XML Data Representation in Document Image Analysis

Abdel Belaïd
Université Nancy 2
LORIA
Vandœvre-lès-Nancy, France
abdel.belaid@loria.fr

Ingrid Falk*
CNRS
ATILF
Nancy, France
ingrid.falk@loria.fr

Yves Rangoni
Université Nancy 2
LORIA
Vandœvre-lès-Nancy, France
yves.rangoni@loria.fr

Abstract

This paper presents the XML-based formats ALTO, TEI, METS used for Digital Libraries and their interest for data representation in a Document Image Analysis and Recognition (DIAR) process. In the first part we briefly present these formats with focus on their adequacy for structural representation and modeling of DIAR data. The second part shows how these formats can be used in a reverse engineering process. Their implementation as a data representation framework will be shown.

1. Introduction

In this paper we discuss the problem of data representation with standards in document image analysis and recognition (DIAR). This task involves the physical and logical modeling which both require a data representation. Since most systems use both structures simultaneously, the relations between them also need to be represented [5]. The task is made more difficult by the complexity of the mapping between physical and logical structure: they are rarely unique and in most cases there is no unambiguous mapping between them. A lot of ad-hoc formats have been developed for this purpose. Most of them have not been published and have only been used in specific applications. Some works such as DAFS [3] exist, but few of them have been generally adopted, indicating that they may not be extensible and general enough to comply with the requirements. The variety of formats prevents the easy exchange of data among different environments, platforms and even researchers. The only common ground in data representation for DIAR systems is probably the use of an XML-based format. This is a step towards the only common point of all the systems. Such a format endorses normalization but the chosen format is rarely immediately applicable for structuring and interchange. To overcome the problem of generic data representation, we developed a solution around the XML formats: TEI[1], ALTO[2] and METS[4]. The TEI is an established XML standard for the logical representation of a large variety of textual documents and is therefore a good candidate for representing the logical structural information. The recently emerged ALTO format is well suited to represent the recognized physical structure, especially when coming from an OCR system. Finally, among the many upcoming digitization projects, emerged a metadata encoding standard, METS, which proved adapted to encode the mapping between physical and logical structures.

2. Data Representation Formats for DIAR

ALTO [2] (Analyzed Layout and Text Object) emerged among the digital libraries communities as a format representing OCR results. It is an XML schema designed to represent the physical structural data together with the recognized text content of a text oriented document image. ALTO defines digitized texts as consisting of <Page> elements. A <Page> contains margin blocks and a <PrintSpace>. Each <PrintSpace> is of a <PageSpaceTypes> type and is divided in <TextBlock>s or graphical blocks. A <TextBlock> may further be divided into <TextLine>s, and these in character string blocks (<String>) and space blocks (<SP>). For each block, geometric and positional information relative to the image is preserved as attributes. Layout and typesetting information may also be recorded, as well as the confidence of a textual block. Information regarding the generating OCR process is stored in a metadata descriptive section. TEI [1] (Text Encoding Initiative) delivers guidelines as a set of standard XML schemes that enable libraries, museums, publishers, and individual scholars to represent a variety of literary and linguistic texts for online research, teaching and preservation. While designed primarily for human transcribers, the TEI guidelines and

*Work supported by the Contrat Plan Etat Région: "Ingénierie des Langues" of the French Région Lorraine.

schemes constantly evolved since the nineties and are now a well known and established representation format for the logical structure of textual content. Because of the great structural variety of texts, the TEI proposes to regard all textual divisions as occurrences of the same neutrally named elements, with an attribute type used to categorize elements independently of their hierarchical level: the ¡div¿ element. Depending on the type of the text the value of the type attribute may be chapter, act, etc. The ¡div¿ element is allowed to nest recursively, thus indicating hierarchic depth. Page breaks and line breaks are encoded as ”milestone” elements, i.e. empty XML elements: ¡pb/¿ and ¡lb/¿ respectively, and may be further specified by attributes to account for the page or line number-depending on the edition, a unique identifier, etc. METS[4] (Metadata Encoding and Transmission Standard) is an XML schema for encoding descriptive, administrative, and structural metadata regarding digital objects within a digital library. In the METS view, a digital object is a set of interlinked files and their sub-areas. Such a link may assign a sub-area of an image file to one or more physical structural units and/or to one or more logical structural units. A METS document consists of seven major sections, but only the file and structural map sections are mandatory. Regarding the metadata sections, its particular interest lies in the fact that different kinds of metadata (bibliographical, administrative, structural. . .) may be separately encoded, allowing other schemes, as ALTO or TEI to be plugged in. The structural map is the central point of a METS document. It outlines one or more interlinked tree structures for the digital object. The elements of that structure are linked to content files and metadata that pertain to each element. It also allows for the encoding of hierarchical structural relations between the files and/or parts thereof. Sub-areas of the context files may be referenced by coordinates (e.g. for image files), a time code (e.g. for audio files), an ID reference (for XML files) and a byte code.

3 Application to DIAR

3.1 Structure Conversion

Figure 1 shows a classical schema of a DIAR system based on data representation by XML. As all the intermediate steps (OCR, physical and logical structure extractions) produce their results in proprietary XML formats, a conversion by XSLT to ALTO/TEI/METS is required.

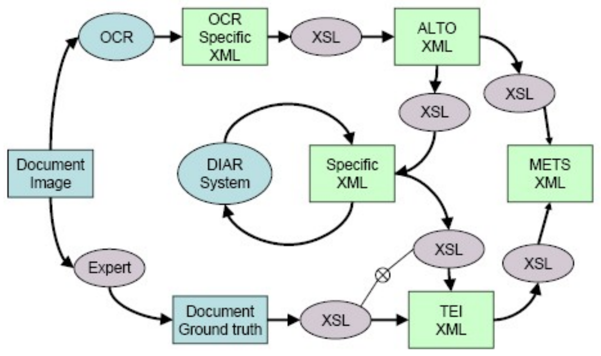


Figure 1. Data representation overview

3.1.1 From OCR to ALTO

The OCR XML transformation to ALTO is performed by conversion rules. Some of them just rewrite tags. The transformations are sufficiently simple to use XSLT. In the following example, we convert the OCR attribute, ”ff” , in the corresponding ALTO attribute ”FONTFAMILY”.

```
<xsl:attribute name="FONTFAMILY">
<xsl:value-of select="@ff"/>
</xsl:attribute>
```

In the following example, we compute the ALTO ”WIDTH” attribute for a block from the position of the lines contained in the block to be converted.

```
<xsl:attribute name="WIDTH">
    <xsl:value-of select="./line[position() = ’1’]/@r -
   ./line[position()= last()]/@l"/>
</xsl:attribute>
```

In the following example, after defining the text style, we can reuse it in several situations. The situation described below shows a text block composed of text lines with pointers (Stylerefs) to physical attribute structures.

```
<TextStyle ID=" Times_New_Roman_9."
FONTFAMILY="Times New Roman" FONTSIZE="9."/>
<TextBlock ID="B1" HPOS="1560" WIDTH="306"
  VPOS="149" HEIGHT="34">
  <TextLine BASELINE="175" ID="B1_L0"   HPOS =
  "1560" WIDTH="306" VPOS="149" HEIGHT="34"
  STYLEREFS="Times_New_Roman_9.">
    <String CONTENT="PS" HPOS="1560" VPOS="150"
       WIDTH="101"/>
       <SP HPOS="1661" VPOS="158" WIDTH="10"/>
    <String CONTENT="RP" HPOS="167" VPOS="158"
     WIDTH="195"/>
  </TextLine>
</TextBlock>
```

3.1.2 From the Logical Structure to TEI

The logical structure, produced either by a system or by an expert (ground truth) may produce a proprietary XML. The conversion to TEI is straightforward, as most of the DIAR system outputs have a TEI correspondent. Starting from this specific format, an XSLT sheet may produce a valid

TEI document using only small transformations. This is illustrated in the following example to create a section with its title. First, an element div is created accompanied by its number, determined between a Carriage Return and the first space, and its type (a part). The tag head is then created by copying the text of the performed block.

```
<xsl:element name="div">
  <xsl:attribute name="n">
    <xsl:value-of select="substring-after(
    substring-before (child::texte,' '),
    '&#xA;')"/>
  </xsl:attribute>
  <xsl:attribute name="type">PART
  </xsl:attribute>
  <xsl:element name="head">
    <xsl:value-of select="./texte"/>
  </xsl:element>

</xsl:element>
```

Finally, a TEI XML file contains only the logical structure, an ID, and the raw text, as shown in the following example:

```
<div n="4.2" type="PART">
   <head>Partitioning for Boundary Cells
   </head>
   <p xml:id="par0"> Boundary cells are partitioned even
   further by constructing two parallel planes
   within the cell.
   </p>
</div>
```

3.1.3 From ALTO-TEI to METS

The mapping between ALTO (representing the physical structure) and TEI (representing the logical structure) by METS is performed by connecting the correspondent elements in both structures. This connection is realized by referring to the (XML) unique identifiers. The specific mapping of elements of the physical structure to elements of the logical structure has to be established by the DIAR system. Provided this precondition, building the METS file can be largely automated. In the following example we connect a TEI paragraph (represented by ID=par1) with several ALTO text line blocks (ID=B3 L25 to B3 L30) and the corresponding image area in the file represented by id JPG_a2.

```
<div ID="par_1" LABEL="paragraph 1">
<fptr>//file pointer
   <area FILEID="TEI_3" BETYPE="IDREF"
   BEGIN="par1"/>
</fptr>
<fptr>
   <area FILEID="JPG_a2" SHAPE="RECT"
   COORDS="73,1361,1216,1669"/>
</fptr>
<fptr>
   <area FILEID="ALTO_a2" BETYPE="IDREF"
   BEGIN="B3_L25"/>

   <area FILEID="ALTO_a2" BETYPE="IDREF"
   BEGIN="B3_L30"/>
</fptr>
</div>
```

3.2 XSL Reverse Engineering

We have extended the use of these standards and the corresponding XSL transforms in the framework of a real and global DIAR system. This is illustrated in Figure 2 . The DIAR system uses as an input a structure model written in TEI-like and the OCR result converted in ALTO. The result corresponding to the specific structures expressed in METS accompanied with ALTO and TEI . This result can be compared to ground truth written in TEI. P and L denote the physical and logical structures.

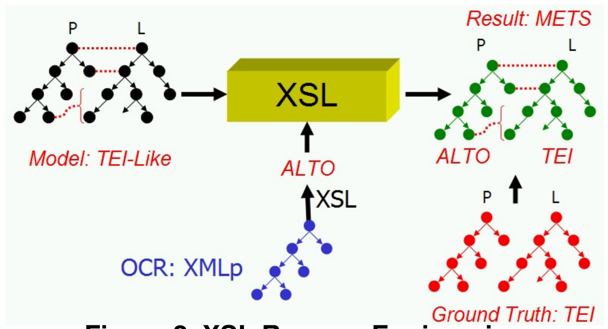


Figure 2. XSL Reverse Engineering

3.2.1 The document Class Model

The model describes [6] the a priori generic physical and logical structures of a document class. In fact, it is implemented by a logical structure (TEI-like) labeled physically. Each element of the structure is accompanied by a constructor like ”sequence” (SEQ), ”aggregate” (AGR), ”choice” (CHO), ”mosaic” (MOS) illustrating the gathering type of its subordinate objects. Separators (text, graphic, etc.) can be inserted between the subordinate objects. A qualifier concerning the occurrence of a subordinate object (i.e. optional (OPT), repetitive (REP), required (REQ), or optional and repetitive (OPT-REP)) are kept as defined in various schemas. Additionally, the model provides another qualifier (Optional Conditional (OPTC)) that can manage several problem such as the overlapping of a logical object onto several physical layout objects.

To represent this class model, we extended the TEI schema by some elements and attributes to take further advantage of the TEI generality. The following examples show the model of the signatory in the letter class. It is composed of a top down aggregate of a text block containing one or two text lines, a date with a specific format, and a figure representing the signature image. The date is optional conditional meaning that the date may appear either at this position or at the beginning but not both.

```
<signatory name="signatory">
  <AGR type="top-down">
    <TextBlock minOccurs="1" maxOccurs="1">
      <TextLine minOccurs="1" maxOccurs="2"/>
    </TextBlock>
    <date format="date" occurs="condition.xsl" params
        ="dateletter"/>
    <figure>
       <graphic />
    </figure>
  </AGR>
</signatory>
```

3.2.2 The Reverse Engineering Process

A top-down analysis is guided by the structure hierarchy in the model. The process starts from the TEI root and descends through the nodes checking the structure of each of them. This verification is done by mapping the logical element in TEI and its corresponding element in ALTO (OCR part) by considering its physical attributes. In case of a good match, the node is stored in an intermediate structure (a tree similar to METS with identifiers in the model and in ALTO). In case of failure, we backtrack and we proceed with another choice making use of the qualifiers. The intermediate structure is employed to generate the final result comprising ALTO-TEI-METS. We will give in the following some steps needed for the top-down sequence extraction.

a. Step1: Sequence location For each constructor in the letter model, we check its type (SEQ, AGR, etc.). We then recursively apply the XSL template corresponding to the constructor. For each subordinate object, we create a structural map as in METS in order to keep track of the recognized logical structure.

```
<xsl:template match="model">
<xsl:for-each select="child::*">
  <structMap>
    <xsl:if test="name()='SEQ' and @type='top-down'">
        <xsl:call-template name="childseqtd">
          <xsl:with-param name="vpos" select="0"/>
          <xsl:with-param name= "cnode" select="child::*[1]"/>
        </xsl:call-template>
    </xsl:if>
  </structMap>
</xsl:for-each>
</xsl:template>
```

b. Step2: Repeating elements The principle is as follows. When the current node has occurrence indications, we apply a template which returns all the ALTO elements reflecting this occurrence. These elements are stored in a local variable ($temp) which helps to control the occurrence limits.

```
<xsl:if test="$cnode/@minOccurs and $cnode/@maxOccurs">
  <xsl:variable name="temp">
     <xsl:call-template name="loopcheckTD">
  </xsl:variable>
   <xsl:if test="count($temp/*) &gt; $cnode/@minOccurs">
    <xsl:copy-of select="$temp/*[position() &lt;=$cnode/
   @maxOccurs or $cnode/@maxOccurs='unbounded']"/>
   </xsl:if>
</xsl:if>
```

c. Step3: Element verification In this template, we first check whether the structure of the current element corresponds to a valid element in ALTO. In case of success, the identifiers of the recognized ALTO elements are returned. These identifiers are employed at the same time for occurrence checking and to help subsequent processing.

```
<xsl:template name="verifelement">
  <xsl:if test="name($cnode)='TextBlock'">
  <xsl:for-each select="document
   ($fichier_alto)//TextBlock[@VPOS &gt; $vpos][1]">
    <xsl:if test="count(./TextLine)&gt;= $cnode
     /ligne[1]/@minOccurs and ((count(./TextLine) &lt;=
     $cnode /ligne[1]/@maxOccurs) or $cnode /ligne[1]/
     @maxOccurs='unbounded')">
   <ident><xsl:value-of select="@ID"/>
   </ident>
 </xsl:if>
</xsl:for-each>
  </xsl:if>
</xsl:template>
```

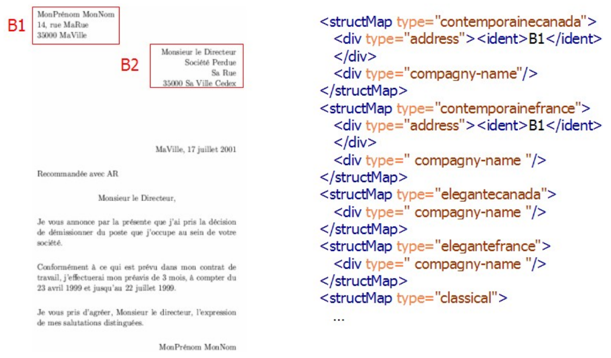
3.3 XSLT vs DOM

We experimented the same reverse engineering process by using JavaScript/DOM. Although the programming is easier because we used the same XML representation, we encounter the recurrent problem of the sequential traversing of XML tree which is more naturally solved in XSL.

4 Experiments

We experimented this data representation with scientific articles, an ancient dictionary (Trvoux) and letter document models. For the former, we used a neural network recognition system [5]. For the two other experiments we only used XSL templates. In our experience, we found that this framework is very appropriated for structure representations, reinforced by the power of ALTO/TEI/METS. XSL appears sufficient when the document structure is not too complex (such as letters, reports, dictionaries, etc.). However, for complex structures as in scientific articles, forms, ancient documents, an external recognition system seems more appropriate.

Figure 3 illustrates the XSL processing on an example of a letter document. The system fails on the company name and then proceeds to another hypothesis. The good result is presented in 4.



MonPrénom MonNom
14, rue MaRue
35000 MaVille

Monsieur le Directeur
Société Perdue
Sa Rue
35000 Sa Ville Cedex

MaVille, 17 juillet 2001

Recommandée avec AR

Monsieur le Directeur,

Je vous annonce par la présente que j'ai pris la décision de démissionner du poste que j'occupe au sein de votre société.

Conformément à ce qui est prévu dans mon contrat de travail, j'effectuerai mon préavis de 3 mois, à compter du 23 avril 1999 et jusqu'au 22 juillet 1999.

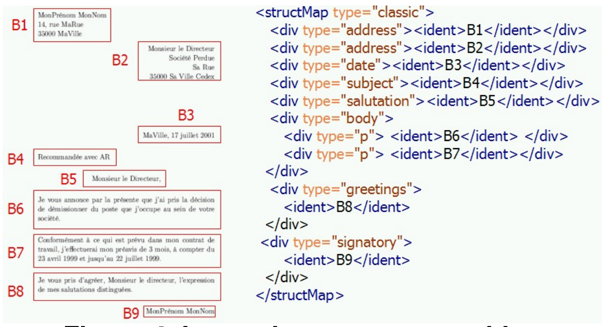
Je vous pris d'agréer, Monsieur le directeur, l'expression de mes salutations distinguées.

MonPrénom MonNom

```
<structMap type="contemporainecanada">
   <div type="address"><ident>B1</ident>
   </div>
   <div type="compagny-name"/>
</structMap>
<structMap type="contemporainefrance">
   <div type="address"><ident>B1</ident>
   </div>
   <div type=" compagny-name "/>
</structMap>
<structMap type="elegantecanada">
   <div type=" compagny-name "/>
</structMap>
<structMap type="elegantefrance">
   <div type=" compagny-name "/>
</structMap>
<structMap type="classical">
   ...
```

Figure 3. Recognition failure and backtrack

Figure 4 illustrates the XSL processing on an example of a letter document.



MonPrénom MonNom
14, rue MaRue
35000 MaVille

Monsieur le Directeur
Société Perdue
Sa Rue
35000 Sa Ville Cedex

MaVille, 17 juillet 2001

Recommandée avec AR

Monsieur le Directeur,

Je vous annonce par la présente que j'ai pris la décision de démissionner du poste que j'occupe au sein de votre société.

Conformément à ce qui est prévu dans mon contrat de travail, j'effectuerai mon préavis de 3 mois, à compter du 23 avril 1999 et jusqu'au 22 juillet 1999.

Je vous pris d'agréer, Monsieur le directeur, l'expression de mes salutations distinguées.

MonPrénom MonNom

```
<structMap type="classic">
  <div type="address"><ident>B1</ident></div>
  <div type="address"><ident>B2</ident></div>
  <div type="date"><ident>B3</ident></div>
  <div type="subject"><ident>B4</ident></div>
  <div type="salutation"><ident>B5</ident></div>
  <div type="body">
    <div type="p"> <ident>B6</ident> </div>
    <div type="p"> <ident>B7</ident></div>
  </div>
  <div type="greetings">
    <ident>B8</ident>
  </div>
 <div type="signatory">
    <ident>B9</ident>
  </div>
</structMap>
```

Figure 4. Letter document recognition

On the left side of the figure, we observe the physical structure as recognized by the OCR. On the right side, the XSL result shows the intermediate structure used for the METS and TEI file generation. It shows the links between physical blocks and logical elements of the letter document model.

5 Conclusion

We showed in this paper that the XML formats ALTO/TEI/METS can be used as a data representation framework for DIAR processes. This data representation model is largely compliant to the data representation model established until now: the physical structure is represented in an ALTO file, the logical structure by one or more TEI schemes and conformant documents, and the mapping between physical and logical structures by a METS document. It has the advantages of using the ALTO, TEI and METS XML schemes which are supported and developed by large and active communities: those working on (ideally) all kinds of texts and the digital libraries communities. These are also potential beneficiaries of document analysis processes and a common data representation format can only be supportive to exchange. We would also like to draw attention to the fact that using this mechanism does not restrict us to use ALTO or TEI. We could use any XML document format for representing a physical structure and equally any XML format for representing a logical content structure. It is also possible to integrate almost any other arising data representation, provided it is in an image format, an audio format, or an XML format.

References

[1] *The Text Encoding Initiative*, `www.tei-c.org`.

[2] *Analyzed Layout and Text Object*, `www.ccs-gmbh.com/alto/`

[3] Dov Dori et al. *The Representation of Document Structure: A Generic ObjectProcess Analysis*, in Handbook on Optical Character Recognition and Document Image Analysis, Eds P. S. P. Wang and H Bunke, (1996) Chapter 17

[4] *Metadata Encoding and Transmission Standard* `http://www.loc.gov/standards/mets/`

[5] Rangoni Y., Belaïd A., *Document Logical Structure Analysis Based on Perceptive Cycles*, 7th IAPR Workshop on Document Analysis Systems - DAS 2006, Springer Verlag (Ed.), (2006), 117–128

[6] Belad A. and Chenevoy Y., Constraint Propagation vs Syntactical Analysis for the Logical Structure Recognition of Library References, BSDIA, p.153-164, Lecture Notes in Computer Science, Vol. 1339. Nov. 1997.

[7] G. Nagy, Twenty Years of Document Image Analysis in PAMI, Pattern Analysis and Machine Intelligence, January, 1, 2000, 22, p38-62,