Dancs Szabolcs

Az XML filológiai alkalmazása

Ma már nem jelent forradalmi újdonságot a dokumentumfeldolgozásban a jelölőnyelvek használata. Ezt támasztja alá, hogy a magyar szaksajtóban napvilágot látott nem egy cikk foglalkozik e kérdéssel (például: Salgáné Medveczki Mária [1]; Bíró Szabolcs [2]). Az XML megjelenése olyan eszközt adott a szakemberek kezébe, amely a szövegfeldolgozások munkafolyamatában történő alkalmazása révén nagymértékben növeli a dokumentumok metaadatok általi visszakereshetőségét. Hogy mindezeken túl a tudományos – filológiai: irodalomtörténeti, nyelvészeti – kutatások is hasznát látják a módszertan ilyetén megújhodásának, az alábbiakban felvázolt példa is alátámasztja.

Cikkemben a 2003-ban *Ljubljaná*ban megrendezett *13. Szlavisztikai Kongresszus*on elhangzott egyik előadás kivonatos ismertetésére vállalkozom azzal a kifejezett céllal, hogy meggyőző érvekkel korteskedjem – többek között – a kéziratoknak az XML jelölőnyelven történő feldolgozása mellett. Mindenekelőtt azonban röviden, áttekintő jelleggel bemutatom azokat az eszközöket és technológiákat, amelyek ismerete a tanulmány lényegi részének megértéséhez elengedhetetlen. Ezek az eszközök: az *XML*, az *XSLT* és az *SVG*.

A dokumentumfeldolgozás és a megjelenítés eszközei

Az XML és a DTD

Az *Extensible Markup Language* (*XML, http://www.w3.org/XML/*) jelölőnyelv (másként: leírónyelv) fontossága a szövegkódolás szempontjából abban rejlik, hogy az ún. tagek (vagy tagok) használatával lehetővé teszi egyrészt az egyes szövegelemek (művek, mondatok, frázisok, szavak) visszakereshetőségét, másrészt az egységek ilyetén jelölése révén megfelelő alapot hoz létre a rajtuk végzendő további műveletek megvalósításához. Az XML az SGML egyszerűsített változata, s abban különbözik a HTML jelölőnyelvtől, hogy az utóbbinak előre definiált taghalmaza van, míg az XML lehetőséget nyújt saját tagek definiálására *(1. ábra*).

Az XML dokumentum egy eleme egy nyitó (<vers>) és egy záró (</vers>) tagből épül fel, és – a nyitó tagen belül – tartalmazhat ún. attribútumokat (strszam, sorszam). Az adott XML dokumentum szintaktikáját ún. *DTD*-ben (*Document Type Definition*) szokás megadni. Megfelelő XML szerkesztő segítségével létrehozhatunk olyan jól formázott XML dokumentumokat, amelyek megfelelnek az általunk definiált (s hivatkozott) DTD-ben meghatározott szintaktikai szabályoknak. DTD-k definiálásához használhatunk különböző célalkalmazásokat (PizzaChef, Roma), de – ha szövegek, illetve dokumentumok kódolásáról van szó – célszerű olyan, nemzetközileg elfogadott és alkalmazott, *kváziszabványnak* tekinthető sémákat használnunk, mint amilyenek a *Text Encoding Initiative* (*TEI, http://www.tei-c.org/*) szervezet szakemberei által előre definiált DTD-k (*2. ábra*). (Sémákat hozhatunk létre más módon is, pl. a Relax NG sémanyelv használatával.)

```
<vers>
<szerzo>Pilinszky János</szerzo>
<cim>Négysoros</cim>
<strofa strszam="1">
<sor sorszam="1">Alvó szegek a jéghideg homokban,</sor>
<sor sorszam="2">Plakátmagányban ázó éjjelek.</sor>
<sor sorszam="3">Égve hagytad a folyosón a villanyt.</sor>
<sor sorszam="4">Ma ontják véremet.</sor>
</strofa>
</vers>
```

1. ábra **Egészen egyszerű XML dokumentum**

Az XSLT

Az XSLT (XSL Transformation, *http://www.w3.org/TR/xslt* = XSL átalakító) az XSL (XML Stylesheet Language = XML stíluslapnyelv) család tagja. Az XSLT XML-ben fejeződik ki, XML-fákat ír és olvas,

```
<?xml version="1.0"?>
<!DOCTYPE vers [
  <!ELEMENT vers (szerzo,cim,strofa+,sor+)>
  <!ELEMENT szerzo (#PCDATA)>
  <!ELEMENT cim (#PCDATA)>
  <!ELEMENT strofa (#PCDATA)>
  <!ELEMENT sor (#PCDATA)>
  <!ATTLIST strofa strszam CDATA #REQUIRED>
  <!ATTLIST sor sorszam CDATA #REQUIRED>
]>
```

2. ábra **Az XML dokumentumhoz tartozó – egészen egyszerű – DTD**

valamint széles körben elterjedt módszer HTML-ek generálására. Segítségével az XML dokumentumokat publikálásra alkalmas formára hozhatjuk (*3. ábra*). Ahogy az a későbbiekből kiderül, az XSLT alkalmazása korántsem merül ki ennyiben.

```
<xsl:stylesheet version=”1.0”>
<xsl:template match=”vers”>
<html>
<h1>
<xsl:value-of select=”szerzo”/>:
</h1>
<h2>
<xsl:value-of select=”cim”/>
</h2>
<p>
<xsl:apply-templates select=”strofa/sor”/>
</p>
</html>
</xsl:template>
</xsl:stylesheet>
```

3. ábra **Az XML dokumentum transzformálása HTML-lé**

Az SVG

Az *SVG* (*Scalable Vector Graphics, http://www.w3.org/TR/SVG/*) komplex grafikus objektumok rajzolására alkalmas XML tagkészlet. Az előzőekhez hasonlóan ez is a W3 konzorcium egyik nyílt szabványa. Az SVG előnye, hogy megfelelő SVG-viewer, illetve böngésző használatával grafikus képként jeleníthető meg, jóllehet valójában olyan XML dokumentumról van szó, amely egyszerű szöveges tartalomból és XML jelölőelemekből épül fel, így szövegként és XML dokumentumként egyaránt megtekinthető és szerkeszthető. Részletes taglalását itt mellőzzük. A legfontosabb, amit a definíciójában már említettünk, hogy XML-alapú, s ez a tulajdonsága teszi lehetővé az alábbiakban felvázolt filológiai alkalmazását [3].

Az XML filológiai alkalmazása középkori miszcelláneák példáján

A cél

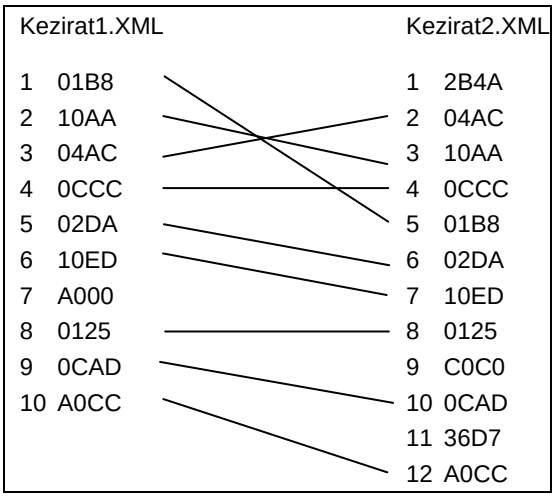
E szükséges kitérő után térjünk vissza a fent említett előadás ismertetésére! Az előadó, *David J. Birnbaum*, a *Pittsburghi Egyetem* szlavista professzora előadásának címe magyarul: „Vegyes tartalmú miszcelláneák struktúrájának számítógéppel támogatott analízise és vizsgálata” [4]. Már a címmel is némi gond támad, mivel a *vegyes tartalmú miszcellánea* kifejezés a magyarban értelmetlen, pontosabban tautológia; a miszcellánea fogalma magában foglalja a *vegyes tartalmúságot*.

A szerző által vizsgált miszcelláneákban fellelhető különböző, voltaképpen tetszés szerint összeválogatott szövegek elrendezése nem követ semmiféle szervezési elvet. Birnbaum ugyanakkor a szövegek egymásutánisága által meghatározott struktúra analízisét tűzi ki célul, pontosabban – a struktúra elemzésével – a szövegek átvételének, transzmissziójának (textual transmission) vizsgálatára tesz kísérletet. A kéziratos könyvek ugyanis tudvalevően másolással jöttek létre, és a másolandó írásokból az írnok (a másoló) olykor saját ízlésének megfelelően − olykor valamely célt szem előtt tartva − szortírozott. Ahhoz, hogy megállapíthassuk, mely kéziratok hasonlítanak a legnagyobb mértékben egymásra, pontosabban, hogy lokalizálni és azonosítani tudjuk az egymással kapcsolatban lévő kéziratokat $n(n–1)/2$ (n = az összehasonlítandó kéziratok száma) összehasonlítás szükséges, ami jelentős méretű korpusz esetén nehezen volna kivitelezhető számítógépes támogatás nélkül. Az összehasonlítási művelet mellett szükség van az eredmények grafikus megjelenítésére is, amit ún. „plektogramok” formájában képzel el a szerző (*4. ábra*).

A megvalósítás lépései

Birnbaum felteszi a kérdést, miként oldható meg a fentiekben felvázolt két feladat (összehasonlítások, grafikus megjelenítés) nyílt és könnyen hozzáférhető szabványok segítségével, olyan fájlok révén, amelyek létrehozása nem igényel többet egyszerű szövegszerkesztőnél. A szerző javaslata: a kéziratok tartalmát a TEI XML DTD-inek módosított

verziója szerint kell kódolni. (A TEI-ről magyarul rövid ismertető készült [5].) Mint tudjuk, az XML-nek nem elsődleges célja a grafikus megjelenítés, de arra is alkalmas. Az XML adatokhoz XSLT-vel férünk hozzá [6], a további műveleteket ugyancsak XSLT-vel (esetleg *Spitbol*lal, *http://www.snobol4.com/*) vezényeljük le. A végső szöveges output XML nyelvű lesz, így bármely XML böngésző meg tudja majd jeleníteni. A plektogramokat SVG-ként formázzák meg.



4. ábra **Egy hipotetikus plektogram**

Birnbaum az egymással kapcsolatban lévő kéziratok azonosítására a következő – ötlépéses – algoritmust dolgozta ki:

- az egyes kéziratokban lévő cikkek szöveges formátumú listáinak létrehozása XSLT segítségével;
- az egyes kéziratok cikklistáinak egyesítése egyetlen listában az ismétlődések kizárásával, ami az Unixban a következő módon oldható meg: a *cat* parancs egyesíti az első lépésben létrehozott listákat, majd a *sort*, valamint az *uniq* parancsokkal elkészítjük az egyesített listát; ezután egy Spitbol szkript segítségével a lista minden egyes sorát ellátjuk egy – négyjegyű hexadecimális – indexszámmal;
- az egyes kéziratokhoz tartozó cikklisták létrehozásával, és az összes cikkhez különálló indexszámok hozzárendelésével megteremtettük az egyes kéziratok tartalmának *kódolt reprezentációját*, amely megkönnyíti a tartalmi összehasonlítást. A cikkcímek összehasonlítása XSLT-vel is lehetséges volna, az indexszámok feldolgozása azonban – uniformizált hosszúságok révén – sokkalta egyszerűbb;
- az összehasonlítandó kéziratok párosítása lista formájában (természetesen minden pár két tagját csak egyszer kell összehasonlítani, vagyis ha egyszer összevetettük *X* tartalmát *Y*-éval, *Y* tartalmának *X*-ével való összevetésétől eltekinthetünk);
- az indexszámok összehasonlítása minden kéziratpár esetében, valamint egy nemnegatív integer érték hozzárendelése minden párhoz a hasonlóság fokának jelölésére.

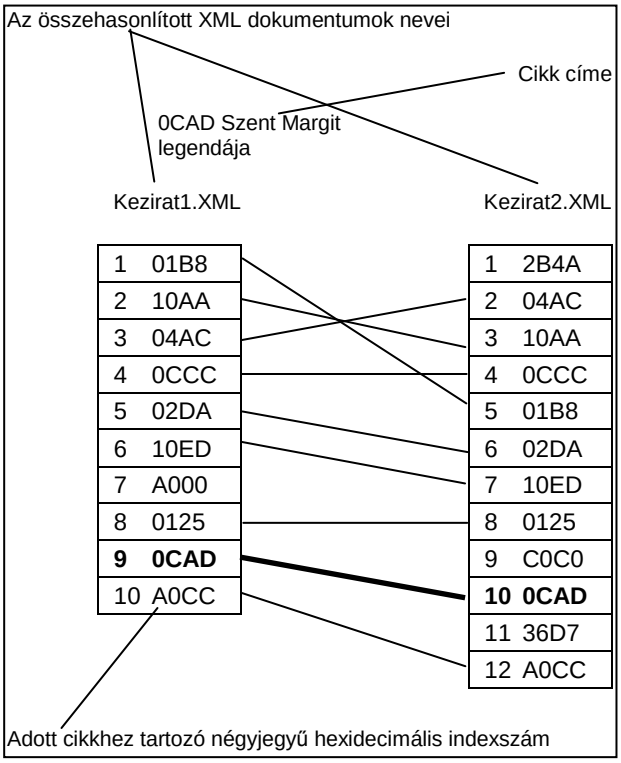
Az összehasonlítás – négylépéses – algoritmusa:

- a hosszabb és a rövidebb kézirat azonosítása – az utóbbit nevezzük el *N*-nek, az előbbit pedig *M*-nek;
- az *n* számú cikket tartalmazó *N* kézirat esetén próbálkozzunk először az *n* számú (vagyis az összes) cikk szekvenciájának *M* kéziratban való megtalálásával, ezután az *n–1* számú cikkek szekvenciájának meglelésével, s így egészen az *egycikkes* találatokig;
- a hasonlóságok mérlegelésekor a *résztalálatok* is számítanak (vagyis azok az esetek, amikor például két kézirat között két-három cikk hosszúságú szekvenciára kiterjedő egyezést azonosítunk). Ugyanakkor ezek kevésbé nyomnak a latban, mint a több cikkre kiterjedő egyezések, vagyis – példánknál maradva – két-három cikk hosszúságú egyezés a hasonlóságok értékelésekor kevesebbet *ér*, mint egy hat cikk hosszúságú egyezés; a cikkek abszolút lokációja (vagyis hogy hányadikak az adott kéziratban), valamint abszolút számuk a kéziratban (vagyis hogy hány cikk van adott kéziratban) nem befolyásolja a hasonlóság értékelését;
- a két kézirat közötti hasonlóság mértéke az összes találat összege.

A megjelenítés

Fontos – és a kérdéskör iránt érdeklődő számára komoly segítséget jelent – a kéziratok közötti rokonság grafikus megjelenítése, amelynek megvalósítására az SVG alkalmazását javasolja a szerző. Az eddigiekből kiderül, hogy mind az inputfájlok, mind az SVG-vel kódolt plektogramok XML dokumentumok, ami egyrészt megkönnyíti az eredeti fájlok XSLT transzformációval történő plektogrammá alakítását, másrészt lehetővé teszi, hogy az ily módon létrehozott SVG outputok ugyancsak jól formált XML dokumentumok legyenek. Megjegyzendő: az XML technika egyik fontos erőssége, hogy az XSLT transzformáció segítségével a szükséges információkat a kéziratleírásokból vonhatjuk ki, majd manipulálhatjuk oly módon, hogy

végül egy plektogramot hozzunk létre, mindezt közbülső fájl használata nélkül. Az SVG mindamellett gazdag eseménykezelő készlettel rendelkezik, mint amilyen például az *onmouseover*. Esetünkben ez azt jelenti, hogy az eszköz megfelelő alkalmazásával pl. elérhetjük, hogy egerünket a létrehozott plektogram egyes cikkeihez tartozó négyjegyű hexadecimális indexszám fölé mozgatva, a képernyő felső részén megjelenjen a cikk teljes címe, mint ahogy ez az *5. ábrán* megfigyelhető.



5. ábra **A hipotetikus plektogram „működés közben", magyarázatokkal**

Mint látjuk, a feldolgozási technológia terén az utóbbi esztendőkben zajló változások hatására ma már olyan – szabványos – módszerek, technikák állnak rendelkezésünkre (XML, XSLT, SVG), amelyek sokrétű alkalmazására, lehető legteljesebb kiaknázására különböző szakterületek különböző nemzetiségű képviselői tesznek erőfeszítéseket világszerte. A szöveg-, illetve dokumentumfeldolgozás területe ilyesformán tükrözi azokat a (mega)trendeket, amelyekkel a világot jellemezni szokásunkká vált. Így például a metodika egységesítésének folyamata a feldolgozandó dokumentumok tipológiai, származási stb. heterogenitásával párosul. Helyzetünk tehát hasonló a nyelvészéhez, akit egyszerre ösztökél a minden nyelv mögött meghúzódó – feltételezett – mélystruktúra feltárásának gondolata, valamint az egyes nyelvekre jellemző, specifikus jelenségek megismerése. Ahhoz, hogy az elektronikus feldolgozás terén is lépést tudjunk tartani a fejlett országokkal, egyrészt igyekeznünk kell a magunk módján hozzájárulni az olyan nemzetközi szervezetek munkájához, amelyek a leírás nemzetközi szabványosítását igyekeznek megvalósítani (TEI), másrészt meg kell honosítanunk a szabványosítás eddigi eredményeit olyan területeken, mint amilyen például a középkori kéziratok, kódexek feldolgozása.

Irodalom

[1] SALGÁNÉ MEDVECZKI Marianna: Az XML: új perspektívák a könyvtár-informatikában. = Tudományos és Műszaki Tájékoztatás, 51. köt. 2. sz. 2004. p. 61–71.

[2] BÍRÓ Szabolcs: Van új a nap alatt – XML alapú webtartalom-generálás Cocoon rendszerrel. = Tudományos és Műszaki Tájékoztatás, 52. köt. 11−12. sz. 2005. p. 510–519.

[3] EISENBERG, J. David: SVG kézikönyv. Budapest, Kossuth, 2003. 342 p.

[4] BIRNBAUM, David J.: Computer-Assisted Analysis and Study of the Structure of Mixed Content Miscellanies. = Scripta & e-Scripta, 1. köt. Sofia, 2003.

[5] GOLDEN Dániel–TÓTH Tünde–TURI László: Virtuális örökkévalóság: objektumok a digitális könyvtárban. = Tudományos és Műszaki Tájékoztatás, 45. köt. 8–9. sz. 1998. p. 299–314.

[6] BÍRÓ Szabolcs: A szövegfeldolgozás modern eszközei – az SGML és XML nyelvek. = Tudományos és Műszaki Tájékoztatás, 51. köt. 10. sz. 2004. p. 453–459.

Beérkezett: 2006. II. 13-án.

Dancs Szabolcs
nyelvész,
informatikus könyvtáros,
az MTA Könyvtára katalogizáló osztályának munkatársa.
E-mail:
dancsz@vax.mtak.hu