A Generation Method of MPEG-4 3D Scene for Mobile Environments

Hee-Sun Kim and Ilgun Ko
Department of Multimedia Engineering, Andong National University
388 Songchun-dong, Andong-si, Gyungbook 760-749 Korea
hskim@andong.ac.kr, ilgunko@gmail.com

Abstract

MPEG-4 LASeR as a standard was established for expressing rich media within a mobile environment. It is difficult to describe 3D nodes of MPEG-4BIFS, however, because LASeR can view only 2D graphics. To play MPEG-4 3D scenes within a mobile environment, this study demonstrated a method of creating 3D nodes with LASeR. First, the values of the attributes of BIFS 3D graphic nodes were calculated. Their 3D coordinates were converted into 2D coordinates, and then coordinate transformation was applied. At the close of the play, 3D scenes were created through LASeR 2D graphic nodes using 2D coordinate values. This made it possible to play MPEG-4 3D geometry objects within a mobile environment.

Keywords: *MPEG-4 Scene, LASeR, 3D Scene generation*

1. Introduction

MPEG-4 defines BIFS, a language that structures audio/visual objects and their scenes, for two-way conversation-type services[1]. BIFS is a major rich media format that has been selected as the standard for conversation contents and data broadcasting through DMB. BIFS can express rich media well through various nodes and abundant user interaction, but it is not appropriate to the mobile environment that has limited performance. With the fundamental start of digital broadcasting, it has become very important to service rich media in various environments, especially mobile devices. Therefore, there is a need to study how to service rich media such as BIFS in the mobile environment.

Therefore, this study proposes a method of converting the existing MPEG-4 BIFS to the MPEG-4 LASeR (Lightweight Application Scene Representation: ISO/IEC 14496-29)[2,3] format that is appropriate to the mobile environment, to service rich media in a mobile environment. LASeR uses the 2D graphic language of W3C, SVG[4], to express graphic animation and handle light resources, and is structured by limited scene technology nodes that are appropriate to the mobile environment.

Studies related to MPEG-4 scene conversion focused on creating MPEG-4 BIFS as XMT[5] or converting XMT to VRML or SMIL. One study on converting XMT to BIFS focused on the authoring tool developed by IBM[6]. This is a 2D scene writing tool that creates XMT alpha and omega, and includes functions that create XMT and convert XMT-A to BIFS. There has been a study on converting MPEG-4 contents to LASeR to play them on mobile devices[7]. This study converted only 2D nodes that can convert

BIFS to LASeR. This is because LASeR uses SVG, the 2D graphic language, and it is possible to convert 2D-graphics-related nodes of BIFS to LASeR, but 3D-related nodes are not supported and are difficult to convert. With the demand for contents such as 3D navigation or 3D games growing in the mobile environment, however, the expression of LASeR-based 3D nodes has become important.

Therefore, this study presents a method of converting 3D nodes to LASeR. This study presented a system of converting XMT files, the textual format of BIFS, to LASeR, and designed and implemented the system. The conversion system parses the XMT files that the user entered to create a DOM tree on the memory. 2D nodes with functions and characteristics similar to those of XMT and LASeR are converted using the XSLT[8] style sheet. 3D geometrical nodes, however, must go through a process of converting the node value to a value that can be handled through a LASeR node. Next, a substitute node that can be mapped as a LASeR standard SVG node on a DOM tree is added, and the value is revised. This conversion method can service rich media, including 3D nodes, through mobile devices.

The second part of this study comparatively analyzes MPEG-4 BIFS and LASeR, and the third part explains the method of converting BIFS to LASeR. Part four shows the results of the materialization of the system that converted the BIFS text format XMT to LASeR, and part five presents the conclusion.

2. Comparison of BIFS and LASeR

The standard plan of BIFS is limited because it is a PC-based standard and demands many resources for the mobile environment. MBEG-4 is therefore an alternative to BIFS, and MOEG-4 part20 LASeR was set as the standard. The characteristics of BIFS and LASeR are shown in Table 1.

BIFS uses VRML as its base technology and can express 2D and 3D scenes, but the standard is complex. On the other hand, LASeR is a base technology that uses SVG and through which only 2D graphic expressions are possible. Moreover, it has a simple standard. In addition, it includes technology and videos where the document itself is not downloaded but scene renewal and conversion are made into binary contents and streamed, and technology that expresses freely together audio and graphic data and logically streams them in multiple transmissions. Therefore, LASeR enables dynamic rich media services that support abundant scene expressions and efficient compression in a mobile environment.

Table 1. Comparison of the characteristics of BIFS and LASeR

	BIFS	LASeR
Standard	MPEG-4 Part1 System	MPEG-4 Part20 LASeR
Device	PC	Mobile
Base Technology	VRML	SVG
Scene Presentation	2D/3D	2D
Binary Encoding	Support	Support
Interactive	Support	Support

Streaming	Support	Support

For scene description, MPEG-4 provides the text format for both a binary format and contents interoperability, which is why MPEG-4 provides XMT (eXtensible MPEG-4 Textual format) as a standard. Table 2 shows the file structure of XMT-A.

Table 2. Structure of XMT-A

```
<?xml version="1.0" encoding="UTF-8" ?>
<XMT-A xmlns="urn:mpeg:mpeg4:xtma:schema:2002"
xmlns:xsi="http://www.w3.org/2001/xMLSchema-instance"
xsi:schemaLocation="urn:mpeg:mpeg4:xmta:schema:2002 xmt-a.xsd">
<Header>
<InitialObjectDescriptor obejectDescriptorID="od1" binaryID="1" >
        Scene description
        -> Size
        -> Streams
<DEscr>
</DEscr>
</InitialObjectDescriptor>
</Header>
<Body>
        Scene Content
        -> Nodes
        -> Routes
</Body>
</XMT-A>
```

LASeR also has the LASeR-XLM format, a text type of XML format. With the LASeR encoder, LASeR-XLM is encoded in a binary stream. Table 3 shows the file structure of LASeR-XML

Table 3. Structure of LASeR-XML

```
<saf:SAFSession>
<saf:sceneHeader>
<svg:LASeRHeader/>
</saf:sceneHeader>
<saf:sceneUnit>
<lsru:NewScene>
<svg:svg>
</svg:svg>
</lsru:NewScene>
</saf:SceneUnit>
```

```
<saf:SAFSession/>
```

3. How to create 3D scenes using LASeR nodes

3.1 Conversion of BIFS to LASeR

XSLT is used to convert XMT files, the text format of BIFS, to LASeR. XSLT is a markup language that explains the process of converting XML-based documents to XML documents with different structures. The node conversion rules for the conversion of BIFS to LASeR are created as an XSL file for the conversion of BIFS documents to LASeR documents. Table 4 shows the nodes that can be converted immediately through one-to-one counteraction because they have similar functions and characteristics.

Table 4. Nodes that can be converted one-to-one from BIFS to LASeR.

BIFS nodes	LASeR nodes
Geometry Nodes(Rectangle, Curve2D, Circle, shape, appearance …)	svg : rect , svg : circle svg : path
Text, FontStyle	svg : text , svg : tspan
Hyperlink nodes(inline, anchor)	svg : a , svg : foreightObject
Transform2D node	svg : g
OrderedGroup, Layer2D	svg : g
WorldInfo node	svg : svg
ImageTexture, PixelTexture MovieTexture nodes	svg : image, svg : video
Interpolator, TimeSensor nodes	lsr : animate
Touch Sensor	lsr : trigger node
Conditional	lsr : script
AudioClip, AudioSource nodes	svg : audio
Switch nodes	svg : g
Valuator node	Unmapped
PointSet2D, CompositeTexture2D TextureCoordinate nodes	Unmapped
TempCap, QuantizationParameter	Unmapped

During the conversion of XSLT from BIFS to LASeR, since the coordinate systems of BIFS and LASeR differ, the conversion of the coordinate values must be considered. The BIFS 2D node coordinates values are accordingly changed to the LASeR coordinates. It is difficult to convert 3D nodes to XSLT through LASeR, though. Table 3 shows BIFS 3D nodes that cannot easily be converted to LASeR. SVG, the scene technology language of LASeR, cannot express 3D coordinate values because it shows 2D graphics. The 3D geometrical objects that are used in BIFS such as boxes, cylinders, cones, and spheres, and parent nodes such as transform and material nodes, are difficult

to convert. Therefore, this study presented a method of converting the nodes that are difficult to convert, as shown in Table 5, and materialized the conversions.

Table 5. Nodes that are not easily converted from BIFS to LASeR.

Class	Node Name	2/3D	Types/Function
General-use	Coordinate	3D	Attribute
General-use	Material	3D	Attribute
General-use	Transform	3D	Group
Geometry	Box	3D	Geometry
Geometry	Sphere	3D	Geometry
Geometry	Cylinder	3D	Geometry
Geometry	Cone	3D	Geometry
Geometry	IndexedLineSet	3D	Geometry
Geometry	IndexedFaceSet	3D	Geometry

3.2 Method of converting BIFS 3D graphic nodes to LASeR

This study separately investigated BIFS 3D nodes and proposed a method of adding nodes to allow their handling as LASeR nodes, so as to change the BIFS 3D nodes to LASeR nodes. Figure 1 shows how BIFS 3D geometric nodes can be expressed as LASeR nodes.

First, as shown in Figure 1 (1), the XMT-A file is parsed and a DOM-Tree is created on the memory. This DOM-Tree, which is shown in Figure 1 (2), finds 3D nodes and extracts attribute values. For example, the box object of XMT has size attribute, and the attribute is used to determine the reference coordinates. The value extracted in Figure 1 (3) is used to create reference coordinates that structure a 3D geometric object. Boxes have size attribute and create reference coordinates that show eight points of the box object, with the starting point as the standard. The coordinates, rotation, and scaling of the XMT nodes are stored in the parent node that is the transform node, so that as shown in Figure 1 (4), the attributes of the transform node are extracted and the coordinates of the object are rotated, scaled, and calculated. For example, the location movement information, the rotate information, and the scale information of the XMT transform node are used to move, rotate, and scaled the reference coordinates in Figure 1 (3) and to find the location where the object will be expressed. Next, the 3D coordinates are projected and converted into 2D coordinates, as shown in Figure 1 (5), and the 2D coordinates are again converted.

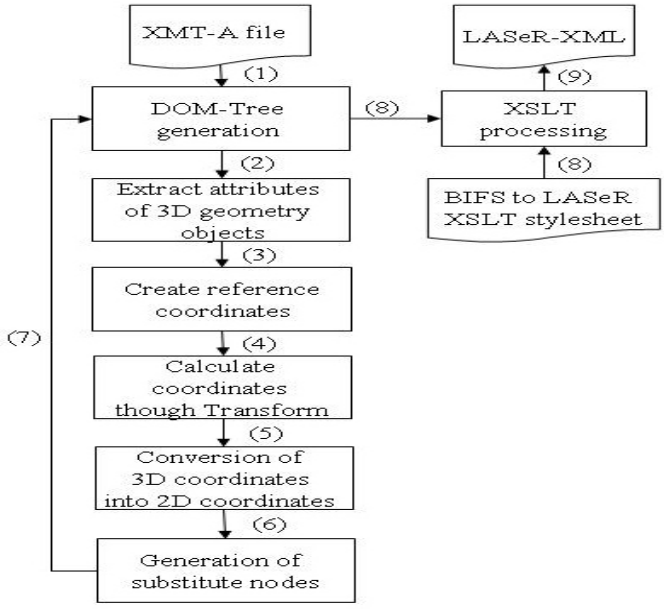

Figure. 1 Method of converting BIFS 2D geometric nodes to LASeR nodes

BIFS defines the center of the scene as the starting point, but LASeR defines the top left as the starting point. Therefore, the coordinates must be converted. Figures 1 (6) and (7) add a substitute node to express the 3D node of BIFS as the LASeR expression on the XMT DOM-tree. Instead of the 3D geometric objects of BIFS, i.e., the box, sphere, cylinder, and cone, alternative nodes with 2D coordinate values, such as a 2D box, a 2D sphere, a 2D cylinder, and a 2D cone are added. In the case of the indexed face set and the indexed line set, the coordinate-related values are converted into 2D coordinate values. Next, in Figures 1 (8) and (9), the XSLT conversion is performed to convert XMT to LASeR.

The box is expressed as svg:polygon; the cone, as svg:circle and svg:polyline; the sphere, as svg:ellipse; and the cylinder, as svg:polygon and svg:ellipse. The indexed line and the indexed face are calculated when there is a transformation, and after the 3D coordinates are converted into 2D, the values are changed and the indexed line 2S and the indexed face set 2D are mapped in the same way on LASeR.

As the coordinate system of LASeR is distinct from that of BIFS, the coordinate values should be physically altered and converted into LASeR nodes using Formula (1).

$$Abifs(x1,y1) = Blaser(x1 + width/2, y1 + height/2) \qquad (1)$$

BIFS scene description uses the 3D vector graphic language VRM, but LASeR applies the 2D vector graphics language SVG. This means LASeR-based one-to-one mapping is impossible for graphic nodes that contain attributes with 3D coordinate values. Therefore, to express BIFS 3D graphic nodes using LASeR nodes, the coordinates were transformed after the 3D coordinates were converted into 2D coordinates.

3.3 Reference coordinates of BIFS 3D geometry and LASeR expression

<Box>, <Cone>, <Cylinder>, and <Sphere>: BIFS geometry object nodes that all have the attribute values of 'size', 'bottomRadius', 'height', and 'topRadius'. The node compositions of <Box>, <Cone>, <Cylinder>, and <Sphere> are shown in Figure 2.

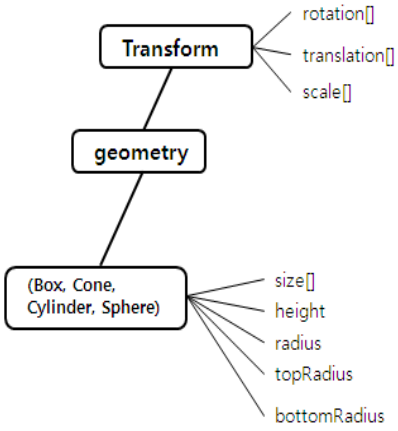

Figure. 2 Node compositions of <Box>, <Cone>, <Cylinder>, and <Sphere>

For example, the <Box> node, as a size attribute, shows the 'width', 'length', and 'height' of the Box. The higher-level node <Transform> describes the information, including 'movement', 'rotation', and 'enlargement' of the Box. LASeR cannot express <Box> with these 'size' attributes, however, which means LASeR's 2D geometry objects, including its 'circle', 'polygon', and 'polyline', must be utilized to show 3D objects. In detail, it is feasible to express BIFS's <Box> node with LASeR's <polygon> node if the coordinate values are calculated by figuring out the vertex coordinates for the <Box> node and applying the data such as the 'coordinates movement', 'rotation', and 'enlargement' to the <Transform> node.

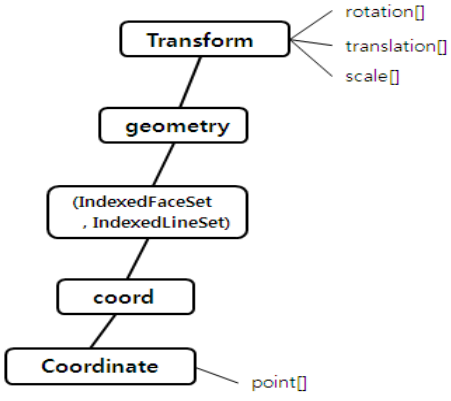

Figure. 3 Node compositions of IndexedFaceSet and IndexedLineSet

Figure 3 shows the node composition of IndexedFaceSet and IndexedLineSet among various BIFS 3D graphic nodes. To describe BIFS 3D graphic nodes such as IndexedFaceSet and IndexedLineSet with LASeR, it is essential to calculate the x, y, and z coordinate values of the <Coordinate> node, a lower-level node; the attribute values of the <Transform> node, a higher-level node; and the coordinate values of graphic objects. Then the projection calculation must be used to express the figured 3D coordinate values A (x, y, z) into the 2D coordinate values A' (x', y') on LASeR's 2D coordinate system. For this process, BIFS's text format 'XMT-A' file should be 'parsed' within the 'transformation system', and DOM-Tree should be created and implemented in memory. The IndexedFaceSet node processes the higher-level <Transform/> node and the projection calculation for the 2D coordinates. Also, an alternative <altFaceSet/> node with a 'point' attribute as the calculated coordinate value should be added to the memory's DOM-Tree. In an XLST processor, XSL transformation rules are used for 'mapping' with the LASeR scene description node ‘svg: polygon'. This is how the <Transform/>, <geometry/>, <coord/>, <Coordinate/>, <CoordinateInterpolator/>, <IndexedLineSet/>, and <IndexedFaceSet/> nodes are expressed as LASeR's svg: g, polyline, polygon, and animate nodes.

3.4 Generation method of 2D coordinates

After figuring out the coordinate values of the vertexes using the 3D Geometry node attributes, the 'Transform' node, a higher-level node of the 'Geometry' node, is used to calculate the actual coordinate values. The actual coordinates of each vertex are calculated by referring to the 'rotation', 'movement', and 'enlargement' attribute values of 'Transform', in that order.

Referring to the attribute values of the 'Transform' node that is a higher-level node of XMT-A's 3D geometry node, the attribute values’ 3D coordinate system is converted into a 2D coordinate system. The following 'coordinate transformation equation' is used to convert 3D coordinates into 2D coordinates. In detail, any 3D coordinate point A (x,y,z) is expressed with A' (x',y') on the display after using the following formula. The distance on the 2D display is the focal length, which is the distance between the screen and the user’s eye, as well as the ‘zpos’, which is the distance between a vanishing point and the actual 3D coordinate point projected parallel to the screen. The calculation of this distance is as follows:

$$x' = \text{distance} * z/x \quad (2)$$

$$y' = \text{distance} * z/y \quad (3)$$

$$\text{distance} = fl\ (\ fl + zpos) \quad (4)$$

4. Implementation of the Conversion System

The conversion system was implemented in a Windows XP environment to Java using the Eclipse 3.4 development platform. Using the conversion system presented in this study, XMT-A files were converted to LASeR. Figure 4 shows the results of the use of the mp4Box of an XMT-A file GPAC that includes 3D geometrical objects to create mp4 files, and the results of playing them on IMI-3D, the IBM MPEG-4 contents player. Figure 2 shows the 3D nodes of BIFS--the box, cylinder, sphere, cone, and indexed face set. Some of the XMT-A files in Figure 4 are shown in Table 6.

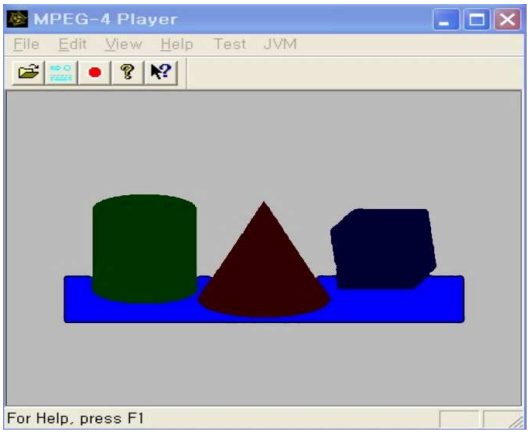

Figure. 4 Results of playing XMT-A, which includes the 3D nodes

Table 6. Parts of the XMT-A file in Figure 4.

```
<?xml version="1.0" encoding="UTF-8"?>
<XMT-A xmlns="urn:mpeg:mpeg4:xmta:schema:2002"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:mpeg:mpeg4:xmta:schema:2002 xmt-a.xsd">
 <Header>
    ... omission ...
 </Header>
 <Body>
 <Replace>
  <Scene>
 <Group>
    <children>
    <Transform rotation="1 0 0 0.349" translation="-180 0 0" scale="1 1 1">
     <children>
      <Shape>
       <appearance>
        <Appearance>
         <material>
          <Material diffuseColor="0 1 0"/>
         </material>
        </Appearance>
       </appearance>
       <geometry>
        <Cylinder height="160" radius="80"/>
       </geometry>
  ... omission ...
 </Body>
 </XMT-A>
```

The XMT-A file in Table 6 was converted using the conversion system suggested in this study. Table 7 shows the LASeR-XML file, which is the result of the conversion of Table 6. The 3D nodes of XMT in Table 6, i.e., the box, cylinder, sphere, cone, and indexed face set, are expressed in Table 7 as g and the polygon and ellipse of LASeR.

Table 7. LASeR-XML file that is the conversion result of Table 6.

```
<?xml version="1.0" encoding="UTF-8"?>
<saf:SAFSession xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:lsr="urn:
mpeg:mpeg4:LASeR:2005" xmlns:saf="urn:mpeg:mpeg4:SAF:2005"  xmlns:ev=
"http://www.w3.org/2001/xml-events"  xmlns="http://www.w3.org/2000/svg">
<saf:sceneHeader>
<lsr:LASeRHeader profile="full" colorComponentBits="8"/>
</saf:sceneHeader>
<saf:sceneUnit>
<lsr:NewScene>
<svg id="root-node" width="480" height="320" viewport-fill="rgb(0,0,0)">
<g>
<polygon points="96.0,221.7 384.0,221.7 384.0,174.7 96.0,174.7 " fill="rgb(0, 0, 255)"/>
<polygon points="294.2,190.1 315.6,166.5 368.2,166.5 346.8,190.0" fill="rgb(0, 0, 55)"/>
<polygon points="284.6,137.5 306.0,114.0 358.6,114.0 337.2,137.5" fill="rgb(0, 0, 55)"/>
<polygon points="346.8,190.0 368.2,166.5 358.6,114.0 337.2,137.5" fill="rgb(0, 0, 55)"/>
<polygon points="294.2,190.1 346.8,190.0 337.2,137.5 284.6,137.5" fill="rgb(0, 0, 55)"/>
<polygon points="315.6,166.5 294.2,190.1 284.6,137.5 306.0,114.0" fill="rgb(0, 0, 55)"/>
<polygon points="315.6,166.5 368.2,166.5 358.6,114.0 306.0,114.0" fill="rgb(0, 0, 55)"/>
<ellipse cx="240.0" cy="195.8" rx="48.0" ry="19.6" fill="rgb(55, 0, 0)"/>
... omission ...
</saf:SAFSession>
```

Figure 5 shows the results of the conversion of XMT to LASeR in Table 7, and of playing it on the GPAC Osmo4 multimedia player. The results confirmed that the XMT file that included 4F nodes used LASeR, which supports only 2D for the same method of expression.

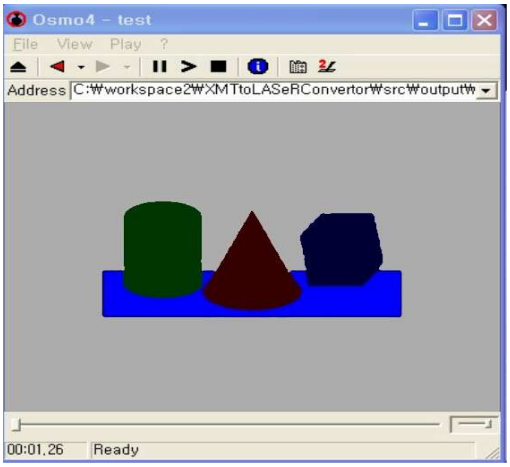

Figure. 5 Example of playing Table 5

5. Conclusion

This study proposed a method of converting 3D-related graphic nodes of BIFS to LASeR for the inter-operation of MPEG-4 contents in a mobile environment, and materialized a conversion system using the proposed method. The problem of 3D-graphics-related node conversion, particularly the difficulty of converting BIFS to LASeR, was solved. The 3D node of BIFS, the transform, and the nodes such as the

coordinate, material, indexed line set, indexed face set, box, cylinder, sphere, and cone, could be expressed as a LASeR g, polyline, polygon, circle, and ellipse. Through this conversion method, rich media, including 3D nodes, can be serviced from a mobile device.

References

[1] Understanding MPEG-4 : Technologies, Advantages, and Market(An MPEGIf White Paper), The MPEG Industry Forum.

[2] ISO/IEC 14496-20:2006(E) Information technology - Coding of audio-visual objects - Part 20: Lightweight Application Scene Representation(LASeR) and Simple Aggregation Format(SAF).

[3] Jean-Claude Duford, Olivier Avaro, Cyril Concolate "LASeR : the MPEG Standard for Rich Media Servicse", IEEE Multimedia.

[4] W3C, Scalable Vector Graphics (SVG) Tiny 1.2 Specification [Last Call], http://www.w3.org/TR/2005/WD-SVGMobile12-20050413/

[5] KIM. M, WOOD. S, "XMT: MPEG-4 Textual Format for Cross- Standard. Interoperability", Overview of XMT. IBM Research, 2004.

[6] IBM Toolkit for MPEG-4, http://www .alphaworks.ibm.com/tech/ tk4mpeg4

[7] Qonita M. Shahab, "A Study on Transcoding of MPEG-4 BIFS scenes to MPEG-4 LASeR Scenes in MPEG-21 Digital Item Adaptation Framework", School of Engineering Information Center Univ, 2006.

[8] Eric M. Burke, Java and XSLT, O'Relly, 2001.