1.DTP ソフトと FO による自動組版

様々なデータを XML 形式で表現しようという動きが急速に広まっています。XML はテキストなどのコンテンツをタグで囲んでデータを表現する方法です。

XML 文書を作成するには、コンピュータを使って自動的に行う方法、編集ソフトを使って行う方法など色々あります。

編集ソフトを使って XML 文書を作成するのは、Microsoft Word などを使って文書を作成するのと比べて手間が掛かります。これは XML がエンドユーザになかなか普及しない原因の１つです。

文書の形式として XML を採用する場合、どのような点にメリットを見出すかを慎重に見極めないと従来よりも作業が複雑になるだけの結果に終わりかねません。

では、XML のメリットはなんでしょうか？良く言われることは、XML を導入することでデータの再利用が容易になるということです。

たとえば、XML で記述した文書を、あるときは HTML に変換して WEB サイトで公開し、同じ文書を別の機会に印刷物で配布するというようなことが、元の XML 文書を手直しすることなく実現できます。

しかし、これだけでは決め手になりません。XML を採用することで、特に、編集ソフトを使って人間が作成する時の文書作成コストは、現在のツールの技術水準を前提にする限り、明らかに大きくなります。このコストを他の工程で吸収して、全体としてのコストを下げることができなければ XML の採用には踏み切れないでしょう。

トータル・コストを削減するための鍵は、コンピュータにより文書を自動処理することです。W3C が標準化した XSL-FO は、XML 文書の自動組版という観点から重要な技術ですが、これについての参考文献がまだ少ないためか、一部では非常に注目度が高いものの、まだあまり広く知られていないようです。

ここでは、XML 文書の自動組版について簡単に説明し、また従来の DTP ソフトによる組版との相違点について述べさせていただきます。

1-1.XML と XSL-FO

印刷関係の展示会等で「XML ってどうやって印刷するの？」というご質問をたくさんいただきます。

XML を自動組版するために XSL-FO という技術があるということは先ほど述べましたが、ここで XSL-FO のサワリの部分を紹介しながら、XML と XSL-FO との関係について簡単に説明します。

下記は簡単な XML 文書の例です。

```
<日記>
  <今日の出来事>
    <日付>2002 年 5 月 26 日</日付>
    <天気>晴れ</天気>
    <出来事>
      <p>遊園地へ行った。</p>
      <p>楽しかった。</p>
    </出来事>
  </今日の出来事>
  <今日の出来事>
    <日付>2002 年 5 月 27 日</日付>
        ～途中省略～
  </今日の出来事>
</日記>
```

これは日記を XML で記述したものです。何が書かれているのかは分かりますが、これをどのように印刷するのかという情報がここにはありません。このように、一般的に XML 文書内には書式情報を持たせません。

このデータの中にたとえば、フォントサイズは 12 ポイントで、などといった情報を持たせることは可能ですが、それはデータの再利用という点から見た場合、意味がないばかりか、邪魔な存在であるわけです。

この書式情報を持たない XML 文書を組版するためには、いったん書式情報を持ったデータに変換する必要があります。実はそれが XSL-FO です。XSL-FO は書式情報を持った XML 文書というわけです。それはたとえば次のようになるでしょう。

```
<fo:block font-family="MS ゴシック"  font-size="12pt">
  <fo:block font-style="bold">
    2002 年 5 月 26 日（晴れ）
  </fo:block>
  <fo:block>
    遊園地へ行った。
  </fo:block>
  <fo:block>
    楽しかった。
  </fo:block>
</fo:block>
```

このように元の XML 文書の中にあった<日付>であるとか<出来事>といった「意味のある」タグはなくなり、代わりに<fo:block>のようなタグに置き換わっています。同時にフォント情報が付加されています。

これを組版エンジンで処理すると MS ゴシック体の 12 ポイントで 3 行印刷され、日付と天気の行は太字になります。これが XSL-FO です。

この例では fo:block オブジェクトしか登場しませんが、これ以外にも表組を実現する fo:table-and-caption、画像ファイルを貼り付けるための fo:external-graphic など、その他様々なオブジェクトが用意されています。それらのオブジェクトを駆使して、複雑なレイアウトを実現することが可能です。

1-2.XSLT スタイルシート

XML 文書を XSL-FO に変換することで組版できることは分かりましたが、ここで「XSL-FO に変換するにはどうすればいいの？」という新たな疑問が生まれてきます。そこで登場するのが XSLT スタイルシートです。

XSLT スタイルシートは、すでにある XML 文書を別の形に変換するための仕様です（元々は XSL-FO に変換するために考えられたものですが、それ以外のものへの変換処理にも応用されています）。

ここでスタイルシートの簡単な具体例を見てみましょう。

```
<xsl:template match="今日の出来事">
 <fo:block font-family="MS ゴシック" font-size="12pt">
  <fo:block font-style="bold">
   <xsl:value-of select="日付"/>
   (
   <xsl:value-of select="天気"/>
   )
  </fo:block>
  <xsl:apply-templates select="出来事"/>
 </fo:block>
</xsl:template>
<xsl:template match="p">
 <fo:block>
  <xsl:apply-templates />
  </fo:block>
</xsl:template>
```

これは先に示した XML で書かれた日記を XSL-FO に変換するスタイルシートの一部分です。
ここではスタイルシートの技術的な側面には触れませんが、元の XML 文書に付けられたタグをどういう FO オブジェクトに変換するのか、というようなことを記述したものがスタイルシートです。
この例のように単純に元の文書を FO オブジェクトに変換することはもちろんのこと、表紙や目次、索引、辞書でよく見かけるツメなどを自動生成することも可能です。また、偶数ページと奇数ページでヘッダやフッタの内容を切り替えたいという要望にも応えられます。
スタイルシートはデータの並び替えや抽出の機能も持っているので、XML で書かれた名簿データをあるときは名前順に、あるときは住所順に出力したり、また東京に住む人だけのリストを作成することも（スタイルシートをほんの２～３行手直しするだけで）可能です。
たとえば上の日記の場合、日付に昇順に並べたり、逆に降順に並べたり、あるいは日付で印刷範囲をしぼり込むといったようなことが容易にできます。
また、日付と天気のみを印刷すれば、まったく別の意味を持った資料を得ることになります。
ここまでで、スタイルシートを少し手直しするだけで、元の XML データはそのままに、様々な出力結果を得られることがご理解いただけたかと思います。

1-3.DTP ソフトとの比較

ここで DTP ソフトによる従来の組版と XML+XSLT スタイルシートによる組版との比較をしてみましょう。
●ワンソース/マルチユース
代表的な DTP ソフトは XML で書かれたファイルを読み込む機能をすでに持っています。しかし、読み込んだファイルを GUI を使って書式情報を付加した時点で、完成したデータはその DTP ソフトの専用フォーマットとなってしまい、それは他の用途に使い回すことができません。
たとえば上記のように日付で印刷範囲を限定したり、印刷の順番を変えたい場合、また１から作業をやり直すことになります。これではせっかく XML でファイルを用意した意義がなくなってしまいます。
それに対し XML+XSTL スタイルシートの方法ですと、スタイルシートを差し替えるだけで様々なレイアウトの印刷結果を得ることができます。
このように XML の目標のひとつであるワンソース/マルチユースといった点で有利なわけです。
●手作業と自動処理
従来の DTP ソフトと XML+XSLT スタイルシートによる組版との最大の相違点は、手作業による組版なのか、それとも自動組版なのかという部分にあります。これにより両者のそれぞれの得意分野が見えてきます。
DTP ソフトの場合、素材を対話式でレイアウトして１ページ毎に手作業で作り上げていくので、コストは単純にページ数に比例することになります。
それに対し XML+XSLT スタイルシートの場合だと、最初にスタイルシートを作成するイニシャルコストがかかるものの、そのスタイルシートは全ページ共通で使用するものですから、ページ数が増えてもコストはあまり変わりません。その代わりタグセットの数やレイアウトの複雑さによってコストが決まります。
つまり、ある程度ページ数が多く、比較的単純なレイアウトが繰り返されるような場合に XML+XSLT スタイルシートによる自動組版が有利になることが分かります。
これを裏付けるように、機械の整備マニュアル等、文書構造がはっきりとし、レイアウトもさほど複雑ではない技術系の書物作成を担当されている方からのお問い合わせが最近増えているようです。
これに対し、チラシのような文書構造というよりはむしろ見た目重視のレイアウトで、しかも１ページしか印刷しないようなものの場合は、従来の DTP ソフトを使った方法が有利と言えます。

1-4.まとめ

DTP ソフトによる組版とスタイルシートによる自動組版にはそれぞれ得意分野があります。

ひとつの文書をいろいろな用途に使い回ししたい場合やページ数の多い書物を組版する場合、自動組版の方がコストパフォーマンスが高いことはご理解いただけたことでしょう。

逆にページ毎に様々な例外が発生し、とても自動組版することができないような場合、従来の方法に頼るしかありません。

まだ歴史は浅いですが、今後徐々に XML+XSLT スタイルシートによる組版が広がって行くのは間違いありません。