IDEA GROUP PUBLISHING
701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com
ITB12857

This paper appears in the book, *Emerging Trends and Challenges in Information Technology Management, Volume 1 and Volume 2* edited by Mehdi Khosrow-Pour © 2006, Idea Group Inc.

Managing Concurrent XML Structures: The Multi-Structured Document Building Process

Noureddine Chatti, Sylvie Calabretto, & Jean-Marie Pinon
LIRIS-INSA de LYON, 7 av. Jean Chapelle, F-69621 Villeurbanne Cedex - France, T 0033472436174, F 0033472438713
{noureddine.chatti, sylvie.calabretto, jean-marie.pinon}@insa-lyon.fr

ABSTRACT

In this paper we deal with the problem of multiple structuring of documents. We have proposed a specific model called Multi-structured Document Model (MSDM) approaching the problem in a generic way. To build a multi-structured document from existing XML structures of the same initial document, we have developed a parser called Multi-XML-Parser (MXP). This parser will be integrated in a multi-structured document management system.

1 INTRODUCTION

Actually, XML is a standard for content document structuring. It allows encoding of hierarchical structures. A problem arises when we want to define and to manage simultaneously different structures for the same contents. In this case, these structures are called concurrent, since they share the same content. For example, humanities and more particularly the study of mediaeval manuscripts, imply concurrent hierarchies or structures. Indeed, we can consider two main structures on manuscripts that overlap: the manuscript book structure (a sequence of columns and lines) and the "syntactic" structure (a sequence of phrase and words). Another structure in this domain can be the "damaged" structure (a sequence of damaged elements). The TEI guidelines [8] provide various examples of possible multiple structures. Among them, we can mention: in verse drama, the structure of acts, scenes and speeches often conflicts with the metrical structure.

It is very difficult to encode multiple structures in the same XML file. In fact, often, the result of the structures superposition cannot be a well formed XML document due to the structures interlacing (elements overlapping problem). The XML tree model is suitable only for a single hierarchy. The concurrent structures management problem has been encountered by our industrial partner, the CNAF-CNEDI (the National Computer Science Research Center of the Caisse Nationale d'Allocations Familiales) which manipulates legal texts through two different structures: logical structure and semantic structure. The logical structure is defined for the visualisation needs, and the semantic structure is specified to be used by inference systems. These structures are encoded separately in XML format. The main disadvantage of this solution is the content redundancy which makes difficult the document evolution management and may lead to content incoherency. To overcome this limitation, we have developed a parser called MXP (Multi-XML Parser) which allows to build a unified representation of several separate XML structures of the same content. This unified representation built by the MXP parser is named a *multi-structured document*. The MXP parser will be integrated in a specific environment dedicated to multi-structured document management.

2 RELATED WORK

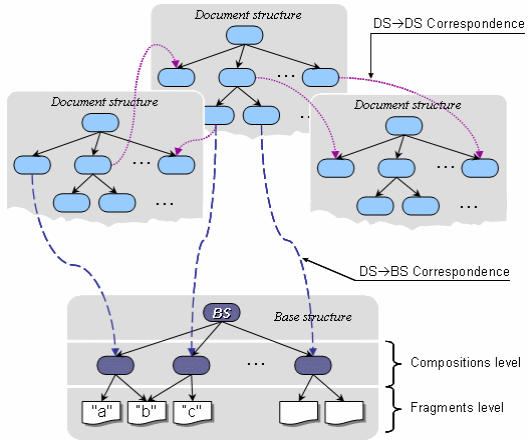
The problem of concurrent structures encoding has attracted many attentions, and several approaches has been proposed. The CONCUR option [1] is an SGML functionality which allows referencing of several parallel DTDs for the same content. In such SGML document, all structures cohabit in a single file. In this file, the first structure is encoded in a standard way, and for every added structure, a special prefix, denoting the reference to the corresponding DTD, is associated with each start tag. This solution is interesting but it was rarely implemented. For XML, that does not support multiple structuring, the problem is more persistent. In the TEI guidelines, several methods have been proposed to allow encoding of multiple hierarchies [9] in XML. These methods consist in fragmenting elements which do not nest within others. The TEI proposals cannot answer the general problem of the multiple structures encoding because they are not based on appropriate and clear models. To bridge the gaps of existing markup languages, some other works have been carried out in order to define new syntaxes. MECS (Multi-Element Code System) [2] was the first proposed language which allows overlapping between elements. TexMECS [3] is based on MECS language, but it is more complex. This language defines complex structures where elements can have multiple parents. LMNL (Layered Markup and aNnotation Language) [4] defines a specific syntax based on the notion of range, allowing the encoding of multiple structures where elements can overlap. Due to their complexity and incompatibility with XML syntax, these languages remained at experimental stages.

3 THE MULTI-STRUCTURED DOCUMENT MODEL

To answer the problem of multiple structuring, we have proposed a specific model, called Multi-Structure Document Model (MSDM) [5]. In MSDM, the problem is approached in a more general way. In fact, we suppose that structures can share just some content fragments, and not necessarily exactly the same content. For our model, concurrent structures are a particular case of multi-structured documents. In this model, which is inspired by the model defined in [6, 7], a multi-structured document is defined using the following notions:

- **Document Structure (DS)**: this is a description of a document content defined to a specific use. Such structure may be, for example, a physical structure defined for a presentation goal.
- **Base Structure (BS)**: this structure is visible only internally within the multi-structured document. It is defined strictly in order to organize the content in disjoint elementary fragments. These fragments serve to reconstitute, by composition, the original content associated initially to the document structure elements.
- **Correspondence**: a correspondence is a relation between two elements of two distinct structures. The origin of a correspondence is always an element of a document structure. If the correspondence target is an element of the base structure the correspondence is noted *DS®BS*. This kind of correspondence associates an element of a document structure to its content in the base structure. For example, in Figure 1 the first correspondence on the left associate the text content "a b" to the origin element in the document structure. When the correspondence target belongs to a document structure the correspondence is

Figure 1: Illustration of the multi-structured document model



denoted *DS®DS*. The correspondences *DS®DS* allow to make explicit some hided relations between document structures. Such correspondence may be used to express a synonymy relation between two elements for example.

As shown in Figure 1 a multi-structured document is defined by a set of document structures, a base structure and a set of correspondences. In a short representation, a multi-structured document may be defined as the following triplet: <*BS*, {*DS*}, {*DS® BS*, *DS® DS*}>.

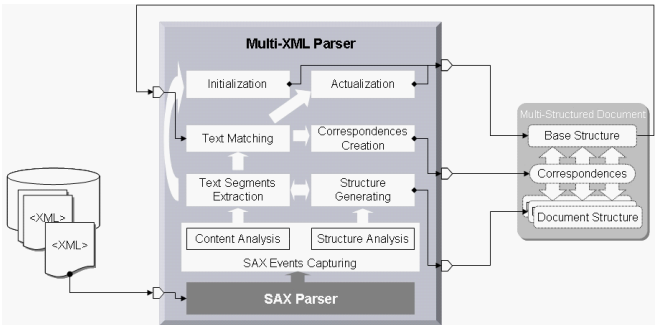
4 THE MULTI-XML PARSER

We are developing, actually, a multi-structured document management system, based on a MSDM implementation. To facilitate multi-structured document production from existing XML structure, we have developed a specific parser called Multi-XML Parser (MXP), which will be integrated to the multi-structured document management system. This parser allows generating a multi-structured document from separate XML structures of a same initial document. To build the multi-structured document the MXP parser performs several steps (Figure 2).

As shown in Figure 2, the MXP parser is based on a SAX parser (*Simple API for XML*) [SAX 02]. SAX is an event-driven model for processing XML. Unlike the DOM (Document Object Model) parser, SAX does not build a complete representation of the parsed document in memory. When SAX reads an XML document, it dispatch events (like start element, end element etc.) that we can capture and treat them in an implementation of the event handler interface. This method is more difficult but it offers better performances.

The parsing process of multiple XML structures may be divided in two main phases: the initialisation phase, and the actualisation phase.

Figure 2: The Multi-XML Parser



4.1 Initialisation of the Multi-structured Document

When the first XML file is passed to the MXP parser the multi-structured document, initialisation has begun. During the analysis of the first file, the base structure is initialized, the first document structure is generated and the needed correspondence relations between these tow structures are established. The base structure is initialized with the set of text fragments (PCDATA) tagged in the first XML structure. For each new analyzed PCDATA a new fragment element in the base structure is inserted. Afterwards, a new correspondence linking this fragment with the element in document structure, containing initially the new PCDATA, is created.

During this first phase some information will be stored in memory in order to be used in the next phase. The textual content of the first structure is stored as a string after removing all whitespaces. This string is named *CompactContent*. In order to facilitate the text-matching process, needed for the second phase, a mapping table is created to store, for each fragment reference (f_1, f_2, etc. in Figure 3) in the base structure, his corresponding text start position in *CompactContent*. The middle frame in Figure 3 shows an example of a multi-structured document with the *CompactContent* string and the mapping fragments table after the initialization phase.

4.2 Actualization of the multi-structured document

After the initialisation phase, a first version of the multi-structured document is created. However, this multi-structured document has only one document structure. The parsing of an additional XML file will then actualize the multi-structured document by inserting a new document structure. During the actualization phase three operations will be performed: the new document structure generation, the actualization of the base structure, and the creation of the correspondence relations. Before creating a new correspondence with the base structure when a new PCDATA is encountered, MXP tries to retrieve all fragments in the mapping table that matches (entirely or partially) with this one. This action is performed by means of a *correspondence retrieval algorithm*, which takes into account the fact that the text content in each structure is not necessarily identical.

We consider the following variables:

- CompactContent is the variable that contains the text content of the first structure without whitespaces.
- FragMap is the mapping table of fragment positions in CompactContent.
- _PCDATA contains the text of a PCDATA without whitespaces. For example if PCDATA = “a text fragment” then _PCDATA = “atextfragment”.

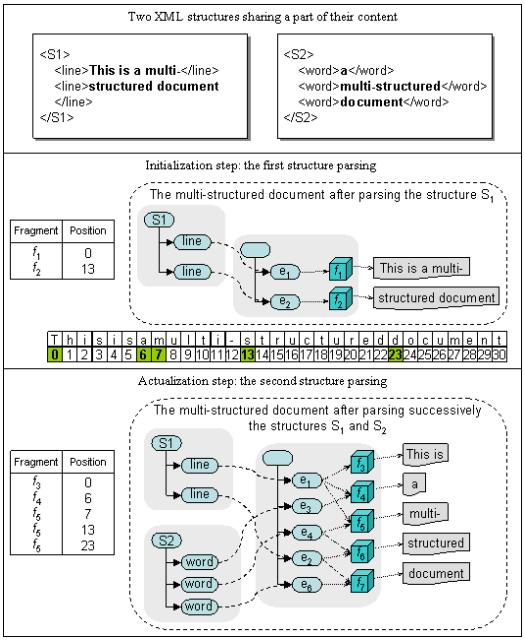
The following steps constitute a part of the *correspondence retrieval algorithm*:

When a new PCDATA is parsed do:

1. Pattern = _PCDATA
2. If CompactContent contains exactly one occurrence of Pattern then:

a. get the start position of Pattern in CompactContent,
b. in terms of this position and the length of _PCDATA get from FragMap all fragments which cover the PCDATA,
c. calculate the real positions (with whitespaces) at which fragments will be split,
d. make the needed fragments decompositions and actualize the base structure and all correspondences associated with these fragments,
e. create the new correspondence relation which links the element containing the PCDATA with the corresponding fragments composition in the base structure.

Figure 3: Illustration of the multi-structured document building precess



3. Else if CompactContent contains several occurrences of Pattern then:

a. Pattern = concatenate (Pattern, next _PCDATA in the parsed structure).
b. Go to 2.

4. …

We have removed the whitespaces from the text content to simplify the algorithm and to avoid possible errors.

In the lower frame of Figure 3, we can see the modifications affected to the multi-structured document in the middle frame, after parsing the second XML structure S_2. The fragment f_1 is split into three others fragments f_3, f_4 and f_5, and the fragment f_2 is split into f_6 and f_7. The document structure S_2 is generated and inserted in the multi-structured document. Finally the correspondence relations between S_2 and the updated base structure are established. The mapping table, which has been updated after parsing S_2, will be used if a third structure is presented to the MXP parser.

5 FUTURE WORK

Recently, we have proposed a multi-structured encoding format based on XML syntax. Now, the main perspective of this work is to define a multi-structured query language which may be exploited in several application areas. In addition to the legal texts of the CNAF-CNEDI, we are envisaging to test our system on a collection of structures describing manuscripts from several points of view. The web may be also an important application area of the multi-structuring. In fact, for example we can add structures including semantic information to existing web pages. With an appropriate query language (multi-structured query language), we can improve the information retrieval results.

6 REFERENCES

[1] Barnard, D., Burnard, L., Gaspart, J., Price, L., and Sperberg-McQueen, C.M. 1995. *Hierarchical Encoding of Text: Technical Problems and SGML Solutions*, Computers and the Humanities, Vol. 29, No. 3, pp. 211 – 231.

[2] C. Huitfeldt. *MECS - A Multi-Element Code System*. Working Papers from the Wittgenstein Archives at the University of Bergen, No 3, Version October 1998. http://helmer.hit.uib.no/claus/mecs/mecs.htm

[3] C. Huitfeldt, C. M. Sperberg-McQueen. *TexMECS: An experimental markup meta*-language *for complex documents*. Rev. 17 February 2001. http://www.hit.uib.no/claus/mlcd/papers/texmecs.html.

[4] J. Tennison, W. Piez. *The Layered Markup and Annotation Language (LMNL)*. In Extreme Markup Languages 2002. August 2002. http://www.extrememarkup.com/extreme/

[5] N. Chatti, S. Calabretto, J.M. Pinon. *Vers un environnement de gestion de documents à* structures *multiples*. 20ème JOURNEES BDA 2004, Montpellier. 19-22 October 2004, pp. 47-64

[6] R. Abascal, M. Beigbeder, A. Benel, S. Calabretto, B. Chabbat, P.A. Champin, N. Chatti, D. Jouve, Y. Prie, B. Rumpler, E. Thivant. *Documents à structures multiples*, SETIT 2004, Sousse Tunisie, Mars 2004

[7] R. Abascal, M. Beigbeder, A. Benel, S. Calabretto, B. Chabbat, P.A. Champin, N. Chatti, D. Jouve, Y. Prie, B. Rumpler, E. *Modéliser la structuration multiple des documents*. Actes de la Conférence H2PTM Hypertexte et Hypermédia Créer du sens à l'ère du numérique, Ed. Hermès, Paris, 24-26 September 2003, pp. 253-258

[8] Sperberg-McQueen, C.M.. and Burnard, L. (eds.) (2002). *TEI P4: Guidelines for Electronic Text Encoding and Interchange*. Text Encoding Initiative Consortium. XML Version: Oxford, Providence, Charlottesville, Bergen.

[9] *The XML Version of the TEI Guidelines : Multiple Hierarchies*. http://www.tei-c.org/P4X/NH.html