Zpravodaj Československého sdružení uživatelů TeXu

Zdeněk Wagner

XML versus TeX, výhody a nevýhody

Zpravodaj Československého sdružení uživatelů TeXu, Vol. 14 (2004), No. 3-4, 211–219

Persistent URL: http://dml.cz/dmlcz/149975

Odkazy

1. VueScan, `http://www.hamrick.com/`
2. Šmok J., Pecák J., Tausk P.: Barevná fotografie. Druhé, upravené vydání. SNTL, Praha 1978.
3. Brabec S.: Grafika v Linuxu (seriál), `http://root.cz`
4. SCARSE, `http://www.scarse.org/`
5. LibTIFF – TIFF Library and Utilities, `http://www.libtiff.org`
6. CinePaint, `http://cinepaint.sourceforge.net`

XML versus TEX, výhody a nevýhody

Zdeněk Wagner

Přednáška volně navazuje na přednášky o XML z minulého SLT. Srovnává možnosti obou systémů. Vysvětluje, co poskytuje zejména TEXistům XML. Zamýšlí se nad tím, kdy je vhodné přímé psaní textů v TEXu. Uvádí možnosti, jak z obou tupů zdrojových textů generovat soubory v jiných formátech a s jakými výsledky. Je též popsána možnost spojení TEXu i XML s databázemi.

Klíčová slova: XML, TEX, LATEX, XSLT

Motto

"Well, in *our* country," said Alice, still panting a little, "you'd generally get to somewhere else—if you ran very fast for a long time as we've been doing."

"A slow sort of country!" said the Queen. "Now, *here,* you see, it takes all the running *you* can do, to keep in the same place. If you want to get somewhere else, you must run at least twice as fast as that."

Lewis Carroll [1]

Úvod

Tištěná kniha se začíná šířit v polovině 15. století po vynálezu knihtisku. V průběhu staletí došlo k mnoha vynálezům a zlepšením, jež zlevnily a zkvalitnily knižní produkci. Ve své formě však až do osmdesátých let minulého století kniha zůstává stále knihou tištěnou na papíře.

V závěru 20. století dochází k bouřlivému rozvoji výpočetní techniky i počítačových sítí, jak lokálních, tak internetu. Roste tlak na urychlení výměny informací. Na světlo světa proto vstupují nové formy publikací, které již nejsou

spojeny s papírovým nosičem informací. Kniha v klasické podobě však ještě nevymírá. Naopak, často je požadováno, aby dokument byl publikován současně v podobě papírové i elektronické.

Stejně jako původní Gutenbergovy dřevěné typy byly vytlačeny typy olověnými a přibližně o pět století později ruční sazbu nahradila sazba počítačová, lze předpokládat, že staré softwarové nástroje vyklidí prostor nástrojům novým, nebo se alespoň přizpůsobí vznikajícím potřebám.

TEX patří nesporně k nejlepším sázecím programům. V souvislosti s rozvojem nových publikačních forem však vyplouvají jeho slabé stránky. Současně se objevují nové nástroje a datové formáty. Jedním z těch, jež si snaží získat prostor, je XML.

Tento příspěvek je zaměřen zejména na XML. Nebude se však vůbec zabývat sazbou dokumentů XML, protože této problematice se věnovaly články ve Zpravodajích Československého sdružení uživatelů TEXu 3–4/2002 a 1/2003. Nebudou zde ani probírány konkrétní nástroje. Článek je autorovým zamyšlením nad tím, co XML přináší zejména TEXistům. Jeho cílem je vzbudit v čtenářích zájem k jejich vlastnímu zamyšlení nad tím, jaké postupy budou v budoucnu používat při zpracování svých dokumentů.

Boření mýtů

Při srovnávání se často argumentuje tím, že na rozdíl od TEXu, jehož značkování je převážně prezentační, XML definuje logickou strukturu dokumentu. Vezměme si však jako příklad libovolný dokument XHTML. Kořenovým elementem dokumentu je `<html>`, v něm se vyskytuje hlavička `<head>` následovaná tělem dokumentu `<body>`. To jsou v zásadě jediné strukturní elementy XHTML verze 1. Zbytek značkování je výhradně prezentační.

Základní struktura LATEXových dokumentů je definována logickými značkami `\title`, `\chapter`, `\section` apod. Pak už záleží pouze na autorovi, zda ve zbytku dokumentu použije značkování logické, nebo prezentační. Pro plain TEX platí v zásadě totéž.

Přísně logické značkování popsal autor ve svém starším článku [2], kde se zabýval spojením databáze s LATEXem. Výstup z databáze měl následující formát:

```
\def\cislo{0001}\def\jmeno{Gorin}\def\inic{A. V.}
\def\tit{Prof.}\def\zeme{Russia}\def\zemecesky{RU}
\def\zemenvel{RUSKO}\def\tel{}\def\fax{}\def\email{}
\def\adri{...}\def\adrii{...}\def\adriii{...}\def\adriv{}\def\adrv{}...\exec
```

Definice makra `\exec` plnila stejný účel jako transformační styl XSLT, úlohu formátovacích objektů převzala makra definovaná v příslušném LATEXovém balíčku.

Při srovnávání tedy nelze používat zevšeobecněná tvrzení. Jak jsme si ukázali, volba mezi logickým a prezentačním značkováním není striktně vnucena tím, zda používáme XML či TEX, ale je do značné míry určena autorem dokumentu nebo autorem specifikace datového formátu, který je na jedné z výše uvedených technologií založen.

Formální správnost a úplnost dokumentu

Klikací sázecí programy obvykle nepovolí vytvoření neplatného dokumentu. Jiná je však situace v systémech, kde se do zdrojového textu zapisují logické či prezentační značky, ať už je to TEX, troff nebo XML. Zde máme možnost vytvořit nepřeberné množství syntaktických chyb, jež se neprojeví při psaní dokumentu, ale až při jeho zpracování. Pokud jsme ručně vytvořili jednoduchý dokument, celkem snadno z chybové zprávy pochopíme, co jsme provedli špatně. U složitějších dokumentů to již může být náročnější. Noční můrou pak bývá hledání chybějící pravé závorky v několikasetstránkové knize, když některé makro bylo definováno jako `\long`. Není-li kniha rozumně členěna do malých, samostatně načítaných souborů, TEXová diagnostika nám příliš nepomůže.

Soubor však může být formálně chybný, přestože LATEX při zpracování žádnou chybu nehlásí. Typickým příkladem je vynechání makra `\chapter` v třídě REPORT. Autor se pak diví, proč má `\section` číslo 0.1.

Kontrola správnosti a úplnosti je velmi důležitá i v případě, kdy si soubor pro další zpracování generujeme jako výstup z databáze. Pokud při zpracování dojde k chybě, potřebujeme jednoznačně určit, zda jsme si vygenerovali chybný soubor, nebo zda je chyba v makrech pro tisk. Můžeme si samozřejmě naprogramovat vlastní validátor v TEXu, ale XML má pro tyto účely hotové nástroje.

Předvedeme si to na konkrétním příkladu, kdy jako validační schéma využijeme Relax NG. Uvedeme jen část souboru.

Data exportovaná z databáze mají obsahovat údaje o přednáškách a vývěskách na jisté konferenci. Program konference zahrnuje sekci plenárních přednášek, několik sekcí přednášek a několik sekcí vývěsek. Část specifikace, uložená v souboru `export-base.rng`, vypadá takto:

```
<?xml version='1.0'?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:a="http://hroch486.icpf.cas.cz/rng-comments"
         datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">

<start>
  <ref name='program.content'/>
</start>

<define name='program.content'>
  <interleave>
    <zeroOrMore>
```

```
      <ref name='def.plenary.lectures'/>
    </zeroOrMore>
    <zeroOrMore>
      <ref name='def.lectures'/>
    </zeroOrMore>
    <zeroOrMore>
      <ref name='def.posters'/>
    </zeroOrMore>
  </interleave>
</define>

<define name='def.time.info'>
  <a:comment>
    Define beginning time of a lecture as a combination of hour and minute.
    These attributes are not required in the preliminary program and
    must be ignored if present.
    Information comes from database, thus it is not necessary to validate
    the values.
  </a:comment>
  <attribute name='hour'>
    <data type='nonNegativeInteger'>
      <param name='minInclusive'>8</param>
      <param name='maxInclusive'>18</param>
    </data>
  </attribute>
  <attribute name='minute'>
    <data type='nonNegativeInteger'>
      <param name='minInclusive'>59</param>
    </data>
  </attribute>
</define>

<define name='choose.time.info'>
  <a:comment>
    To be redefined in the parent grammar, see def.time.info
  </a:comment>
  <notAllowed />
</define>

<define name='def.lecture.attrib'>
  <a:comment>
    Attributes for lectures and plenary lectures.
  </a:comment>
  <ref name='def.serial.num'/>
  <ref name='choose.time.info'/>
</define>

<define name='def.plenary.lectures'>
  <a:comment>
    Section with plenary lectures
```

```
  </a:comment>
  <element name='plenary-lectures'>
    <ref name='def.halfDayId'/>
    <oneOrMore>
      <a:comment>
        The section contains one or more plenary lectures.
      </a:comment>
      <element name='plenary-lecture'>
        <ref name='def.lecture.attrib'/>
      </element>
    </oneOrMore>
  </element>
</define>

...
</grammar>
```

Ze specifikace byly úmyslně pro stručnost vyhozeny definice mnoha atributů a vše si ukazujeme jenom na plenárních přednáškách. Důležitá je v této ukázce definice vzoru `choose.time.info`. Je v ní specifikováno, že použití této konstrukce není povoleno. Tento vzor se však používá v `def.lecture.attrib`. Funguje to jen proto. že validaci provádíme pomocí souboru `export.rng`, v němž je výše zmíněný soubor načten a vzor `choose.time.info` je předefinován:

```
<?xml version='1.0'?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:a="http://hroch486.icpf.cas.cz/rng-comments"
         datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">

<start>
  <element name='chisa-program' xmlns="http://relaxng.org/ns/structure/1.0">
    <a:comment>
      preliminary - for preliminary program, no numbers, no time information
      final - for final program, time information required
    </a:comment>
    <choice>
      <group>
        <attribute name='type'>
          <value>preliminary</value>
        </attribute>
        <grammar>
          <include href='export-base.rng'>
            <define name='choose.time.info'>
              <a:comment>Time info is optional.</a:comment>
              <optional>
                <ref name='def.time.info'/>
              </optional>
            </define>
          </include>
        </grammar>
```

```
      </group>
      <group>
        <attribute name='type'>
          <value>final</value>
        </attribute>
        <grammar>
          <include href='export-base.rng'>
            <define name='choose.time.info'>
              <a:comment>Time info is required.</a:comment>
              <ref name='def.time.info'/>
            </define>
          </include>
        </grammar>
      </group>
    </choice>
  </element>
</start>

</grammar>
```

Všimněte si, že vzor `choose.time.info` obsahuje pouze odkaz na vzor `def.time.info`. Rozdíl je v tom, že v předběžném programu jsou časové informace nepovinné. Proto je příslušný vzor uzavřen v elementu `<optional>`. Podařilo se nám tedy mírně komplikovaná pravidla zapsat poměrně snadno. Kdybychom totéž chtěli řešit pouze TEXovými nástroji, vyžadovalo by to jistou dávku programátorského úsilí.

TEXisté mohou namítnout, že by to přece jen šlo naprogramovat snadno. V tomto konkrétním případě však byla situace komplikovaná tím, že soubor, vytvořený jedním člověkem, byl zpracováván jiným člověkem. Validační schéma tedy sloužilo mimo jiné i k tomu, aby v případě potřeby rozhodlo, kdo udělal chybu. Pokud by vše bylo řešeno výhradně na úrovni TEXu, zbývala by stále možnost, že chybu udělal programátor validačního TEXového makra.

Hledání specifických informací

Chceme-li najít určitou informaci, nemusí být fulltextové vyhledávání optimální metodou. Občas nás zajímá slovo či slovní spojení vyskytující se ve struktuře určitého typu. Je tedy nutné, aby zdrojový soubor měl dobré logické značkování. Kromě toho potřebujeme odpovídající prohlédávací nástroje. Takové možnosti nám TEX nenabízí, museli bychom si je naprogramovat. Ve světě XML však takové nástroje máme.

Podívejme se na knihu XML pro každého [3]. Autor chtěl z textu vytáhnout seznam odkazů pro zveřejnění na webu [4]. Dosáhl toho jednoduchým transformačním stylem [5]. Podívejme se na jednu šablonu:

```
<xsl:template match="chapter/section/title">
   <xsl:if test="..//ulink">
      <h3>
         <xsl:value-of select="."/>
      </h3>
   </xsl:if>
</xsl:template>
```

Šablona zpracovává elementy `<title>`, jejichž rodičem je `<section>` mající za rodiče element `<chapter>`. Nadpis se na webové stránce zobrazí jen tehdy, jestliže rodič elementu `<title>` má potomka `<ulink>`. Pokud bychom stejnou úlohu řešili v TEXu, bylo by příslušné makro jistě delší.

Výstup do více formátů

XML nabízí mocný trransformační nástroj, kterým lze generovat výstupy v několika formátech. Implementace formátovacích objektů pro tisk již tak dobrá není. Zde se nabízí tisk prostřednictvím TEXu.

Generování jiných typů výstupů z TEXových souborů je problematičtější. Jedna metoda byla zveřejněna v již zmíněném článku [2], jinou metodu popsal Petr Olšák. V obou případech se však jednalo o převod souboru poměrně pevně definovaného formátu.

Existují nástroje pro konverzi LATEXových souborů do HTML. Nesmíme však zapomenout na známou skutečnost: „Only TEX can read TEX.“ Nemůžeme počítat s tím, že si takové nástroje poradí se všemi konstrukcemi. Čím více se odchýlíme od standardních LATEXových šablon, tím větší je pravděpodobnost, že konverze selže.

Jistou možnost představuje též metoda, o níž se zmínil Sebastian Rahtz [7]. Sice prováděl konverzi z LATEXu do SGML, ale úprava pro XML by měla být snadná. Pokud však původní dokument nebyl dobře logicky strukturován, získáme logické značkování v XML jedině ruční editací.

Vraťme se ještě k validačnímu schématu z kapitoly ?? uvedenému na straně 213. Validovali jsme jím soubor, z něhož měla být vytištěna brožura. Ta měla být následně převedena do PDF pro účely zveřejnění na CD. Současně měly být vytvořeny WWW stránky. Ve všech verzích měl být autorský rejstřík s hyertextovými odkazy. Kromě toho se ze souboru měly tisknout potvrzovací dopisy autorům a vybrané informace měly být načteny do databáze pro další zpracování. Na těchto úkolech spolupracovalo několik lidí, a to na různých operačních systémech. V takové situaci je XML ideálním nástrojem, protože vše se dělá vlastně jen pomocí transformačních stylů.

Nevýhody XML

Nasazení XML není vždy spojeno s optimistickými náladami, jak jsme si to až dosud líčili. Prvním problémem jsou matematické rovnice. Jejich ruční zápis přímo v MathML je s výjimkou triviálních vzorečků úlohou pro masochisty. Nemáme-li konverzní program pro převod z lidské podoby do MathML, je lepší zapsat alternativní text v TEXové notaci a vygenerovat z něj obrázek.

Na druhou nevýhodu narazíme v okamžiku, kdy se snažíme soubor XML vytisknout. Pokud nám záleží na kvalitě, měli bychom jít z XML přes TEX. Přibyl nám tedy jeden krok. Není-li krok navíc kompenzován ziskem lepší struktury nebo jiných výhod, jedná se pouze o časovou ztrátu. Pak je přímé nasazení TEXu rozhodně přirozenějším řešením.

TEX je i pro mírně zkušeného uživatele mocným nástrojem, který lze snadno využít i v osobních tiskovinách a písemnostech. Dobře se hodí zejména v případech, kdy grafický vzhled je výrazně důležitější než logická struktura dokumentu. Nedovedu si představit, že bych DocBook používal na psaní milostné korespondence, sazbu reklam, vizitek či akcidenčních tiskovin, jako jsou třeba svatební oznámení.

Závěr

Co říci závěrem? Na jedné straně jsou situace, kdy XML představuje ideální nástroj, na druhé straně však máme případy, kdy nám XML pouze přidělává práci a přímé využití TEXu či LATEXu je nespornou výhodou. Mezi oběma krajnostmi se prostírá široké pole, kde nelze jednoznačně určit, která z metod je lepší. Vždy bude do značné míry záležet na vkusu, znalostech a dovednostech uživatele, který má konkrétní datový soubor zadaným způsobem zpracovat.

Odkazy

1. Carroll L.: Through the Looking-Glass and what Alice Found There. Chapter II: The Garden of Live Flowers. In: The Annotated Alice. Penguin Books, Harmondsworth, Middlesex, England 1970.
2. Wagner Z.: Spolupráce databáze s LATEXem. Zpravodaj Československého sdružení uživatelů TEXu, **10**(1–3), 49–78 (2000).
3. Kosek J: XML pro každého. Grada Publishing 2000. ISBN 80-7169-860-1. `http://www.kosek.cz/xml/`.
4. `http://www.kosek.cz/xml/odkazy.html`
5. `http://www.kosek.cz/xml/odkazy.xsl`

6. Olšák P.: Jak používám TeX? 3. Makro pro layout požadovaný Verlag Dashofer,
http://www.olsak.net/texpraxe.html; též diskusní příspěvek na
http://groups.google.com/groups?hl=cs&lr=&ie=UTF-8&selm=
200403080914.i289ECsT004145%40relay.felk.cvut.cz
7. Rahtz S.: The inaugural meeting of TUG India. Zpravodaj Československého sdružení uživatelů TEXu, **7**(4), 173–177 (1974).