Web ČKR: návrh a realizace (2)

Šárka Ocelková, ÚVT MU

Předchozí díl byl věnován počátkům tvorby webu ČKR, především stanovením jednotlivých úkolů a jejich rozdělením mezi jednotlivé zúčastněné strany. Dále byly nastíněny různé možnosti tvorby www stránek, od „ručního“ psaní přes programové řešení až po on-line či off-line generování. Poslední, co se v předchozím díle řešilo, byla problematika způsobu uložení dat. Zmíněny byly možnosti strukturovaný a nestrukturovaný soubor nebo databáze. V ukázkách byla na dvou typických příkladech – Adresář a Usnesení – naznačena možná struktura dat, jak ve strukturovaném souboru, tak v databázi.

Tento díl bude věnován další činnosti, která je zapotřebí při tvorbě webu – vygenerování www stránek z uložených dat. Dále budou zmíněny vnitřní mechanismy, které zajišťují vlastní funkčnost webu, a nakonec pak nástroj, který byl poskytnut laickému zákazníkovi, k umožnění aktualizace a přegenerování celého webu.

1 Generování www stránek z uložených dat

K vygenerování www stránky, ať je již uložena ve strukturovaném souboru nebo v databázi, je vždy zapotřebí vytvořit program, který v podstatě udává předpis, jak se mají strukturovaná data převést do výsledné HTML podoby. Na známých příkladech Adresáře a Usnesení si ukážeme, jak by vypadal takový program pro transformaci dat z relační databáze a z XML souboru. Vycházet budeme z databázové a XML-struktury dat Adresáře a Usnesení v podobě, v jaké byly navrženy a ukázány v předchozí části našeho článku; obdobně požadovaný výsledný HTML tvar bude v podobě ukázané minule.

1.1 Generování z relační databáze

Vhodným a nejčastěji používaným nástrojem pro prezentování dat z databáze a jejich převedením do HTML podoby je skript (.asp, .php, .jsp, ...). Ať je již zvolen kterýkoliv z programovacích jazyků, princip skriptu je prakticky vždy podobný: nejprve se dotáže databáze na potřebná data a po té je záznam po záznamu dle algoritmu zpracovává. V následujících ukázkách jsou orientačně načrtnuty ukázky programu pro Adresář a Usnesení, na atributy z databázového dotazu je v ukázkách odkazováno pomocí znaku „$“, na proměnné pomocí znaku „@“.

Adresář. Následuje obecná ukázka programu pro vygenerování adresáře, který převede data z databázové struktury do požadovaného výsledného HTML tvaru.

```
-- Dotaz do databáze
SELECT č.jméno, č.příjmení,
  č.titul_před_jménem,
  č.titul_za_jménem,
  č.funkce_ve_škole,
  š.název, š.url_www_stránek,
  š.adresa_ulice, š.adresa_číslo,
  š.adresa_PSČ, š.adresa_město,
  f.název, č.telefon, č.fax, č.e-mail
FROM člen AS č
LEFT OUTER JOIN
  škola AS š ON č.id_školy=š.id
LEFT OUTER JOIN
  funkce AS f ON č.id_funkce_v_ČKR=f.id
-- Zpracování dotazu
PROMĚNNÁ @pořadí
@pořadí=0
<TABLE>
OPAKUJ pro každý řádek výsledku
{ @pořadí=@pořadí+1
  <TR><TD>@pořadí</TD>
      <TD>$č.titul_před_jménem
          $č.jméno $č.příjmení,
    PODMÍNKA
      (existuje-li $č.titul_za_jménem)
      { $č.titul_za_jménem, }
    $č.funkce_ve_škole <BR>
    <A HREF="$š.url_www_stránek">
       $š.název</A> <BR>
    PODMÍNKA (existuje-li $f.název)
      { $f.název <BR> }
    $š.adresa_ulice $š.adresa_číslo <BR>
    $š.adresa_PSČ $š.adresa_město
      </TD>
  <TD><TABLE>
      PODMÍNKA (existuje-li $č.telefon)
        { <TR><TD>tel.:</TD>
              <TD>$č.telefon</TD></TR> }
      PODMÍNKA (existuje-li $č.fax)
        { <TR><TD>fax:</TD>
              <TD>$č.fax </TD></TR> }
      PODMÍNKA (existuje-li $č.e-mail)
        {<TR><TD>e-mail:</TD>
         <TD><A HREF="mailto:$č.e-mail">
             $č.e-mail</A></TD></TR> }
```

```
      </TABLE></TD>
  </TR> }
</TABLE>
```

Usnesení. Obdobné ukázky následují pro příklad usnesení, první pro výpis přehledu usnesení, druhá pak pro vlastní text jednoho konkrétního usnesení.

```
-- Dotaz do databáze
SELECT u.id, u.místo_konání,
  u.datum_konání
FROM usnesení AS u
-- Zpracování dotazu
<TABLE>
OPAKUJ pro každý řádek výsledku
{ <TR><TD>$u.id. zasedání ČKR</TD>
      <TD>($u.místo_konání,
           $u.datum_konání)</TD>
  </TR> }
</TABLE>
```

Aby nebyla ukázka na formátování a výpis jednoho usnesení příliš dlouhá na úkor přehlednosti a názornosti, je zde podstatně zjednodušeno formátování úrovní textu. Pouze je testováno, zda se úroveň zvyšuje (pak přibývají počáteční značky seznamu <UL>) nebo zmenšuje (pak přibývají koncové značky seznamu </UL>). Ve skutečnosti by mělo být testováno nejen zda má být příslušná část s odrážkami (<UL>) nebo číslováním (<OL>), ale především kolik úrovní přibylo, resp. ubylo. Případné vylepšení algoritmu již ponecháme na čtenáři.

```
-- Dotaz do databáze
SELECT u.id, u.místo_konání,
  u.datum_konání, uč.úroveň,
  uč.typ_číslování, uč.text_části
FROM usnesení AS u, usnesení_část AS uč
WHERE u.id=uč.id_usnesení
  AND u.id=id konkrétního usnesení
ORDER BY uč.pořadí_části
-- Zpracování dotazu
Usnesení $u.id. zasedání ČKR
$u.místo_konání, u.datum_konání
PROMĚNNÁ @předchozí_úroveň
@předchozí_úroveň=0
OPAKUJ pro každý řádek výsledku
{ PODMÍNKA
    (@předchozí_úroveň<$uč.úroveň)
  { <UL> }  /* zvětšuje se úroveň */
  PODMÍNKA
    (@předchozí_úroveň>$uč.úroveň)
  { </UL> } /* zmenšuje se úroveň */
  PODMÍNKA ($uč.úroveň>0)
  { <LI TYPE="$uč.typ_číslování"> }
      /* pouze je-li nenulová */
  $uč.text_části
  @předchozí_úroveň=$uč.úroveň
}
```

1.2 Generovaní z XML souboru

Pro vygenerování www stránky z dat uložených ve strukturovaném XML souboru je, stejně jako v předchozím případě, zapotřebí vytvořit program. Může být prakticky v libovolném programovacím jazyce, nejlépe za přispění specializované knihovny pro práci s XML. S výhodou lze využít skutečnost, že HTML je v podstatě také XML, a tudíž se nabízí použití XSL transformace.

Adresář. Následuje ukázka XSL transformace pro vygenerování adresáře z dat uložených v XML souboru.

```
<xsl:variable name="pořadí" select="0"/>
<TABLE>
<xsl:apply-templates select="ČLEN"/>
</TABLE>
<xsl:template match="ČLEN">
  <saxon:assign name="pořadí"
     select="$pořadí+1"/>
  <TR>
  <TD><xsl:value-of select="@pořadí"/>
      </TD>
  <TD><xsl:value-of
    select="@titul_před_jménem"/>
<xsl:value-of select="@jméno"/>
<xsl:value-of select="@příjmení"/>,
<xsl:if test="@titul_za_jménem">
     <xsl:value-of
        select="@titul_za_jménem"/>,
</xsl:if>
<xsl:value-of
   select="@funkce_ve_škole"/>
<BR/>
<A HREF="{/ŠKOLA[@ID=current()/
       @id_školy]/@url_www_stránek}">
  <xsl:value-of
     select="/ŠKOLA[@ID=current()/
             @id_školy]/@název"/></A>
<BR/>
<xsl:if test="@id_funkce_v_ČKR ">
  <xsl:value-of
     select="/FUNKCE[@ID=current()/
            @id_fce_v_ČKR]/@Název"/>
<BR/>
</xsl:if>
<xsl:value-of
    select="/ŠKOLA[@ID=current()/
```

```
        @id_školy]/@adresa_ulice"/>
  <xsl:value-of
      select="/ŠKOLA[@ID=current()/
         @id_školy]/@adresa_číslo"/>
  <BR/>
  <xsl:value-of
     select="/ŠKOLA[@ID=current()/
          @id_školy]/@adresa_PSČ"/>,
  <xsl:value-of
     select="/ŠKOLA[@ID=current()/
        @id_školy]/@adresa_město"/>
    </TD>
    <TD><TABLE>
        <xsl:apply-templates
            select="TELEFON/FAX/E-MAIL"/>
        </TABLE>
    </TD>
    </TR>
  </xsl:template>
  <xsl:template
    match="TELEFON/FAX/E-MAIL">
    <TR><TD><xsl:if test="position()=1">
          tel./fax/e-mail: </xsl:if></TD>
        <TD><xsl:value-of select="."/>
        <BR/>
    </TD></TR>
  </xsl:template>
```

Usnesení. Následuje obdobná ukázka pro přehled usnesení.

```
Přehled usnesení:
<TABLE BORDER="0"
  CELLSPACING="0"
  CELLPADDING="3">
  <xsl:apply-templates
       select="USNESENI">
    <xsl:sort select="@pořadí"
         order="descending"/>
  </xsl:apply-templates>
</TABLE>
<xsl:template match="USNESENI">
  <TR>
    <TD><xsl:value-of
        select="@pořadí"/>.
        zasedání ČKR</TD>
    <TD>(<xsl:value-of
            select="@místo_konání"/>,
         <xsl:value-of
            select="@datum_konání"/>)
    </TD>
  </TR>
```

Navržená XML struktura pro vlastní texty usnesení je jiná než v databázi, mnohem vhodnější a pohodlnější pro evidenci strukturovaných textů. XSL transformace je ale o to delší a komplikovanější, proto ji zde již nebudeme uvádět. Opět ponecháme na případné vyzkoušení čtenáři.

1.3 XML – proč bylo vybráno?

Aby mohl zákazník svá data libovolně modifikovat, aniž by musel stále žádat tvůrce stránek, je třeba zřídit mu k nim přístup. Zpřístupnění relační databáze je komplikovanější z hlediska nároků na klientský počítač, neboť je potřeba určitý software pro přístup do databáze. Pro zpřístupnění XML souborů s daty postačuje pouze vytvoření účtu zákazníkovi na www serveru a zpřístupnění části disku, kde jsou uložena jeho data. K editaci XML souboru postačuje obyčejný textový editor.

Pro variantu XML také hovoří možnost jeho strukturování, jak ukazují dříve uvedené příklady. Nevýhodou naopak je nutnost zaškolit zákazníka do formátu zápisu dat. Může se zdát, že v případě použití relační databáze by tento problém odpadl, neboť zápis dat je vlastně jen pouhé vyplňování tabulek. Složitě strukturované texty, kterých je v ČKR hodně (např. zápisy ze zasedání, výroční zprávy apod.), by se ale bez speciálně navržených podpůrných formulářů vkládaly do databáze jen velmi obtížně.

Po celkovém zhodnocení tedy nakonec zvítězila varianta XML díky možnosti strukturování dat a minimálním softwarovým nárokům.

2 Vnitřní architektura

2.1 Vygenerování www stránek

Nyní je již k dispozici sada XML souborů, v nichž jsou uložena potřebná data, a ke každému typu HTML stránky je již naprogramována XSL transformace (šablona), která udává předpis, jak se mají konkrétní data v XML převést do HTML podoby. Z toho pak vyplynul další úkol – zajistit automatické vygenerování celé www prezentace. Pro vytvoření každé html stránky je nutné vědět, z jakých dat (xml souboru) má vzniknout, kterou šablonou mají být data transformována a kam se má výsledek transformace uložit (tj. jméno a umístění výsledného souboru, který již bude součástí www prezentace). Z těchto důvodů vznikl *konfigurační soubor* (rovněž ve formátu XML), kde jsou tyto informace popsány. Aby se

při uvádění souborů nemusela neustále opakovat celá cesta k souborům, jsou v konfiguračním souboru navíc dodány atributy xml-root-dir, xsl-root-dir a html-root-dir, v nichž je uložena společná část cesty (jednotlivě pro xml zdroje, xsl transformace a html cíle) od kořene www serveru. Následuje ukázka konfiguračního souboru.

```
<FILES
  xml-root-dir="XML zdroje"
  xsl-root-dir="XSL transformaceme"
  html-root-dir="www serveru crc.muni.cz">

<ITEM xsl="adresar.xsl" xml="adresar.xml"
      html="directory/index.html"/>
... ... ...
<ITEM xsl="usneseni_index.xsl"
      xml="usneseni.xml"
      html="resolution/index.html"/>
<ITEM xsl="usneseni.xsl"
      xml="usneseni.xml"
      html="resolutions/65.html"/>
<ITEM xsl="usneseni.xsl"
      xml="usneseni.xml"
      html="resolutions/66.html"/>
... ... ...
<ITEM xsl="aktuality.xsl"
      xml="aktuality.xml"
      html="news/index.html"/>
<ITEM xsl="kalendar.xsl"
      xml="kalendar.xml"
      html="calendar/index.html"/>
... ... ...
<FILES>
```

Toto řešení má zatím ještě jeden nedostatek, který není nijak závažný a navíc je řešitelný (lze doprogramovat). Slouží-li jeden XML soubor jako zdroj dat pro několik html souborů (viz příklad usnesení, kde jsou všechna data o usneseních v jediném souboru a následně se z nich generují samostatné www stránky jednotlivých usnesení), musí být v konfiguračním souboru uveden tolikrát, kolik z něj má být vygenerováno html souborů. Tento nedostatek je však pozitivně využit, neboť cílové jméno html souboru slouží jako parametr pro xsl transformaci (např. vygeneruje se stránka pro jedno konkrétní usnesení).

Pro zpracování XSL transformací dle konfiguračního souboru byly použity knihovny SAXON (provádění transformací – viz [3]) a XALAN (parsování XML – viz [4]).

Při požadavku na vygenerování stránek se vždy vygeneruje celá sada stránek. Není zde řešeno částečné přegenerování (zvolených) stránek pro případ, kdy je potřeba uplatnit změnu jen na jedné či několika málo stránkách. Zatím je však toto řešení dostatečné, neboť vzhledem k rozsahu webu trvá přegenerování celé prezentace jen několik desítek sekund.

2.2 Jednotný vzhled stránek a navigační logika

Kapitola 2 v minulém díle pojednávala o různých možnostech tvorby stránek, oddělujících vlastní data od formátujícího HTML kódu stránky, výhodou pak je jeho uložení centrálně na jediném místě. Zbývá proto objasnit, jak je v případě ČKR řešen jednotný vzhled stránek.

V sadě XSL transformací je jedna speciální obecná transformace, která se využívá při generování všech stránek. Jedná se o soubor obecných pravidel a funkcí (např. pro výpis jednotného záhlaví a zápatí www stránky, nadpisů různých úrovní, odkazu „o úroveň výš“ ve stránkové struktuře apod.) zpřístupněný všem ostatním transformacím, které tak mohou tato pravidla a funkce libovolně využívat nebo se na ně odkazovat.

Jednotné grafické provedení je sice pro uživatele většinou přitažlivější, ale samotná grafika nestačí k tomu, aby se uživatel v celé stránkové struktuře dobře a správně orientoval, proto je vhodné také pouvažovat o rozvržení informací na stránkách a navigační logice.

Několikaletá zkušenost s vytvářením webů na ÚVT MU v Brně potvrdila, že na celý web lze pohlížet jako na kolekci www stránek, jež dohromady sice tvoří obecný graf, ale většinou vždy lze nalézt hlavní kostru, která vytváří stromovou strukturu. Tuto strukturu je možné procházet dvěma směry: vertikálně („shora dolů“) a horizontálně („vodorovně“). Vertikálním procházením se postupně zpřesňuje informace, horizontálním je pak možné získat tutéž informaci jen s jinými vstupními parametry nebo informaci na stejné úrovni, tj. ve stejném vztahu k nadřazené. Horizontální procházení webu je zpravidla reprezentováno ve formě svislé nabídky v levé části

stránky, k vertikálnímu procházení dochází postupně volbou odkazů v hlavní části stránky. Aby byl umožněn návrat zpět (tj. procházení „zdola nahoru“), objevují se s každou další úrovní postupně v pravém horním rohu stránky ikony, odkazující na předchozí nadřazené úrovně (viz také [1]).

2.3 Vývoj webu a transakční přegenerování

Jelikož www prezentace ČKR neustále „žije“ (dle potřeby se obměňují stránky aktualit nebo kalendáře, v současné době se připravuje historie atd.), je potřeba mít možnost podívat se, jak bude po provedení změn vypadat výsledná www stránka. Je také nutné počítat s případným překlepem či jinou chybou ve zdrojových datech nebo vyvíjených transformacích, což může zapříčinit buď nemožnost uplatnění transformace a tím zhavarování programu pro generování, nebo zveřejnění překlepu ve výsledném souboru prezentace. Obojí je samozřejmě nežádoucí a u veřejné www prezentace nelze takto riskovat. Proto byla zvolena (jak se tvůrcům již osvědčilo v jiných projektech) varianta dvou www serverů (nejsou nutné dva fyzické počítače), z nichž jeden je vývojový a druhý ostrý. Oba mají identickou adresářovou strukturu, tj. všechny soubory s daty, transformacemi i vygenerovaná prezentace existují dvakrát na dvou oddělených místech. Na vývojovém serveru je možné soubory s xml daty a xsl transformacemi modifikovat a generovat z nich pokusně výsledné www stránky, na druhém, ostrém serveru, nejsou určeny k modifikování, ale pouze k nahrazovaní z otestované varianty vývojové části.

Přegenerování www prezentace má ještě jednu „pojistku“ proti vzniku chyb (nelze jí ošetřit vlastní překlepy v textu, ale koncepční chybu v XML nebo XSL). Ze zkušeností s vývojem a provozem oficiální www prezentace MU v Brně byla do programu, který generuje www stránky, implementována vlastnost transakčního přegenerování, které zajistí, že v případě chyby v tomto procesu zůstává zveřejněna stará verze stránek. Toto řešení je založeno na skutečnosti, že se celá www prezentace nachází na UNIX serveru a je tak možné využít vlastnosti symbolických linků. O této problematice podrobněji pojednává [1].

3 Rozhraní pro zákazníka

Zákazník byl úspěšně zaškolen do formátu zápisu dat v XML a do logiky přegenerovávání www stránek, kterou pečlivě dodržuje. Pro editaci dat mu byl doporučen jednoduchý textový editor se zvýrazňováním syntaxe, spuštěním jednoho příkazu pak může přegenerovat vývojový web. S přenosem na ostrý web to již tak snadné není, neboť na vývojovém webu mohou být některé soubory rozpracovány. Z tohoto důvodu není možné nabídnout zákazníkovi jediný příkaz, který by najednou překopíroval všechny datové zdroje na ostrý web a přegeneroval www stránky. Proto byla vytvořena a zákazníkovi předložena sada vhodně nazvaných příkazů, které překopírují na ostrý web vždy jen ty datové zdroje, které se týkají určité samostatné části www prezentace, a následně přegenerují všechny www stránky na ostrém webu.

Závěr

Všechny www prezentace, které tvůrci až doposud vytvořili, vždy vytvářeli s vědomím, že s jejich „zákulisím“ budou operovat jen odborníci v oboru. Nebylo tedy třeba zohledňovat laický přístup. Právě proto zakázka na www prezentaci ČKR přinesla cenné zkušenosti se zákazníkem, pro nějž musel být vytvořen vhodný nástroj, umožňující nenáročnou a srozumitelnou aktualizaci datových zdrojů a následné vygenerování www stránek.

Literatura

[1] Procházková, Š., Ocelka, J.: Internetová prezentace MU v Brně. In UNINFOS 2000. Zborník príspevkov. Nitra: SPU v Nitře, 2000. ISBN 80-7137-713-9, s.186-189.

[2] Procházková, Š., Ocelka, J.: Automatizace univerzitní prezentace. In UNINFOS 2001. Zborník príspevkov. Zvolen: Vydavateľstvo TU vo Zvolene, 2001. ISBN 80-228-1062-2, s.124-128.

[3] Knihovna SAXON: `http://saxon.sourceforge.net/`

[4] Knihovna XALAN: `http://xml.apache.org/xalan-j/` □