XML Representation of a Sheet Music for Chorus

Jounghoon Kim, Sunho Kim and Jinseok Chae
Dept. of Computer Science, Univ. of Inchon
177, Dohwa-Dong, Nam-Gu, Inchon 402-749, Korea
E-mail: {j_hon@isis, gksh@muse, jschae@lion}.inchon.ac.kr

Abstract

Extensible Markup Language (XML) is proposed to overcome drawbacks of HTML's simplicity and SGML's complexity. This paper describes a technique to represent a sheet music for chorus using XML. In this paper, we propose a new markup language, ScoreML (Score Markup Language) by defining a Document Type Definition (DTD) which has an enough expressive power to represent a sheet music for chorus, and implement a ScoreML browser using Java programming language to display a sheet music in general Web browser.

1 Introduction

HTML[1] is the most successful document format in history, but often it can seem to have either too many tags or not enough. If you just want to display text, there's nothing wrong with HTML, but for automated Web processing - enriching documents in a way that enables computer programs (like Web robots) to do something with them - you certainly need another markup language which has the extensibility.

Extensible Markup Language (XML)[2] is designed to do some jobs that HTML isn't built to handle but that really need doing. Extensibility is the reason for XML. XML promises to increase the benefits that can be derived from the wealth of information found today on IP networks around the world.

XML is a subset of Standard Generalized Markup Language (SGML)[3] that is optimized for delivery over the Web; it is defined by the World Wide Web Consortium (W3C)[4], ensuring that structured data will be uniform and independent of applications or vendors. This resulting interoperability kick-starts a new generation of Web applications. XML is valuable to the Internet because it provides interoperability using a flexible, open, standards-based format, with new ways of accessing legacy databases and delivering data to Web clients. Applications can be built more quickly, are easier to maintain, and can easily provide multiple views on the structured data.

XML is so extensible that even the music information can be represented in XML.

To represent the music information by the markup language, there was the Standard Music Description Language (SMDL)[5]. SMDL was defined by International Organizations for Standardization (ISO) for providing a comprehensive machine- and human-readable means for expressing music information for interchangeability. SMDL is a Hypermedia/ Time-based Structuring Language (HyTime)[6] application and an SGML application. However, SMDL is rarely used since it is too complex to represent the music information and implement the application software.

Another approach based on XML is The Connection Factory's effort[7]. The Connection Factory defined its own language specifically for a sheet music called MusicML (Music Markup Language). Based on the definition of MusicML they wrote a MusicML browser in Java to be able to display a sheet music. Their goal was not to define a sheet music specific language, but to explore the capabilities and limits of XML. It is the first attempt to represent a sheet music using XML, but it has an insufficient expressive power to represent a real sheet music since there is no consideration to represent ties, slurs, crescendo, decrescendo, and so forth.

This paper describes a technique to represent a real sheet music for chorus using XML. In this paper, we propose a new markup language, ScoreML (Score Markup Language) by defining a new Document Type Definition (DTD) which has an enough expressive power to represent a sheet music for chorus, and implement a Java applet to display a sheet music by upgrading a MusicML browser.

The other types of sheet music such as piano sonata, symphony, and so forth can be represented by defining the new DTDs, e.g., piano sonata DTD, symphony DTD, etc.

2 Overview of ScoreML

The ScoreML is a markup language originated from XML and developed to represent a sheet music. The flow diagram of ScoreML is shown in Fig. 1.

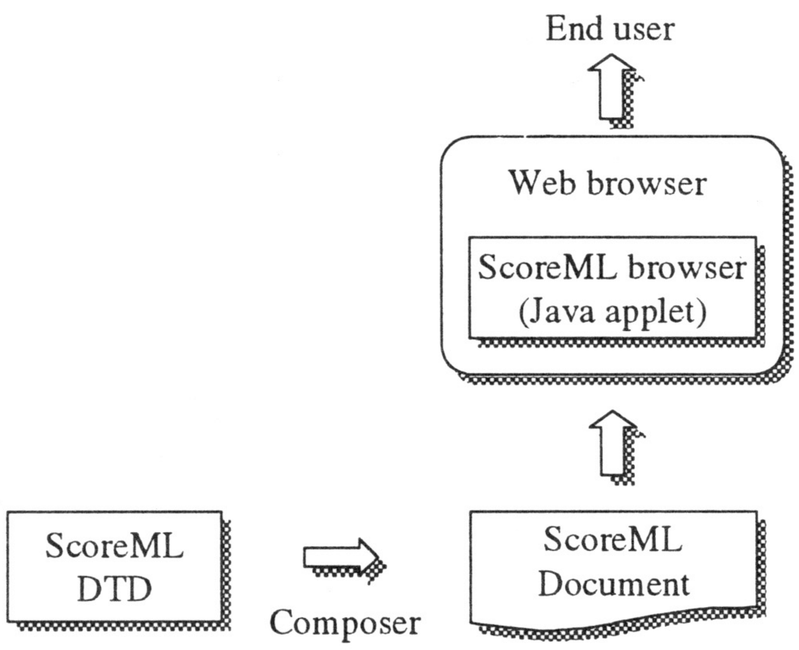

Fig. 1 Flow diagram of ScoreML

A composer makes ScoreML documents based on the ScoreML DTD by a text editor like Notepad or vi editor. At this time, we cannot afford to provide a visual ScoreML editor which has a visual editing capability. An end user views the ScoreML documents in its sheet music form through a Web browser like Netscape Communicator 4.0 or later. Since the ScoreML browser is a Java applet program, a Web browser downloads the program from the Web server which stores the ScoreML documents at the user's connection time.

3 DTD of a Sheet Music for chorus

In this section, we define a DTD of a sheet music for chorus, called a *chorus DTD*. The beginning of a chorus DTD is as follows:

```
<!ELEMENT sheet_music ( heading, body )>
```

The `sheet_music` element is composed of two parts: *heading* and *body*. The `heading` element has four sub-elements as follows:

```
<!ELEMENT heading
  ( title, lyric_writer?, composer? )>
<!ELEMENT title (#PCDATA)>
<!ELEMENT lyric_writer (#PCDATA)>
<!ELEMENT composer (#PCDATA)>
```

Example 1 shows the usage of `heading` element.

```
<heading>
 <title> Auld Lang Syne </title>
 <lyric_writer> Anonymous
  </lyric_writer>
 <composer> Folk song of Scotland
  </composer>
</heading>
```

Example 1 Usage of `heading` element

The `body` element consists of more than one `measure_group` element.

```
<!ELEMENT body ( system+ )>
```

The `system` element is composed of one `clef_meter_group` and more than one `measure` element.

```
<!ELEMENT system
  ( clef_meter_group, measure+ )>
```

Fig. 2 illustrates components of `system` element.

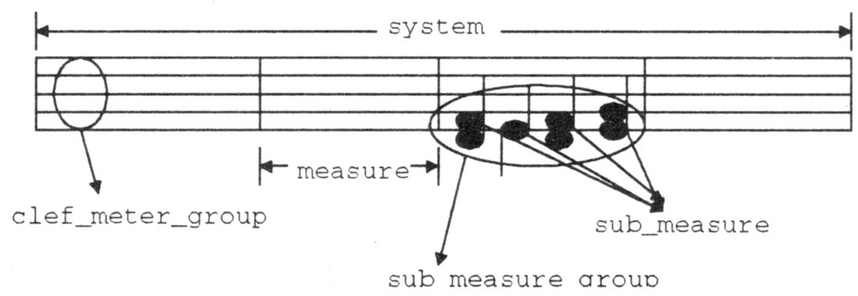

Fig. 2 Components of `system` element

Example 2 illustrates the part of ScoreML

document which describes the system element and Fig. 3 shows the screen display of Example 2 in ScoreML browser.

```
<system number_of_staff="one">
 <clef_meter_group>
  <clef_meter clef="treble" staff="one">
   <key_signature tonic="G"
     scale="major" />
   <meter_signature_list meter="4-4">
  </clef meter>
 </clef_meter_group>
 <measure>
  <sub_measure_group staff="one">
   <sub_measure voice="one">
    <note tone_name="D" />
   </sub_measure>
   <sub_measure voice="two">
    <note direction="down"
      tone_name="D" />
   </sub_measure>
  </sub_measure_group>
 </measure>
</system>
```

Example 2 Contents of system element

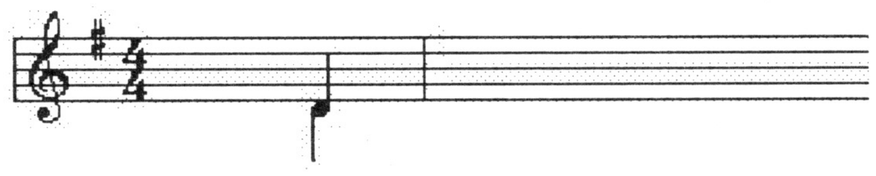

Fig. 3 Screen display of Example 2

The clef_meter_group element has more than one clef_meter element.

```
<!ELEMENT clef_meter_group
  ( clef_meter+ )>
```

The clef_meter element has one key_signature element and one meter_signature_list or meter_signature_other element.

```
<!ELEMENT clef_meter ( key_signature,
  (meter_signature_list |
  meter_signature_other)? ) >
```

The meter_signature_list element selects the frequently used meter signatures such as C, ₵, 2-4, 3-4, 4-4, 6-8.

The meter_signature_other element is used to denote arbitrary meter signature by assigning numerator and denominator separately.

The attributes of clef_meter element are clef and staff. The clef attribute has one of two values: treble or bass. The staff attribute denotes the staff number of the system. In the Example 2, we show the case that only one staff is used.

The measure element has optional clef_meter_group element and more than one sub_measure_group element. sub_measure_group element has more than one sub_measure

```
<!ELEMENT measure (clef_meter_group?,
  sub_measure_group+)>
<!ELEMENT sub_measure_group
  (sub_measure+)>
```

The note element has attributes such as tone_name, length, dot, direction, and so forth. Table 1 shows the mapping between length attribute and note.

Table 1 Length attribute and note

Value	Note
whole	𝅝
half	𝅗𝅥
quarter	♩
eighth	♪
sixteenth	𝅘𝅥𝅯

The rest element has length and dot attributes whose meaning is same as the note element.

4 ScoreML browser

Fig. 4 shows the system architecture of ScoreML browser. In the figure, rectangles indicate processors and arrows flow of processing.

For a given ScoreML document, the XML parser performs parsing and generates a XML parse tree. In this paper, we use Microsoft XML parser which was written in Java programming language.

The Parse tree navigator traverses the XML parse tree and transfers each element to the

Painter & drawer. The Score drawer draws the sheet music on the Web browser like Netscape Communicator.

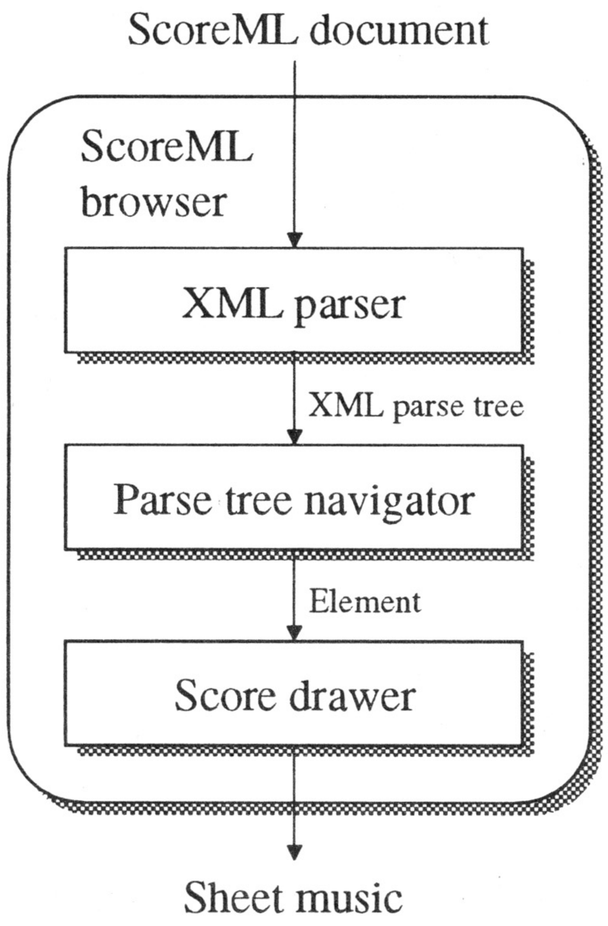

Fig. 4 System architecture of ScoreML browser

Fig. 5 shows the screen display of a real sheet music of Auld Lang Syne written in ScoreML by Java applet in Netscape Communicator 4.5. The URL of this site is http://muse.inchon.ac.kr.

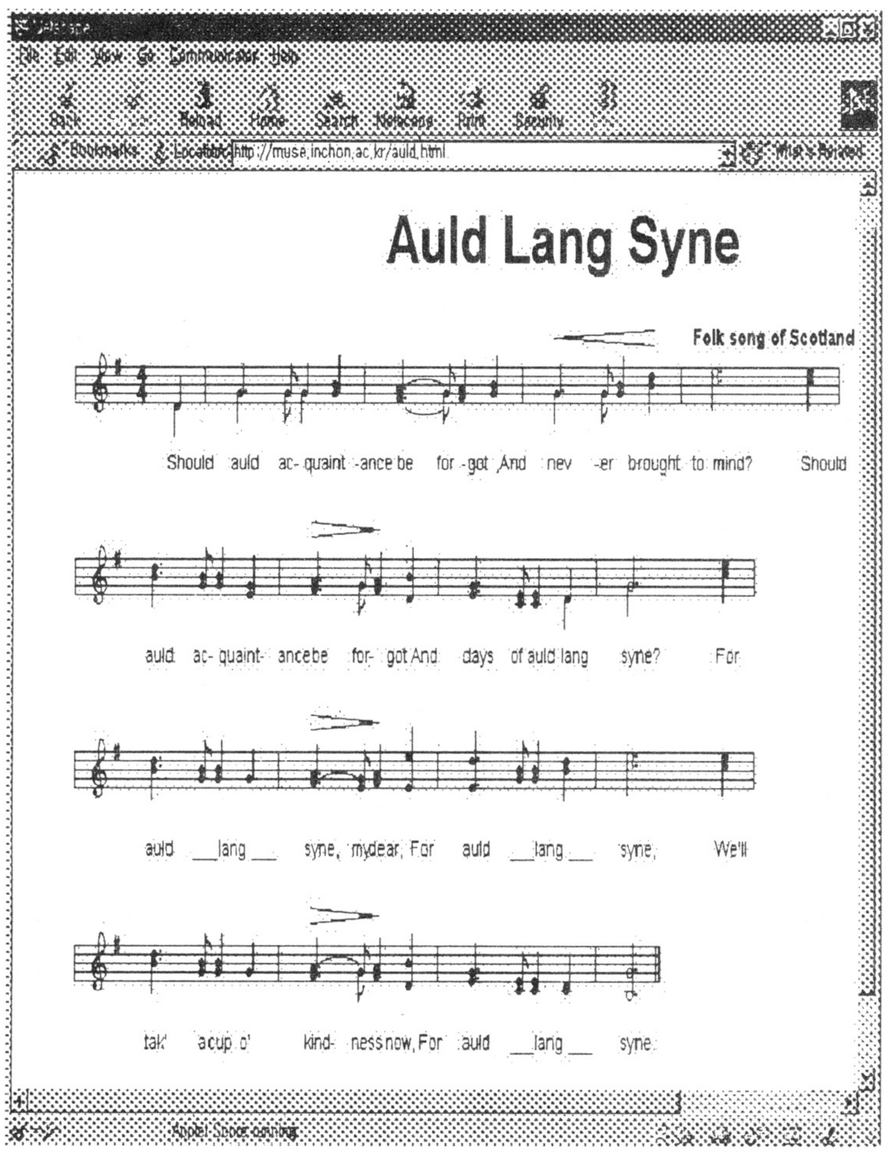

Fig. 5 Screen display of Auld Lang Syne

5 Conclusion

In this paper, we define a DTD to represent a sheet music for chorus, called a chorus DTD. By using this DTD, a new markup language, ScoreML (Score Markup Language), is proposed and it is shown that the ScoreML has an enough expressive power to represent a sheet music for chorus. The ScoreML browser is implemented by using Java programming language and it is compiled into Java applet to display a sheet music in general Web browser like Netscape Communicator. End users can view a sheet music written in ScoreML by downloading the Java applet into their Web browsers.

Since it is inconvenience to input the music information, much efforts should be concentrated on the development of the visual editing tool such as ENCORE and SCORE to edit the music information easily. For utilizing the structured querying facility of XML document, more research is necessary to store and retrieve the ScoreML documents by using object-oriented databases or object-relational databases.

The current prototype system has been implemented on a Sun Workstation by using Java Development Kit 1.1 and Micorsoft XML parser in Java called MSXML.

References

[1] Dave Raggett, Arnaud Le Hors, Ian Jacobs, HTML 4.0 Specification, REC-html40-19980424, http://www.w3.org/TR/REC-html40, 1998.

[2] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Extensible Markup Language (XML) 1.0, REC-xml-19980210, http://www.w3.org/TR/1998/REC-xml-19980210, 1998.

[3] ISO 8879: 1986, Information processing - Text and office systems - Standard Generalized Markup Language (SGML), 1986.

[4] http://www.w3.org

[5] ISO/IEC 10743: 1995, SMDL - Standard Music Description Language, 1995.

[6] ISO/IEC 10744: 1992, Information technology - Hypermedia/Time-based Structuring Language (HyTime), 1992.

[7] http://195.108.47.160/3.0/musicml/index.html