	5:47 PM	d	11	Light Rain	14	N/A	46
3	Thursday	Oct 25	11	6	7:30 AM	5:46 PM	d	11	Light Rain	10	N/A	72
4	Friday	Oct 26	12	6	7:31 AM	5:44 PM	d	11	PM Light Rain	16	N/A	76
5	Saturday	Oct 27	14	4	7:33 AM	5:42 PM	d	9	Drizzle	11	N/A	84
6	Sunday	Oct 28	14	4	6:34 AM	4:40 PM	d	30	Partly Cloudy	13	N/A	108
7	Monday	Oct 29	13	3	6:35 AM	4:38 PM	d	30	Partly Cloudy	18	N/A	132
8	Tuesday	Oct 30	12	3	6:37 AM	4:36 PM	d	11	Light Rain	13	N/A	127
9	Wednesday	Oct 31	10	4	6:39 AM	4:35 PM	d	11	Light Rain	11	N/A	349

Obrázek 9 - část tabulky exportované do HTML

4.8 Validace dokumentu

Validace dokumentu probíhá jednoduchým načtením zvoleného XML dokumentu, a následně jeho XML schématu s příponou xsd. Při úspěšném ověření validity dokumentu je uživateli zobrazena informativní zpráva. Při neúspěšné validaci je zobrazeno chybové hlášení s vysvětlením chyby validace.

Implementace validace XML dokumentu byla převzata z [8].

4.9 Tisk tabulky

Při tisku datové tabulky je možno zvolit několik možností tisku a následně zobrazit náhled před samotným tiskem. Více o tomto tématu v kapitole 7.1 Uživatelská příručka programu XML Dabster.

Implementace tisku komponenty `DataGridView` byla převzata z [7].

4.10 Změny v implementaci

Tato kapitola se zabývá změnami v původně plánovaném návrhu implementace. Od popsaných prvků bylo upuštěno buď z důvodu výsledků jejich funkcí, které nebyly pro implementaci použitelné, nebo z důvodu nutnosti implementovat příliš složité algoritmy, které mohly být bez vlivu na výsledek nahrazeny.

4.10.1 Využití třídy XmlDataDocument

První změnou v původně plánované implementaci byla nutnost nahradit zpracování tabulek, při kterém se využívá spojení instance třídy `XmlDataDocument` s instancí třídy `DataSet`. `XmlDataDocument` je třída odvozená od třídy `XmlDocument`. Její výhodou je funkcionalita, díky které kooperuje se třídou `DataSet` na velmi úzké úrovni. `DataSet` si tvoří tabulky automaticky podle dat v XML dokumentu a práce s ním se tak velice zjednoduší. Problém nastal v okamžiku, kdy byla načtena tabulka elementů, které nebyly toho jména v celém dokumentu jediné. `DataSet` vzhledem k nedostupnosti XML schématu nerozeznal úroveň daného elementu v dokumentu a seskupil i elementy, které nebyli sourozenci. To bylo pro potřeby aplikace nepřípustné a načítání tabulek do `DataSetu` muselo být implementováno explicitně vlastními metodami.

4.10.2 Použití XSL transformací

Druhou změnou v původní myšlence implementace bylo upuštění od XSL transformace pro převod do HTML. Formátovací styl by musel vzhledem k obecnosti otevíraných dokumentů být vytvářen dynamicky. Vezmeme-li v úvahu požadovaný výsledek konverze, zjistíme, že XSL transformace by nepřinesla lepšího výsledku, než je dosaženo použitým způsobem, naopak by vyžadovala mnohem větší úsilí při vlastní implementaci.

4.10.3 Využití XML schématu

Poslední změnou základního návrhu bylo vynechání generování XSD schématu otevíraných dokumentů. Využití schématu by ulehčilo práci zejména při tvorbě zjednodušeného stromu reprezentujícího strukturu dokumentu. Bohužel jazyk C# v tomto ohledu neposkytuje nástroj funkční natolik, aby bylo výsledné schéma použitelné. Spojením

dokumentu načteného v `XmlDataDocumentu` s `DataSetem` totiž dojde k úpravě některých referenčních vazeb. Schéma, které s pomocí `DataSetu` vytvoříme, není správným schématem původního XML. Tento postup je určen pro předem známé a očekávané typy dokumentů, pro zpracování obecného XML se nehodí.

5 Závěr

Tato kapitola se zabývá zhodnocením výsledků práce, možnostmi dalšího vývoje v oblasti vizualizace XML a aplikace XML Dabster a vlastním zhodnocením celé práce.

5.1 Zhodnocení výsledků

Jako součást této práce, věnující se problematice vizualizace obecného XML dokumentu, byla vyvinuta aplikace XML Dabster. Tato aplikace byla vytvořena jako řešení některých problémů s vizualizací XML s důrazem na uživatelskou přívětivost a jednoduchost. Díky této aplikaci je schopen běžný uživatel otevřít jakýkoliv XML dokument, prohlédnout si jeho obsah, potřebné informace exportovat do jiného formátu nebo je vytisknout. To vše při zachování jednoduchosti a přehlednosti.

5.2 Možnosti dalšího vývoje

Aplikace XML Dabster si neklade za cíl stát se profesionálním či komerčním nástrojem, přesto je její vývoj do značné míry otevřený nové funkcionalitě a možnostem.

5.2.1 Editace

Hlavním nedostatkem aplikace je nemožnost otevřené dokumenty editovat. Aplikace je navržena pro vizualizaci a proto tato možnost v jejím základním vybavení chybí. Možnost editace by byla z uživatelského hlediska jistě vítána.

5.2.2 Vícenásobné otevírání tabulek

Zvýšení přehlednosti zobrazení XML dokumentů by prospěla možnost zobrazit více tabulek najednou. To by umožnilo uživateli lépe se orientovat v obsahu dokumentu, popřípadě porovnávat data z různých tabulek XML dokumentu. Implementačně by šla tato možnost realizovat prostým rozdělením tabulkového vizualizačního panelu na dvě horizontálně dělené části. Uživatel by pak zvolil, do které části chce vizualizovanou tabulku umístit.

5.2.3 Otevírání více dokumentů najednou

Další možností vylepšení funkcionality aplikace by mohla být možnost otevření více XML dokumentů najednou. Pro příklad si představme otevření dvou takových dokumentů. Podobně jako v příkladu z předchozí podkapitoly by byl zobrazovací prostor rozdělen na dva horizontální panely. Stejně tak by byl také rozdělen prostor pro stromovou reprezentaci dokumentu. Tím by vznikl dostatek prostoru pro zobrazení dvou různých dokumentů a jejich vzájemné porovnávání, sjednocování a podobné operace. Pro větší komplexnost by bylo možné tuto funkci realizovat klasickým otevíráním dokumentových oken aplikace.

5.2.4 Vyhledávání

Pro větší komfort při vizualizací obsáhlejších XML dokumentů by mohla sloužit funkce vyhledávání. Pro tuto funkci se také otevírá mnoho možností specifikace vyhledávání. Uživatel by mohl hledat buď elementy, nebo vlastní hodnoty, případně by mohl zvolit podstrom či oblast dokumentu, které by chtěl prohledat.

5.2.5 Rozšířené možnosti exportu do HTML

Export datové tabulky do webové stránky HTML nabízí mnoho možností pro další rozvoj. Uživateli by mohlo být nabídnuto vlastní nastavení stylu výsledné stránky od volby typu a velikosti písma, přes volbu barev až po definování stylu exportované tabulky. Implementačně by toto rozšíření mohlo být řešeno jako nové aplikační okno s nastavením veškerých podrobností exportu s možností přímého náhledu v některém z internetových prohlížečů (implicitně v primárním prohlížeči operačního systému).

5.2.6 Operace nad tabulkami

V rámci implementace editace dokumentu by mohlo být uživateli umožněno provádět operace s tabulkami typu sjednocení dvou tabulek, porovnání dvou tabulek, odstranění duplicitních dat atd. V tomto směru je aplikace XML Dabster otevřena dalšímu vývoji a jeho možnostem.

5.3 Zhodnocení práce

Tato bakalářská práce se zabývá problematikou vizualizace dokumentů XML. Toto téma je dosud dosti otevřené a ve světě softwaru nejsou všechny jeho aspekty zcela vyřešeny, díky tomu je řešení problematiky zobrazení obecných XML dokumentů dobrým námětem pro podobnou práci.

Výsledkem analýzy tématu práce je implementace programu XML Dabster, který je navržen jako uživatelsky přívětivý software pro zobrazení XML dokumentů. Jeho funkcionalita splňuje požadavky práce a poskytuje několik funkcí nad jejich rámec, jako je export či tisk datových tabulek.

Další vývoj vizualizace obecných XML dokumentů je nastíněn v kapitole 5.2 Možnosti dalšího vývoje. Implementace popsaných možností by posunula vizualizaci XML kódu o další stupeň a poskytla by uživateli mocný nástroj pro komplexní správu XML dokumentů.

Jak teoretický, tak praktický výzkum zadané problematiky byl také přínosný pro rozvoj obecných i praktických programovacích znalostí a technik. Na jedné straně to bylo studium standardů publikovaných konsorciem W3C [1], na straně druhé zdokonalování implementačních dovedností v jazyce C# a prostředí Microsoft .NET.

6 Literatura a odkazy

[1] *World Wide Web Consortium* [online]. 1994 , 2008/05/06 [cit. 2008-05-01]. Dostupný z WWW: <http://www.w3.org/>

[2] KOSEK, Jiří. *XML -- Staronový formát* [online]. 1999 [cit. 2008-05-01]. Dostupný z WWW: <http://www.kosek.cz/clanky/xml/xml-historie.html>

[3] *Extensible Markup Language : Wikipedie, otevřená encyklopedie* [online]. 2004 [cit. 2008-05-01]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/Extensible_Markup_Language>

[4] BŘÍZA, Petr. Základy jazyka XPath. *Interval.cz* [online]. 2004 [cit. 2008-05-01]. Dostupný z WWW: <http://interval.cz/clanky/zaklady-jazyka-xpath/>

[5] *Microsoft .NET Framework* [online]. 2007 [cit. 2008-05-01]. Dostupný z WWW: <http://www.microsoft.com/net/Information.aspx>

[6] *Namespaces in XML 1.0 (Second Edition)* [online]. 2006 , 16 August 2006 [cit. 2008-05-01]. Dostupný z WWW: <http://www.w3.org/TR/REC-xml-names/>

[7] CHERAGHI, Afrasiab. *Code Project : DataGridView Printing by Selecting Columns and Rows* [online]. 2006 , 11 Mar 2007 [cit. 2008-05-01]. Dostupný z WWW: <http://www.codeproject.com/KB/grid/PrintDataGridView.aspx>

[8] YOUNG, Kenny. *Code Project : C# - XML Schema Validator* [online]. 2004 , 12 Nov 2004 [cit. 2008-05-01]. Dostupný z WWW: <http://www.codeproject.com/KB/cpp/XmlSchemaValidator.aspx>.

[9] Esposito, D. *XML efektivní programování pro .NET.* Praha: Grada Publishing, 2004. 2034 s. ISBN 80-247-0775-6

[10] Quin, L. W3C. *Extensible Markup Language (XML)*. 2006. Dostupný z: http://www.w3.org/XML/

[11] Troelsen, A. *C# a .NET 2.0 profesionálně*. Brno: Zoner Press, 2006. 1197 s. ISBN 80-86815-42-0

[12] Jelinek, J. - Slavik, P. XML visualization using tree rewriting**.** *Proceedings of the 20th spring conference on Computer graphics*. ACM, 2004. pp. 65-72. ISBN:1-58113-967-5

[13] *C#: programujeme profesionálně* /Brno : Computer Press,2003. 1. vyd. xxx, 1130 s. : il. ISBN 80-251-0085-5

[14] Sells, Chris. *C# a WinForms: programování formulářů Windows* /Brno : Zoner Press,2005. Vyd. 1. 648 s. ISBN 80-86815-25-0

[15] SPERBERG-MCQUEEN , C. M., THOMPSON, Henry. *W3C XML Schema* [online]. 2000 , 2008/01/18 [cit. 2008-05-01]. Dostupný z WWW: <http://www.w3.org/XML/Schema>

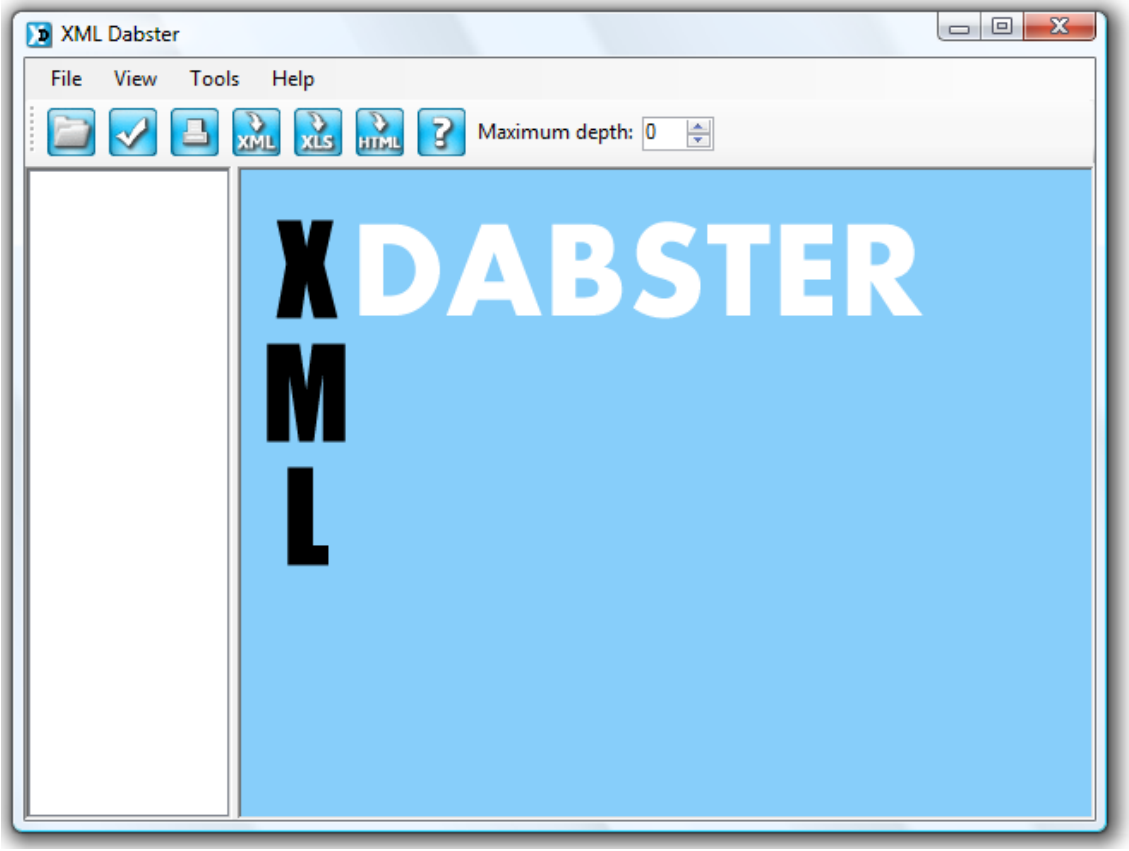
7 Přílohy

7.1 Uživatelská příručka programu XML Dabster

Tato kapitola slouží jako uživatelská příručka programu XML Dabster, který byl vytvořen jako součást práce na interaktivní vizualizaci XML. Návrh a popis implementace programu byly popsány výše. V této kapitole bude stručně popsáno ovládání a funkce programu.

7.1.1 Instalace a spuštění programu

Program XML Dabster se nemusí instalovat. Pro jeho provoz stačí soubor `XMLDabster.exe` a běhové prostředí Microsoft .NET 3.5. Program může běžet jak z lokálního disku, tak z mobilního zařízení jako je flash disk nebo CD. Pro otevření aplikace spusťte soubor `XMLDabster.exe`. Uživatelské rozhraní programu je v anglickém jazyce. Ten byl zvolen pro jeho univerzálnost ve světě počítačů. Po spuštění programu vidíme okno aplikace tak, jak jej zachycuje obrázek P1.



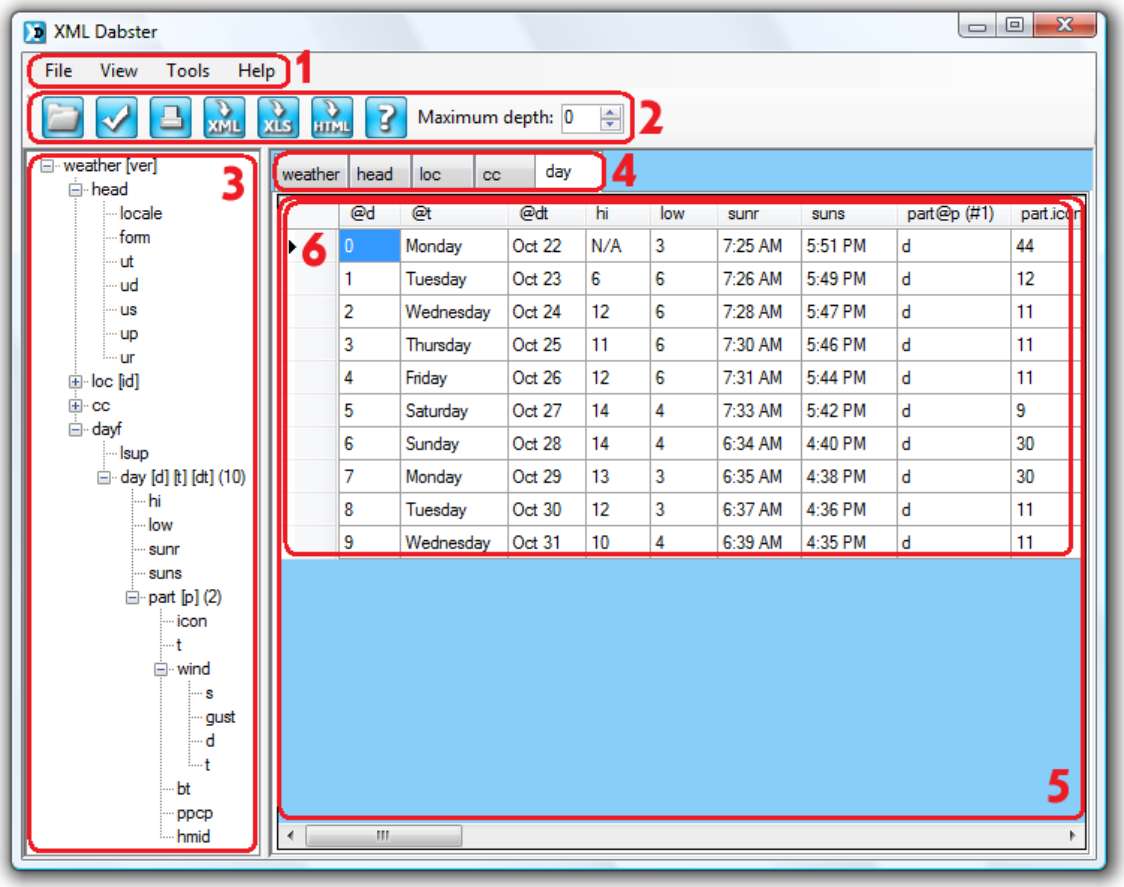
Obrázek P1 – aplikační okno programu XML Dabster

7.1.2 Otevření XML dokumentu

Otevřít XML dokument můžeme dvěma různými způsoby. První způsob je přes programové menu. Klikneme na položku File a následně na Open. Druhou, jednodušší možností je stisk tlačítka Open v panelu nástrojů. Toto tlačítko je označeno symbolem složky. V otevřeném dialogovém okně zvolíme XML dokument, který chceme otevřít, a stiskneme Open (případně otevřít, podle jazykové verze operačního systému). V levé části aplikačního okna se objeví kořen stromového zobrazení dokumentu. Nyní si můžeme rozevřít jednotlivé větve stromu a prohlédnout si strukturu dokumentu. Po poklikání na jakýkoli uzel stromu se zobrazí v pravé části aplikačního okna tabulka reprezentující data zvoleného elementu.

7.1.3 Hlavní ovládací a vizuální prvky programu

Nyní máme otevřen XML dokument a víme jak zobrazit tabulku některého elementu. Díky tomu si můžeme popsat ovládací prvky celé aplikace, jak je vidíme na obrázku P2.



Obrázek P2 - rozdělení ovládacích prvků aplikace

V rámečku 1 vidíme aplikační menu. To reprezentuje základní ovládací prvek, jak jej známe z jiných aplikací. Volbami aplikačního menu můžeme například otevřít dokument (jak jsme si popsali výše), konvertovat datovou tabulku nebo ukončit běh aplikace. Až na poslední zmíněnou funkci jsou všechny ostatní možnosti dostupné jiným způsobem, takže se nebudeme popisem aplikačního menu zabývat detailněji.

Rámeček označený číslem 2 obsahuje panel nástrojů. Ten je hlavním ovládacím prvkem celé aplikace. Funkce jednotlivých tlačítek jsou popsány v tabulce P1. Popis tlačítek v tabulce je řazen tak, jak jdou tlačítka zleva.

Tabulka P2 – popis tlačítek a funkcí panelu nástrojů

Ikona	Název	Funkce
	Open XML document Otevřít XML dokument	Otevření existujícího XML dokumentu ze zadaného umístění
	Validate XML by XSD file Zkontrolovat XML dokument podle XSD souboru	Ověření správnosti dokumentu podle schématu XSD
	Print table Vytisknout tabulku	Vytisknutí aktuálně zobrazené tabulky
	Export table to XML Exportovat tabulku do XML souboru	Export aktuálně zobrazené tabulky do samostatného XML souboru
	Export table to XLS Exportovat tabulku do XLS souboru	Konverze zobrazené tabulky do souboru s příponou XLS pro program MS Excel
	Export table to HTML Exportovat tabulku do stránky HTML	Konverze zobrazené tabulky na webovou stránku HTML
	About O programu	Zobrazí okno „O programu“
-	Maximum depth Maximální hloubka	Nastavení maximálního zanoření pro vytvoření nové tabulky

Další rámeček, na obrázku P2 označený číslem 3, je panel pro zobrazení zjednodušené stromové reprezentace dokumentu. Tento prvek slouží pro snadnou a přehlednou navigaci strukturou otevřeného XML dokumentu. Skrze tento prvek také otevíráme tabulky dat

elementů, které chceme vizualizovat. Jak bylo popsáno výše, tuto činnost realizujeme dvojklikem na zvolený element.

Číslem 4 je na obrázku P2 označen prostor kde se zobrazují záložky panelů se zobrazenými tabulkami. Název záložky je stejný, jako název zobrazeného elementu. Panelů se záložkami můžeme mít otevřeno libovolné množství. V případě, že chceme některý z panelu zavřít, provedeme to buď přes hlavní aplikační menu v podmenu View, nebo pomocí kontextového menu, které bude popsáno níže.

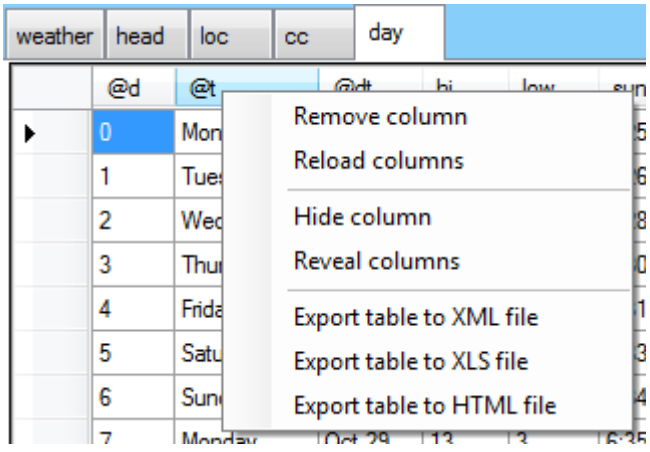
Číslo 5 zachycuje celý prostor panelu, jenž je rezervován pro datovou tabulku s obsahem elementu/ů. Tu vidíme na obrázku označenu rámečkem s číslem 6. Na obrázku vidíme otevřenou příkladnou tabulku s několika sloupci. Také je vidět, že tabulka se do vymezeného prostoru nevměstnala celá. Proto je na spodní straně panelu posuvná lišta, jenž slouží pro rolování obsahu tabulky. Pokud bude mít tabulka tolik řádků, že se do panelu nevejde i vertikálním směrem, bude zobrazena i svislá posuvná lišta.

7.1.4 Kontextová menu

V aplikaci jsou přítomna dvě kontextová menu. Ty vyvoláme stisknutím pravého tlačítka myši na určitém vizuálním prvku.

První kontextové menu vyvoláme pravým tlačítkem myši stisknutým nad záložkami otevřených panelů. Jedinou jeho funkcí je možnost „Close tab“, která uzavře panel patřící záložce, nad kterou jsme klikli.

Druhé kontextové menu můžeme vyvolat pravým kliknutím nad hlavičkami sloupců v otevřené tabulce. Všechny jeho volby vidíme na obrázku P3.



Obrázek P3 – kontextové menu nad hlavičkou sloupce tabulky

Funkce jednotlivých položek menu jsou popsány v tabulce P2. Funkce posledních tří položek jsou popsány již výše v tabulce P1, proto se jimi nebudeme zabývat.

Tabulka P2 – položky a funkce kontextového menu nad hlavičkou sloupce tabulky

Jméno položky	Funkce
Remove column Odstranění sloupce	Odstraní z tabulky sloupec, kterým se dále nechceme zabývat, ani ho případně nechceme exportovat.
Reload columns Znovunačtení sloupců	Znovu načte celou tabulku včetně dříve odstraněných sloupců.
Hide column Schování sloupce	Schová zvolený sloupec. Ten je tak odstraněn z vizuálního pohledu, při exportu se však exportuje také.
Reveal columns Zobrazení sloupců	Zobrazí všechny schované sloupce zpět do tabulky. Zobrazení se netýká odstraněných sloupců

7.1.5 Exportní funkce

Po spuštění některé z exportních funkcí, kterého docílíme některou z výše popsaných možností, je otevřeno dialogové okno pro ukládání souborů. V tom vybereme klasickým způsobem složku, kam chceme cílový soubor uložit a vyplníme kolonku pro jeho jméno. Následným stiskem tlačítka Save (případně Uložit, podle jazykové verze operačního systému) bude soubor uložen pod zvoleným jménem do vybrané cílové složky.

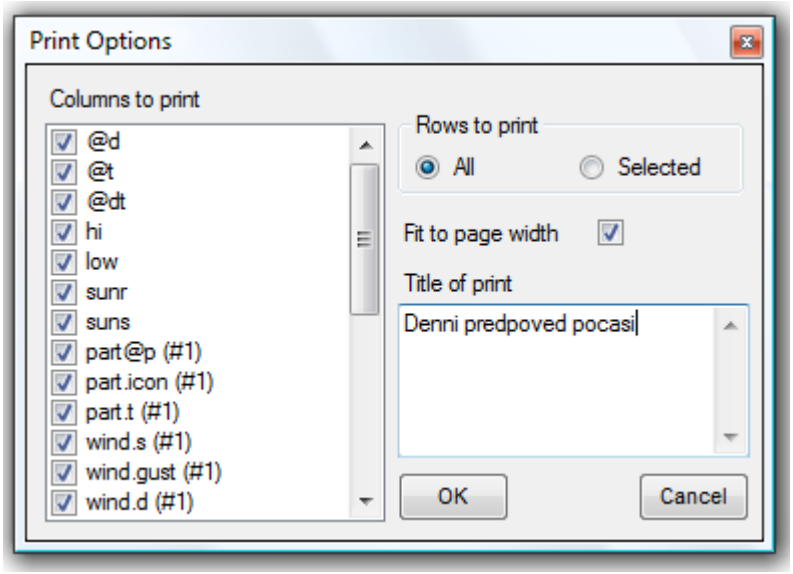
7.1.6 Kontrola validity dokumentu

Kontrolu správnosti XML dokumentu spustíme některou z možností popsaných výše. K provedení kontroly potřebujeme soubor se schématem XML dokumentu ve formátu XSD.

Po spuštění kontroly se nás aplikace dotáže klasickým dialogovým oknem pro výběr souborů na dokument, který chceme zkontrolovat. Ve druhém dialogovém okně pak vybereme soubor se schématem dokumentu. Po správném otevření obou souborů je zobrazen výsledek kontroly. V případě úspěchu je zobrazena informační zpráva. V případě neúspěchu je zobrazena chybová zpráva s odůvodněním selhání kontroly.

7.1.7 Tisk tabulky

Tisk tabulky vyvoláme buď tlačítkem v panelu nástrojů, nebo z aplikačního menu. Po spuštění tisku je zobrazen nastavovací dialog, kde zvolíme některá nastavení tisku. Zde můžeme definovat slupce tabulky, které chceme vytisknout, nebo název tiskové úlohy. Jak takový dialog vypadá, vidíme na obrázku P4.



Obrázek P4 – dialogové okno nastavení tisku

Po nastavení vlastností tisku zmáčkneme tlačítko OK. Před samotným tiskem je zobrazeno okno s náhledem na stránku/y připravenou k tisku. Tisknout začneme stisknutím tlačítka se symbolem tiskárny v levém horním rohu náhledového okna.

7.2 CD příloha

Na přiloženém CD nosiči jsou umístěny zdrojové kódy programu XML Dabster, použité obrázky a ikony a několik vzorových XML dokumentů používaných pro testovací účely.

Obsah a struktura obsahu CD jsou popsány v tabulce P3.

Tabulka P2 – struktura obsahu CD přílohy

Adresář	Obsah
/XMLDabster	Zkompilovaný, spustitelný program XML Dabster
/source	Zdrojové kódy aplikace XML Dabster
/images	Obrázky a ikony použité při tvorbě programu a taxtu této práce
/.NET	Balíček pro instalaci prostředí Microsoft .NET 3.5
/doc	XML soubor obsahující popis metod použitých v implmentaci
/XML	Příklady datově orientovaných XML dokumentů
/	Text této práce v souboru `interaktivniVizualizaceXML.pdf`