2-2002

SMIL vs MPEG-4 BIFS

Lai-Tee CHEOK
Singapore Management University, LAITEECHEOK@smu.edu.sg

Alexandros Eleftheriadis

Citation

CHEOK, Lai-Tee and Eleftheriadis, Alexandros. SMIL vs MPEG-4 BIFS. (2002). *Columbia University Department of Electrical Engineering Technical Report*.
Available at: https://ink.library.smu.edu.sg/sis_research/1916

DEPARTMENT OF ELECTRICAL ENGINEERING
TECHNICAL REPORT

SMIL VS MPEG-4 BIFS

Lai-Tee Cheok, Alexandros Eleftheriadis

CU/EE/TR 2002-02-111

Columbia University
Department of Electrical Engineering
500 W. 120th Street, MC 4712
New York, NY 10027

http://www.ee.columbia.edu

The technical reports in this series are considered to be semi-formal.
The ideas expressed are solely those of the authors, and questions about the content should be directed to them.

SMIL vs MPEG-4 BIFS

Lai-Tee Cheok*, Alexandros Eleftheriadis

February 10, 2002

Department of Electrical Engineering
Columbia University
500 West 120 Str., MC 4712
New York, NY 10027
Tel: 212-854-0605
laitee,eleft@ee.columbia.edu

EDICS: 8-STDS

Abstract

[1]We present the results of a comparative analysis between the Synchronized Multimedia Integration Language (SMIL) and MPEG-4 BInary Format for Scenes (BIFS). SMIL is a language developed by the W3C consortium for expressing media synchronization among objects of various media types. MPEG-4 BIFS is the scene description scheme of MPEG-4, an international standard for communicating interactive audiovisual scenes. They are both facilities for representing and synchronizing multimedia content, and have a wide range of support for interactivity, animation and object composition features, etc. We compare their scope and purposes, the level of support for the multimedia features and investigate the degree of complexity of each of their representation formats. This comparison study is primarily based on SMIL 2.0 and version 3 of BIFS. The analysis shows that although MPEG-4 has better support for 3D features, on the things that both can do, SMIL appears to be better and easier to use. SMIL also provides better timing, animation controls, more transition effects, and supports keyboard events which are missing in MPEG-4. In addition, although MPEG-4 has been defined with the aim of standardizing many aspects of a multimedia streaming application, there are no well-defined interfaces in place for its streaming mechanism.

[1]Work supported in part by IBM Research and Sun Microsystems Computer Corporation.

Keywords: MPEG, SMIL, synchronization, scene description, authoring, interactivity, multimedia applications.

1 Introduction

Rich multimedia presentations are becoming more and more common on the Web. They include newscasts, educational material, entertainment, etc. The SMIL language can be used to create dynamic multimedia presentations by synchronizing the various media elements in time and space. SMIL is designed based on XML and its synchronized hypermedia can be created without using any sophisticated authoring tool. SMIL 1.0 [1] is the first version of the SMIL language developed by the W3C consortium to describe the temporal behavior and layout of a 2D presentation, as well as to associate hyperlinks with media objects. Among the various commercially available SMIL players are Apple QuickTime 4.1 [2], Internet Explorer 6.0 [3] and NIST S2M2 Player [4], which is written using Sun's JDK 1.1 and Java Media Framework (JMF 1.0). Several authoring tools are also available on both PC and Macintosh platforms, such as the Oratrix's GRiNS authoring tool [5] and RealNetworks' RealSlideshow 2.0 [6]. Other players and authoring tools are also available from the web site of the W3C World Wide Web Consortium [7]. To extend the SMIL functionality such that a subset of which can be used for a particular multimedia authoring environment in mind (thus forming SMIL "profiles"), the Synchronized Multimedia Working Group (SYMM) was established to define SMIL 2.0 [8] which partitions SMIL functionality into sets of reusable modules.

MPEG-4 is an ISO/IEC standard developed by the MPEG (Movie Pictures Expert Group) that aims to address the need for emerging multimedia applications in terms of interactivity, content description, scene description and programmability [9]. In addition to the Internet, the standard is also designed for low bit-rate communication devices that are usually wireless [10]. Furthermore, it encompasses a world of 2D and 3D objects, as well as those from both natural and synthetic (computer generated) sources. The way these objects are composed to form an MPEG-4 scene is dependent on the scene description information which is coded in a binary format known as BIFS (BInary Format for Scenes). Example MPEG-4 applications include Internet and Intranet video, wireless video, interactive home shopping, virtual reality games, etc. A reference implementation of an MPEG-4 player is also available [11]. There are also commercial players, including EnvivioTV from Envivio [12] and WebCine [13] from Philips. Additional information on commercial products can be found at the MPEG-4 Industry Forum [14].

We conducted a comparative analysis of SMIL and MPEG-4 BIFS by comparing their various multimedia features, their modules and functionalities, level of abstraction and complexity of their scene structure as well as their scopes and purposes. We then describe a new textual format called XMT [15] that aims to provide a higher level of abstraction for the MPEG-4 features. The rest of the paper is structured as follows:we provide a technical overview of both standards in Section 2 and compare their architectures and multimedia features in Sections 3 and 4 respectively. We then devote Section 5 to describe XMT and discuss if and how it manages to combine the best features of both SMIL and MPEG-4. We finally conclude our paper in Section 6.

2 Technical Overview

2.1 MPEG-4 and BIFS

The MPEG-4 standard is designed “to facilitate interoperability of multimedia terminals, products or services” [10]. The standard thus includes audio and video coding, coding of text/graphics and synthetic objects, systems multiplexing/demultiplexing, scene composition and interactivity [16]. Figure 1 shows a high level view of an MPEG-4 terminal [19, 20, 21]. Individually coded audiovisual objects are multiplexed from a storage or transmission medium, together with scene description information that describes how these objects should be combined in space and time. The user can interact with the composed and rendered scene either locally or with the remote source via an upstream channel. MPEG-4 Systems [22] defines the overall architecture of MPEG-4 and provides tools to combine elements defined in other parts of the specification. The overall architecture of an MPEG-4 terminal is depicted in Figure 2. Delivery Multimedia Integration Framework (DMIF) [23] was defined as a set of abstract procedures for initializing an MPEG-4 session and providing access to the individual elementary streams which carry both media and meta data and are encapsulated in SL (Sync Layer) packets that include timing and fragmentation information for clock recovery and synchronization in MPEG-4. Streams arriving at a terminal are sent to their respective decoders for processing. The type of content conveyed in each stream is identified by the object descriptors [24]. The compositor uses the scene description information, together with decoded audio-visual object data to render the final scene.

The BInary Format for Scenes (BIFS) [25] is the MPEG-4 Systems facility that defines the composition and the interactive behaviors of MPEG-4 objects. MPEG-4 scene has a structure partly inherited from the Virtual Reality Modeling Language (VRML) [26], with additional mechanisms: compression, data streaming and scene updates. MPEG-4 uses the VRML nodes, fields and events to represent the scene elements. Events are propagated using the VRML ROUTEs. An example of a tree structure is shown in Figure 3. The scene content and BIFS are typically streamed from a server. This contrasts with the VRML model where content has to be completely downloaded before it is rendered. Once the scene is in place, the server can further modify it using a scene update mechanism, a.k.a BIFS Command protocol or BIFS-Update. This mechanism allows a scene to be remotely manipulated, and portions of the scene to be progressively streamed in order to reduce bandwidth requirements.

2.2 SMIL 2.0

The W3C established the first working group in March 1997 to focus on the design of the SMIL 1.0 specification. SMIL is written as an XML-based language which is self-describing and familiar to HTML users. It allows integrating a set of independent media objects into a synchronized multimedia presentation. SMIL 2.0 partitions the SMIL features into sets of markup modules, and provides a framework for adding these modules into other XML document formats, including the Scalable Vector Graphics (SVG)[27]. Besides modularity, SMIL 2.0 is designed to also improve several features in SMIL 1.0. Each module is uniquely indentified by an XML Namespace Identifier. The modules are given below (readers can refer to SMIL 2.0 specification for further details): (1)the Animation Module (incorporates animation using both timing and animation elements and attributes); (2)the Content Control Module (provides runtime content choices and optimized content delivery); (3)the Layout Module (allows positioning of objects onto the rendering surface; can use the Cascading Style Sheet 2 (CSS2) [28]; (4)the Linking Module (enables navigation through the SMIL document. Most of SMIL linking constructs are similar to that from XLink [29]. Supports XPointer [30]); (5)the Media Object Module (describes SMIL media objects such as `img`, `audio`, `video`, `text`, `textstream`, `animation` and `ref` and object attributes); (6)the MetaInformation Module (describes the properties of the SMIL document and media objects. Supports use of the Resource Description Framework (RDF) [31, 32]); (7)the Structure Module (contains elements for structuring SMIL content); (8)the Timing and Synchronization Module (specifies the begin, end and duration of an element, etc.);(9)the Time Manipulations Module (controls the speed of playback of a media object,etc.); (10)the Transition Effects Module (describes transition effects such as Edge, Iris, Clock and Matrix wipes).

Besides modularity as described above, SMIL 2.0 defines a set of language profiles to allow a subset of SMIL functionality to be used for a particular authoring environment. In the following sections, we present comparisons on the architecture and feature set of both standards. For brevity, we will refer to SMIL 2.0 as simply SMIL in the remainder of this paper.

3 Architectural Comparison

We compare the various modules and functionalities, the level of abstraction and the degree of complexity of their scene representation formats and discuss the standards' scopes and purposes.

3.1 Representation format, scene complexity, and level of abstraction

MPEG-4's textual representation is based on VRML, augmented with timed BIFS command updates (a.k.a scene updates) and 2D nodes (VRML supports only 3D nodes). Let us refer to such a textual format as textual-BIFS or TEFS (as there is not yet a name convention in MPEG). TEFS is an unofficial format and this represents a drawback in MPEG-4. We shall use the term BIFS when we need to describe its bitstream or concept, and TEFS when we need to compare BIFS in its textual form. In many cases, BIFS requires a combination of routes, sensors and BIFS updates to achieve some simple content scenarios defined in SMIL. To illustrate this, we consider a scenario where the user clicks on an image button to start playing a video. This is written in both SMIL and TEFS as shown in Figures 4 and 5 respectively. In Figure 4 the object is positioned using the <`region`> attribute in the header of the document. The <`root-layout`> defines the dimension of the display window. The video object is enclosed in a <`par`> container that defines a parallel timing relationship between the objects (time containers to be described later) . The video begins playing when the user clicks on the image button.

The same scenario is described in Figure 5 using TEFS. The root of the scene is specified using `Layer2D` node (line L1), and the positioning of objects using `Transform2D` node (L3,L19). The image object is an `ImageTexture` (L8) mapped onto a `Rectangle` node (L12), and is associated with a

TouchSensor (L15). A conditional node (L18) is also defined to contain a BIFS update command for insertion of the video node into the scene. The video node is described as a MovieTexture node (L24) mapped onto a Rectangle node (L28). A ROUTE (L33) is defined to link the event source (mouse click event) to the event sink (activation of conditional node). When user clicks on the image object, the event is sensed by the TouchSensor which routes it to the conditional node that executes the BIFS command to insert the video into the scene. By default, once video is inserted into the scene, it starts playing when the video stream is available.

The positioning information given in both examples is different since the coordinates in MPEG-4 BIFS are BIFS coordinates with the origin at the center of the display window, and positive x and y directions to the right and top respectively. BIFS coordinates have to be converted to client window coordinates before rendering the objects onto a visual surface. The TEFS scene is compressed into BIFS by taking advantage of the context dependency of the nodes. SMIL presentations can also be compressed using typical compression software such as gzip, etc.

From the above examples, we can see that SMIL is a higher level textual format and is relatively easy to author. The MPEG committee is working on specifying a higher-level textual format known as eXtensible MPEG-4 Textual Format (XMT) (to be covered in Section 5). This will allow simple authoring and at the same time reap the benefit of scene compression. The price of complexity will then have to be migrated to the software that translates the higher level textual scene description to MPEG-4 BIFS. Besides the lower level of abstraction, content authors who work directly with the TEFS format will have to know the dimension of the media clips (see Figure 5) and be comfortable working with the BIFS coordinate system, which differs from that used in most authoring languages or tools.

3.2 Modules and Functionalities

Figure 6 presents the various functionalities in SMIL and MPEG-4. Unlike MPEG-4, SMIL deals only with textual description of scenes, and hence does not provide buffering, streaming and mux/demux facilities. While a SMIL document is created using SMIL, an MPEG-4 scene is authored using TEFS that uses VRML nodes and BIFS update mechanism. The created scene is then converted to BIFS stream. Metadata information are embedded as scene information at the beginning of a SMIL document, and in MPEG-4, it is stored in object descriptors (ODs) separate from the scene. An Object Descriptor (OD) consists of an Elementary Stream (ES) descriptor and several other descriptors, and is encoded and transported in a dedicated OD stream, separate from the BIFS stream. IPMP (Intellectual Property Management and Protection) is stored in IPMP Descriptors within an OD whereas profile and level information are stored in the InitialObjectDescriptor (initialOD). Decoder and QoS parameters as well as parameters for configuring the Sync Layer (SL), such as the resolution and accuracy of time stamps and clock references are stored in the ESDescriptor within an OD. Configuring the SL layer results in little overhead on the packet headers for a low bit-rate stream.

Both SMIL and MPEG-4 provide a textual syntax for describing a multimedia scene and for associating meta information. Although both share a few common modules, MPEG-4 provides additional modules for stream level management. MPEG-4 standardizes its binary format for carrying scene description, metadata, stream-level information etc., while SMIL standardizes its textual format. Unlike SMIL, MPEG-4 TEFS can be used for composing 3D objects, in addition to 2D ones (though the <`ref`> element in SMIL can be used for defining generic media types and hence certain 3D objects, it does not readily support 3D composition, such as texture mapping, lighting, etc.). BIFS contains geometrical primitives for defining graphics content, which is lacking in SMIL. The W3C has recently developed the Scalable Vector Graphics (SVG)[27] for describing

2D vector and mixed vector/raster graphics in XML, by integrating SMIL's Animation module for expressing powerful animation in 2D graphics. The integration of SMIL's Animation module into SVG is an example of how SMIL's functionality can be reused in other XML-based, non-SMIL languages. SMIL can also be used to define 3D animations as well.

Another major difference between SMIL and BIFS lies in their timing model. MPEG-4 can be configured to support various timing models, such as those based on clock recovery and timestamps (the traditional MPEG model), rate-based (e.g., fixed number of frames per second), or fully asynchronous operation (process on arrival). SMIL, on the other hand, assumes the presence of a global clock. A more detailed comparison of their timing models can be found in section 4.3.

In terms of the different purposes and scopes of both standards, it is clear that SMIL originates from the web community and as its name implies, the prime purpose of the standard is to integrate and synchronize multimedia data. It does not contain primitives for creating graphics content. Instead, it provides a framework that allows content authors to exercise options of using other non-SMIL languages while reaping the additional functionality that SMIL provides by integrating the desired SMIL modules into the host languages. MPEG-4 BIFS, on the other hand, originates from the TV community with the aim of standardizing many aspects of a multimedia streaming system, such as the networking component, object coding, stream-level components, etc. Since BIFS was aimed to be also a scene description facility for 3D objects, VRML-like syntax was naturally chosen to be the textual format at the time when the early specification was drafted. Unfortunately, the VRML syntax, unlike SMIL does not allow authors to work with existing practices, limiting the choice of authoring languages and tools. We will cover in section 5, a possible solution to such a problem. A summary of the architectural comparison of both standards is given in Table 1.

4 Feature Comparison

This section compares multimedia features such as the structure and layout schemes for describing a scene, the timing and animation controls, the interactive features, the document linking capabilities, the segmentation features, the transition effects and the support for dynamic update of scenes.

4.1 Structure and Layout

A SMIL document consists of a head and a body section. Spatial layout of SMIL objects, their metadata and transition effects are defined in the header section whereas the content body contains objects, their temporal relationships, interactive behaviors, timing manipulations and animation effects. Transition effects can also be applied in-line with the corresponding media object in the content body. SMIL also supports alternative layout scheme, such as Cascading Style Sheet (CSS) [28]. On the other hand, an MPEG-4 textual scene description consists of three parts:(1)VRML tree structure that defines the objects, their groupings, timing and spatial relationships;(2)interactive behaviors, animation and transition effects;(3)BIFS updates. Metadata and elementary stream level information (e.g. buffer sizes etc.) are stored in OD and ES descriptors. Unlike SMIL, the dimension of an MPEG-4 window is defined in the BIFS configuration which is carried in one of the object descriptor (`DecoderConfigDescriptor`), separate from the textual scene description. This allows the window size to be changed based on OD Update commands. A drawback is that the content authors will have to get used to defining the window's size not in the scene description, but in a separate OD file.

4.2 MPEG-4 Updates and SMIL Document Object Model (DOM)

MPEG-4 updates can occur at three levels: the scene level, the OD and the ES level. BIFS updates for modifying an MPEG-4 scene are time-stamped to indicate the time instant at which they take effect. A scene is modified when one or more objects are inserted or deleted, or when the interactive behaviors or the properties of the objects are changed. ES updates allows stream-level information (e.g. QoS parameters) of an existing OD to be updated independent of the scene description. Updates can be effected at certain instant of times or be triggered conditioned upon the occurrences of certain events. Updates can also be streamed from a remote source. SMIL conforms to the XML DOM [33, 34], which is a language and platform-independent interface written using OMG IDL [35]. DOM provides a set of objects to represent a document's content and a standard set of interfaces for accessing and manipulating them. This allows programmers to navigate and update a SMIL document by writing to the standard interface instead of product-specific APIs. SMIL however, does not provide textual description of such real-time updates.

4.3 Timing, Synchronization and Streaming

The timing model in MPEG-4 requires transmitted data streams to contain timing information such as Decoding Time Stamp (DTS) and Composition Time Stamp (CTS) that determines when data should be available for decoding and composition respectively. These time stamps are measured with respect to the Object Time Base (OTB) that can be reconstructed either from Object Clock Reference (OCR) inserted in the stream or by an indication that it is slaved to a time base conveyed with another stream. Unlike MPEG-4, SMIL's timing model does not rely on timing information carried in the streams. The time values in SMIL can be expressed either in UTC (Coordinated Universal Time) or the local time. The local time zone of the end-user platform is used.

SMIL objects can be temporally related in sequence, in parallel, or exclusively (using the time containers <seq>, <par> and <excl> respectively). As an example in Figure 7, for simplicity, we may consider the <par> as the first time container in the document. Both the audio and JPEG image are children of the <par> container and hence are in parallel sync. Both starts 5s while the video starts 10s after the beginning of document. The GIF image appears after the video stops playing and remains for as long as the presentation is active. We can use the BIFS command protocol to schedule such playback in MPEG-4. BIFS update commands are used to insert the MPEG-4 nodes corresponding to the objects into the scene at the specified begin times, and delete them from the scene after the indicated lifetime. The semantics of this process is shown as follows:

- *At 5s, insert image node "jpeg-image" and audio node "myaudio"*
- *At 10s, insert video node "myvideo" and MediaControl node*
- *At 15s, delete image node "jpeg-image"*
- *At 200s, delete audio node "myaudio"*
- *ROUTE command to route output of MediaControl node to a conditional node that contains an INSERT command to insert the image node "gif-image"*

A ROUTE command will route an output field of the MediaControl node (which becomes active when the associated video ends) to a BIFS update command (INSERT command) to insert the last image node. When a node is inserted, an OD update command is sent to create an OD and associate it with the corresponding node (process not shown in the figure). In other timing scenarios, an object can start relative to another object using the syncbase and syncToPrev timing constraints as shown in Figure 8. The video starts playing 10s after the JPEG image appears, while the audio starts playing 30s before the end of the video clip. In terms of timing controls, both SMIL and BIFS support altering of playback speed, accelerate and decelerate. Though BIFS can specify indefinite repetition of playback (i.e., loop), unlike SMIL, it cannot specify number or duration of repetitions.

During transmission, streams may arrive later than desired due to network delay, jitter, or due to the fact that these streams are served from different sources with different time base. E.g. a video is to be mapped onto a circle. The circle can be rendered locally, so there is no latency. However, the video might be streamed from a remote location with a different time base or through a network with significant delay and jitter. The objects can be made to start at the same time (co-start), or end at the same time (co-end), using the FlexTime model [36] in MPEG-4, thus ensuring accurate synchronization. SMIL addresses similar problem by providing finer control over the runtime synchronization behavior of a document, e.g. element can slip under such condition (soft-sync behavior), or the time container can wait till media delivery catches up (hard-sync behavior). Synchronization tolerance can also be specified to ignore a given amount of slew without forcing resynchronization. Authors can also assign an element to be the sync master, similar to the behavior of many players that slave video and other elements to audio.

Based on the above comparisons, we can see that the timing and synchronization capabilities in both SMIL and MPEG-4 are rather sophisticated, although a few features and timing controls are not readily supported in MPEG-4. Both MPEG-4 and SMIL are concerned with real-time streaming issues. In addition, SMIL allows authors to define synchronization tolerance and a sync master for the entire presentation.

4.4 Spatial and temporal segmentation of objects

Objects may need to be spatially segmented for several reasons, one of which is to define hotspot regions. Likewise, they may also be segmented into temporal subparts for multimedia indexing or for supporting certain gaming applications where the computer program selects subsequent video segments to play based on users' interaction at several points throughout the game. SMIL supports both spatial and temporal segmentation using the <`area`> element in its Linking Module. In MPEG-4 BIFS, hotspots are supported although not in a straightforward manner. Hotspots can be defined by overlaying transparent graphics objects onto the source object, and then associating these transparent objects with a sensor for sensing user events such as mouse clicks. The FlexTime Model in MPEG-4 is used for flexible management of media streams, including the decomposition of an object into temporal segments.

4.5 Interaction

Interaction is achieved using hyperlinks in SMIL and ROUTEs in MPEG-4. In Figure 9, the <`href`> element points to an audio element in the document as the link target, such that the audio plays when user clicks on the video object. In this case, the hyperlink is associated with the entire media object. To associate links to spatial portions of an object's visual display and to support other events such as double clicks, mouse move, etc., the <`area`> element is used instead. The same interactive behavior can also be achieved in MPEG-4 using a combination of route, sensor and BIFS command. The syntax is somewhat similar to Figure 5 and will not be shown here since syntactic comparison is not the subject of discussion in this section. SMIL can also load a new document from the beginning or middle of another presentation when the user clicks on an object in the current presentation. This can be done by having the hyperlink point to a new SMIL document. To load from some point in the middle of another document, we can have the hyperlink point to the identifier of the element/object at that point in the middle. In MPEG-4, loading a new scene as a result of user interaction can be done easily by routing sensor outputs to conditional node for inserting an `Inline` node that contains description of another scene. In MPEG-4, a new scene is always loaded from the beginning. Loading from any point of the new scene is not supported. A variant of this feature is to define subparts of a scene and load them exclusively via a `Switch` node.

Interaction can also involve displaying a web page or an external application when the user clicks on an object. Such interactive capabilities are supported in SMIL by having the hyperlink pointing to the appropriate destinations, and in MPEG-4 by using the `ApplicationWindow` node. The window region defined by the node is controlled by the external application, allowing natural user interaction with the application. Unlike SMIL, the web page and external application can be displayed within the display window of an MPEG-4 player, but not as an external window, such that the application co-exist with other objects within the scene. After an interaction has occurred, the state of the source object can be changed in SMIL by selecting the desired attribute values for the <a> element, and for the case of MPEG-4, by transmitting commands back to an MPEG-4 server to pause or stop the media stream using the `ServerCommand` node.

From the above comparison, we conclude that both SMIL and MPEG-4 support almost identical sets of linking capabilities, though MPEG-4 does not provide the flexibility of loading or linking into any arbitrary point of a new presentation. In terms of the range of interactive events, both share support for mouse events. Although MPEG-4 do not support keyboard events, it has a number of sensors for sensing events occurring in both 2D and 3D spaces, which are not available in SMIL, such as rotation of 2D objects around an axis (`DiscSensor` node), collision among objects (`Collision node`), mouse drag (`PlaneSensor/PlaneSensor2D node`), and timed events (`TimeSensor` node). The `TimeSensor` node is very useful for controlling timing with specified interval and for creating repeated animations, as will be discussed in the next section.

4.6 Animation and transition effects

Animation in MPEG-4 can be achieved via the use of a combination of ROUTEs, `TimeSensor` nodes and interpolator nodes, or via the use of BIFS-Anim. The former can be used to perform simple animation using various kinds of interpolator nodes, such as `ColorInterpolator`, `Position2DInterpolator`, etc. As shown in Figure 10, a circle is animated along a horizontal path by routing the output of the `TimeSensor` node to the input of the `Position2Dinterpolator`, and then the output of the same node to the input of the `Transform2D` node. The animation/control points (pairs of values for specifying animation path, i.e., (550,270), (600, 270) and (650,270) in the above e.g.) are specified as the `keyValue` of the interpolator node. The animation takes place over a duration of 5s which is the value of the cycleInterval field of the TimeSensor node.

We may use BIFS-Anim for facial, body animation, and others that require a large number of control points. As shown in Figure 11, the initialOD received at an initial stage is decoded to retrieve a set of BIFS configuration data, from which we derive the AnimationMask to decode the AnimationFrames from the BIFS-Anim stream into animation values and frame rate value. Once these animation values are processed by the adaptive arithmetic decoder, they are applied to animate the properties/attributes of an object at a rate matching the frame rate. The attributes and object to animate are identified by the AnimationMask. Note that BIFS-Anim is an animation technique at the BIFS level. It has been included in a proposal for extending VRML to VRML200x for supporting all MPEG-4 features at a higher level suitable for content authoring. Analogous to specifying nodes and fields in the AnimationMask, animation in SMIL can be achieved by specifying the objects and their properties to animate. In Figure 12, the <animateMotion> defines the type of animation to apply on the image, while the timing attributes control the timing of the animation, causing the image to accelerate for the first 2 seconds and decelerate in the last 2 seconds.

SMIL can also be used to animate the properties of other non-SMIL languages. For example, in Figures 13 and 14, SMIL is used to animate the width and position of a rectangle defined using SVG, a non-SMIL language. Animation can also be linearly paced as shown in Figure 14. The same animation can be realized in MPEG-4 by recognizing that the keyTimes attribute in SMIL has the same semantics as the key field of interpolator nodes (as shown in Figure 10), while values attribute is the same as keyValue field.

Besides linear interpolation, SMIL supports cubic Bezier animation path as well while MPEG committee is working on piecewise curves-based animation [37]. This work suggests an animation technique that eliminates redundant data such as unchanged control/animation points along an animation path, which will result in an improvement over BIFS-Anim. It is easier to create animation using SMIL than MPEG-4, which has limited support for transition effects. A summary of the feature comparison is available in Table 2.

5 The eXtensible MPEG-4 Textual Format (XMT)

In view of the lower level of abstraction that BIFS offers, the MPEG committee is working on the eXtensible MPEG-4 Textual Format (XMT) [15]. XMT is a framework for describing an MPEG-4 scene using a textual syntax. Like SMIL, XMT is XML-based, relatively easy to author and provides high level constructs. XMT is designed based on a two-tier architecture: XMT-O provides a high-level abstraction of MPEG-4 functionality, while XMT-A, provides a one-to-one deterministic mapping to MPEG-4 binary representation. While XMT is designed based on SMIL, XMT-A contains a subset of X3D [38], which is a direct XML encoding of VRML 200x [37]. This way, an MPEG-4 scene described using XMT-O can be played back by a SMIL player, or be converted to its lower-level XMT-A construct and be played back by a VRML player. Since the XMT-A mirrors MPEG-4 binary representation, an XMT document can naturally be played back on an MPEG-4 player as well. Thus, the goal of XMT is ultimately to allow content authors to work with existing practices, to exchange their content with other authors, tools or service providers and to facilitate interoperability with both X3D and SMIL. In a sense, XMT is the result of combining the strengths of both SMIL and BIFS.

Interoperability with SMIL is achieved by integrating a few of SMIL's modules. However, preservation of some of SMIL's semantics in XMT is a non-trivial task. To enforce compliance with such semantics will require more complex mapping to several MPEG-4 nodes, routes and update commands. By integrating SMIL's modules, XMT enhances MPEG-4 functionality. By adopting some SMIL-like syntax for describing the layout and structure of an MPEG-4 scene, it allows an MPEG-4 content to be described in a more natural way. XMT has a powerful <`Group`> construct for allowing operations to be applied collectively on a group of elements, though it does not seem to support relative spatial placement/arrangement of groups of objects, nor does it allow alignment of objects within a group yet. XMT does not currently support keyboard events due to the lack of support in MPEG-4. It also supports the "compromise" form of representation which includes elements like <`children`> etc. that seems quite redundant in a high-level textual format.

A vital and difficult part of the work revolves around finding a balance between a high-level construct and one which adequately exposes MPEG-4 functionality. Bridging these two different levels of abstraction within MPEG-4 is a very difficult task. The additional constraint of staying close to SMIL's architecture makes the problem even harder. We should also point out that there are no quantitative comparison metrics that one can apply in evaluating such specifications.

6 Conclusion

We have presented our comparative analysis on version 3 of MPEG-4 BIFS and SMIL 2.0. Scene composition in BIFS is more complex and of lower-level compared to SMIL. Both also differs in their copes and purposes. The prime purpose of SMIL (from web community) is to integrate and synchronize multimedia data. Instead of supporting all kinds of content (including graphics), it provides a framework that allows content authors to use its functionality in other non-SMIL languages. BIFS (TV/broadcasting) on the other hand, aims to standardize many aspects of a multimedia streaming system. DMIF specifies only the semantics for channel signaling and data streaming and hence its practical utility is limited. Although BIFS has better support for 3D features, on the things that both can do, SMIL appears to be better and easier to use and it also provides better timing, animation controls and more transition effects. The work on XMT aims to combine the strengths of both standards.

Regardless of the merits of the two specifications, we must note that today there is very little content developed using either of the two. It appears that, for some reason, neither of them resonated with the content creation community. To a large extent this can be attributed to lack of good content development tools. We have found that an additional reason is the fact that development of this content is a completely new component of the traditional content development pipelines (particularly in news and entertainment), and as a result it is difficult to integrate it within it. The conceptual merit is though clear: instead of trying to make a sophisticated computer into a simple video or audio player, one should identify its strongest platform features and make sure that the content and the applications surrounding it are using them in the best possible way. New content representation standards that embrace these features (along the lines of MPEG-4 and SMIL) are necessary in order to make this new content a reality, as they can tie together the necessary technology, business, as well as artistry.

7 Acknowledgement

We would like to thank Michelle Kim of IBM Research and Aaron Cohen of Intel Corporation for their valuable inputs on XMT and SMIL respectively.

References

[1] W3C Recommendation, `http://www.w3.org/TR/REC-smil`. *"Synchronized Multimedia Integration Language (SMIL) 1.0 Specification"*, June 1998.

[2] *Apple QuickTime 4.1*. `http://www.apple.com/quicktime`.

[3] *Microsoft Internet Explorer 6.0*. `http://www.microsoft.com`.

[4] *NIST S2M2 Player*. `http://smil.nist.gov/player`.

[5] *Oratrix GRiNS Editor*. `http://www.oratrix.com/GRiNS/index.html`.

[6] *RealNetworks RealPlayer 7*. `http://www.real.com`.

[7] *W3C World Wide Web Consortium*. `http://www.w3c.org`.

[8] W3C Recommendation, `http://www.w3.org/TR/2001/REC-smil20-20010807`. *Synchronized Multimedia Integration Language (SMIL) 2.0 Specification*, August 2001.

[9] O.Avaro, A.Eleftheriadis, C.Herpel, G.Rajan, L.Ward. MPEG-4 Systems Overview. In A.Puri, T.Chen, editor, *Signal Processing:Image Communication*. Marcel Dekker, 1999.

[10] R.Koenen, editor. *MPEG-4 Overview*, ISO/IEC JTC1/SC29/WG11 N2564, Rome, December 1998.

[11] *ISO/IEC 14496-5 Coding of Audio-Visual Objects: Reference Software, Final Draft International Standard*, ISO/IEC JTC1/SC29/WG11 N2505, October 1998.

[12] *EnvivioTV Web Site*. `http://www.envivio.com/solutions/etv`.

[13] *WebCine Web Site*. `http://www.mpeg-4.philips.com`.

[14] *The MPEG-4 Industry Forum (M4IF)*. `http://www.m4if.org`.

[15] M. Kim, S. Wood, and L-T. Cheok. Extensible MPEG-4 Textual Format. In *Proceedings of the workshops on ACM Multimedia 2000 Workshops*, 2000.

[16] A. Puri and A. Eleftheriadis. MPEG-4: An Object-Based Multimedia Coding Standard Supporting Mobile Applications. In *ACM Mobile Networks and Applications Journal, Special Issue on Mobile Multimedia Communications*, June 1998.

[17] *ISO/IEC 11172 International Standard (MPEG-1), Information Technology – Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to About 1.5 Mbit/s*, 1998.

[18] *ISO/IEC 13818 International Standard (MPEG-2), Information Technology – Generic Coding of Moving Pictures and Associated Audio (also ITU-T Rec.H.262)*, 1995.

[19] *Requirements Group, MPEG-4 Requirements version 4*, ISO/IEC JTC1/SC29/WG11 N1727, Stockholm, July 1997.

[20] *Requirements Group, MPEG-4 Applications Document*, ISO/IEC JTC1/SC29/WG11 N1729, Stockholm, July 1997.

[21] *Requirements Group, MPEG-4 Overview*, ISO/IEC JTC1/SC29/WG11 N1730, July 1997.

[22] *ISO/IEC 14496-1 Coding of Audio-Visual Objects: Systems*, ISO/IEC JTC1/SC29/WG11 N3850, October 2000.

[23] *ISO/IEC 14496-6 Coding of Audio-Visual Objects: Delivery Multimedia Integration Framework, Final Draft International Standard*, ISO/IEC JTC1/SC29/WG11 N2506, October 1999.

[24] C.Herpel, A.Eleftheriadis, and G.Franceschini. MPEG-4 Systems: Elementary Stream Management and Delivery. In A.Puri, T.Chen, editor, *Multimedia Systems, Standard, Networks.* Marcel Dekker, 2000.

[25] J. Signes, Y. Fisher, and A. Eleftheriadis. MPEG-4: Scene Representation and Interactivity. In A.Puri, T.Chen, editor, *Multimedia Systems, Standard, Networks.* Marcel Dekker, 2000.

[26] *ISO/IEC 14772-1:1997 International Standard, Information technology – Computer graphics and image processing – The Virtual Reality Modeling Language (VRML) – Part 1: Functional specification and UTF-8 encoding*, 2000.

[27] W3C Recommendation, `http://www.w3.org/TR/SVG`. *"Scalable Vector Graphics (SVG) 1.0 Specification"*, September 2001.

[28] W3C Recommendation, `http://www.w3.org/TR/REC-CSS1`. *"Cascading Style Sheets, level 1"*, January 1999.

[29] S.DeRose, E.Maler, D.Orchard and B.Trafford. *"XML Linking Language (XLink)"*. W3C Recommendation, `http://www.w3.org/TR/xlink`, June 2001.

[30] S.DeRose, R.Daniel Jr. *"XML Pointer Language (XPointer)"*. W3C Last Call Working Draft 8, `http://www.w3.org/TR/xptr`, January 2001.

[31] O.Lassila and Ralph R. Swick. *"Resource Description Framework (RDF) Model and Syntax Specification"*. W3C Recommendation, `http://www.w3.org/TR/REC-rdf-syntax`, February 1999.

[32] D.Brickley and R.V.Guha. *"Resource Description Framework (RDF) Schema Specification"*. W3C Candidate Recommendation, `http://www.w3.org/TR/rdf-schema`, March 2000.

[33] W3C Recommendation, `http://www.w3.org/TR/REC-DOM-Level-1`. *"Document Object Model (DOM) Level 1 Specification"*, October 1998.

[34] W3C Recommendation, `http://www.w3.org/TR/DOM-Level-2-Core`. *"W3C Document Object Model (DOM) Level 2 Specification"*, November 2000.

[35] OMG (Object Management Group)IDL (Interface Definition Language), `http://www.omg.org`. *"The Common Object Request Broker:Architecture and Specification, version 2.3.1"*, October 1999.

[36] *ISO/IEC 14496-1:PDAM2, The FlexTime Model*, ISO/IEC JTC1/SC29/WG11 M5682, February 2000.

[37] *ISO/IEC 14772-1:200x International Standard, Information technology – Computer graphics and image processing – The Virtual Reality Modeling Language (VRML) – Part 1: Functional specification*, 2000.

[38] *Extensible 3D (X3D) Graphics.* `http://www.web3d.org/fs_specifications.htm`, 2001.

TABLE AND ILLUSTRATION CAPTIONS

TABLE 1: ARCHITECTURAL COMPARISON BETWEEN SMIL AND BIFS

		SMIL	BIFS
Timing		supported	supported
Interaction		supported	supported
Animation and transition effects		supported	supported
Buffering		not supported	supported
Streaming interface		not supported	supported
Mux/Demux facilities		not supported	supported
2D object	description	supported	supported
	composition	supported	supported
3D object	description	supported	supported
	composition	not supported	supported
Graphics content		not supported	supported
Text-based scene description		SMIL syntax (standardized)	VRML-like syntax and udpate mechanisms (syntax for update mechanism is not standardized)
Output format		not standardized, can be a text document or can use gzip or other compression tools to create a binary stream	standardized binary stream format
Meta-data information		uses RDF at the beginning of the document	stored in ODs in a file separate from the scene description file
Framework for integrating modules with other languages		supported	supported (via XMT)

TABLE 2: FEATURE COMPARISON BETWEEN SMIL AND BIFS

		SMIL	BIFS
Structure and Layout	scene description file	head and body section alternative layout scheme: CSS	(1) VRML tree (objects, spatial and temporal relationships,groups) (2) interactive, behaviors, animations and transition effects (3) scene updates
	Dimension of application window	specified at the head section	specified in one of the descriptors in a separate file
Timing, synchronization and streaming	Timing model and mechanism	clock values rely on timing information in elementary streams	clock values measured in local time dependent on end-users' platform
	Time containers	`<par>`,`<seq>`, `<excl>`	Flextime model
	Relative timing	synToBase, synToPrev	Flextime model
	Timing controls	playback speed, accelerate, decelerate, auto-reverse, n repetitions, repetitions over specified duration,	playback speed, indefinite repetitions (looped playback), no fine control over number of repetitions

	Runtime object synchronization	soft-sync,hard-sync behavior, synchronization tolerance, and sync master	Flextime model, allows objects to co-start or co-end
Interaction	loading a new presentation upon user interaction	supported	supported
	loading from the middle of a new presentation	supported	not supported
	displaying external application	supported	supported
	change the state of the source object when clicked (e.g: to pause)	`<a>` element with the desired attribute values	Servercommand node to transfer commands to server to pause stream
	mouse and keyboard events	supported	keyboard events not supported
	2D sensor events	supported	supported
	3D sensor events (e.g: collision)	not supported	supported
Dynamic updates of BIFS scenes and SMIL documents		(1) event-based updates (2) DOM API as programmatic interfaces to effect updates (3) no textual description of real-time updates	(1) scene, OD and ES level updates (2) time-based and event-based updates (3) remote source updates
Spatial and temporal segmentation of objects		decomposes objects into temporal segments, hotspots can be easily defined	Temporal segmentation using Flextime model, hotspots defined in an unnatural manner
Animation and transition effects		uses both timing and animation elements, less verbose, easier to express. More intuitive. Large support of transition effects. Includes both 2D and 3D animations	uses BIFS-Anim or a series of routes and sensors, less intuitive, more verbose. Limited support for transition effects. Includes both 2D and 3D animations

Figure 1: HIGH-LEVEL VIEW OF AN MPEG-4 TERMINAL

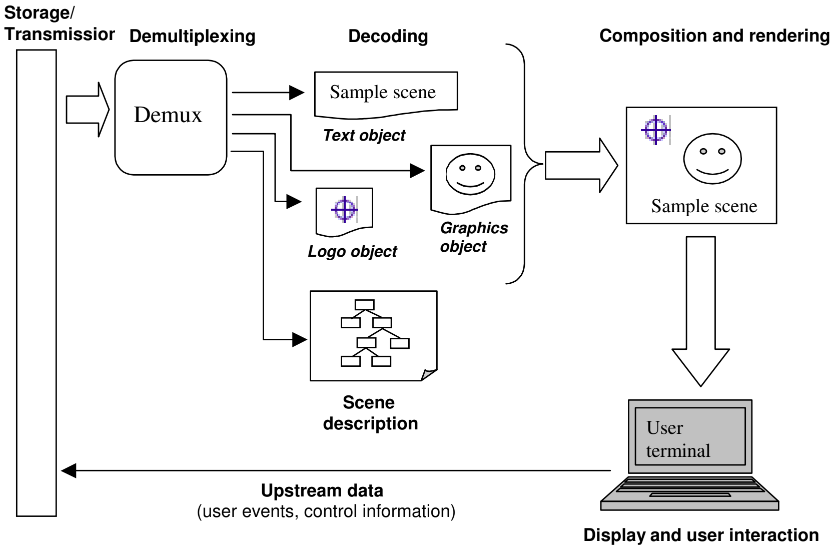

Figure 2: OVERALL ARCHITECTURE OF AN MPEG-4 TERMINAL

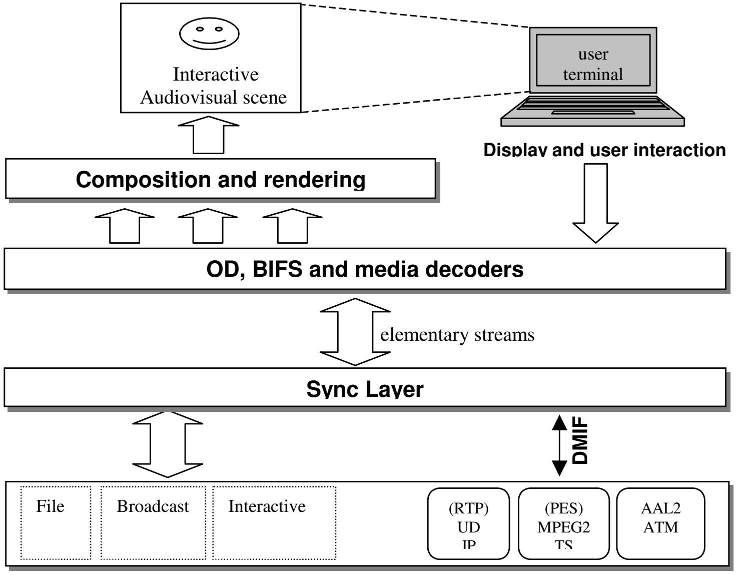

Figure 3: LOGICAL TREE STRUCTURE OF AN EXAMPLE SCENE

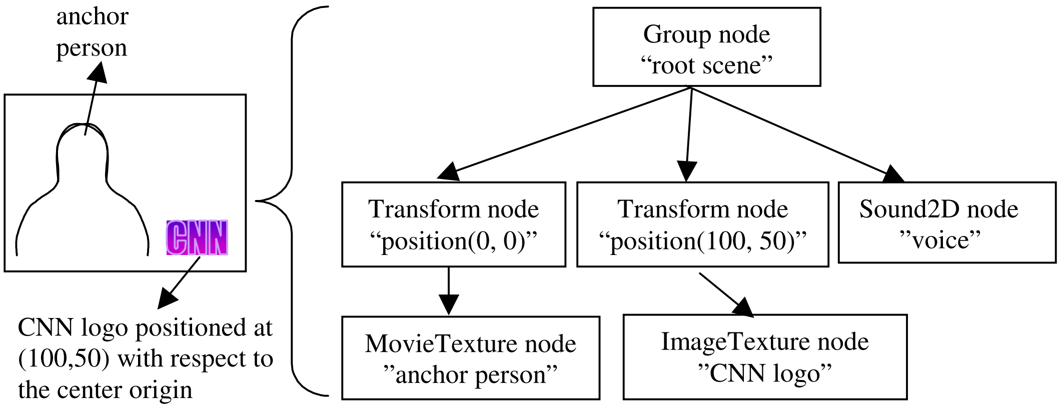
anchor person
CNN
CNN logo positioned at (100,50) with respect to the center origin
Group node “root scene”
Transform node “position(0, 0)”
Transform node “position(100, 50)”
Sound2D node ”voice”
MovieTexture node ”anchor person”
ImageTexture node ”CNN logo”

Figure 4: CONTENT REPRESENTATION USING SMIL.

```
<smil>
    <head>
        <layout >
            <root-layout width="320" height="240"/>
            <region id="reg1" left="170" top="110"/>
            <region id="reg2" left="50"  top="50"/>
        </layout>
    </head>
    <body>
        <par>
        <img id="imageBtn" src="myimage.jpg" region="reg1"/>
        <video src="myvideo.mpg" begin="imageBtn.click" region="reg2"/>
        </par>
    </body>
</smil>
```

Figure 5: CONTENT REPRESENTATION USING TEFS AS A TEXTUAL FORMAT WITH A ONE-TO-ONE MAPPING TO MPEG-4 BIFS.

```
DEF ROOTSCENE Layer2D {
 children [
   Transform2D {
     translation 10, 10
     children [
     Shape {
           appearance Appearance {
               texture ImageTexture {
                   url myimage.jpg
               }
           }
           geometry Rectangle {
               size 40, 30
           }
       } DEF TS TouchSensor {}
     ]
   }
   DEF COND Conditional { INSERT NODE ROOTSCENE.children
   Transform2D {
     translation -10, -10
     children [
       Shape {
           appearance Appearance {
               texture MovieTexture {
                   url myvideo.jpg
               }
           }
           geometry Rectangle {
               size 160, 120
           }
       }]
   }
   }] } ROUTE TS.isActive TO COND.Activate
```

Figure 6: SMIL AND MPEG-4 FUNCTIONALITY

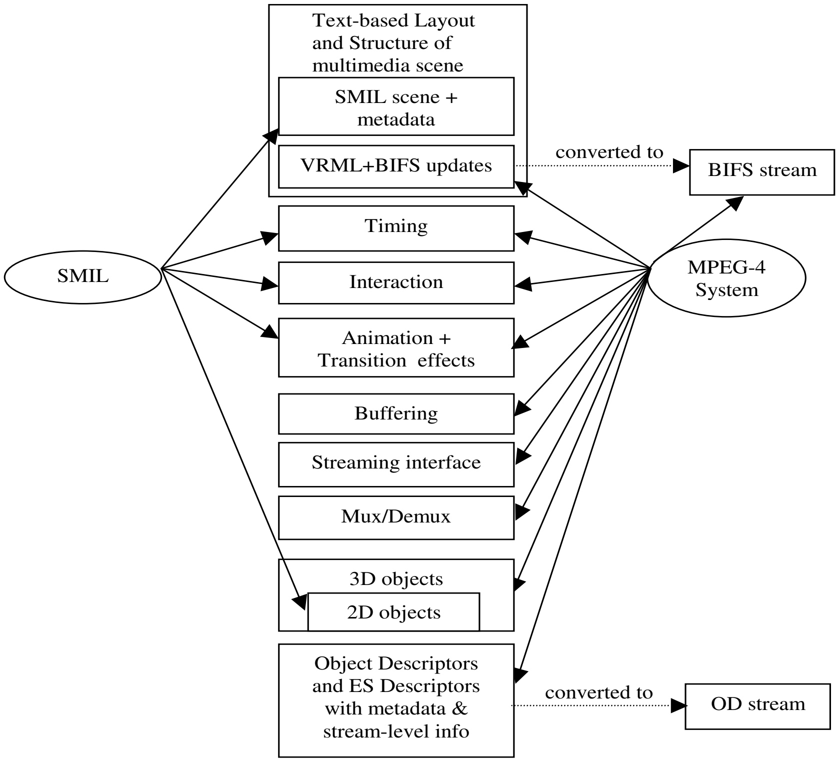

Figure 7: SEQUENTIAL AND PARALLEL PLAYBACK OF MEDIA OBJECTS IN SMIL

```
<par begin="0s">
    <img id="jpeg-image" begin="5s" dur="10s" src="image1.jpg" />
    <audio id="myaudio" begin="5s" end="200s" src="myaudio.wav"/>
    <seq>
        <video id="myvideo" begin="10s" src="myvideo.mpg"/>
        <img id="gif-image" src="image2.gif"/>
    </seq>
</par>
```

Figure 8: TIMING BEHAVIOR USING `syncbase` IN SMIL

```
<par>
    <img id="myimage" begin = "20s" src="myimage.jpg"/>
    <video begin="myimage.begin + 10s" src="myvideo.mpg"/>
    <audio begin="video.end - 30s"/>
</par>
```

Figure 9: USING HYPERLINKS TO DEFINE INTERACTIVE BEHAVIOR IN SMIL

```
<a href="http://www.somewebsite.edu/currentDoc.smi#AudioObjectID">
    <video src="myvideo.mpg"/>
</a>
```

Figure 10: ANIMATION USING `ROUTEs`, `TimeSensor` NODE AND INTERPOLATOR NODE

```
DEF TR Transform2D {
    translation 500 270
    children [
        Shape {
            appearance {

            }
            geometry Circle {
                radius 20
            }
        }
    ]
    } DEF TS TimeSensor { enabled TRUE
        startTime 0
        stopTime -1
        cycleInterval 5
    } DEF PI Position2DInterpolator {
        key [0.0 0.5 1.0]
        keyValue [ 550 270, 600 270, 650 270]
    }
ROUTE TS.fraction_changed TO PI.set_fraction   ROUTE
PI.value_changed TO TR.translation
```

Figure 11: PROCESSING BIFS-Anim STREAM IN MPEG-4

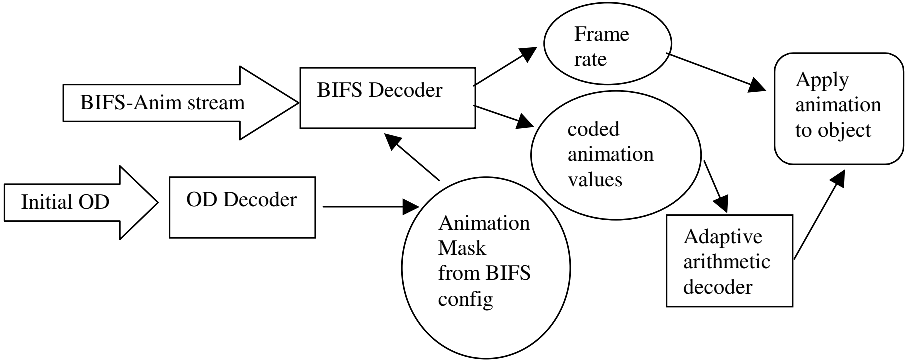
Frame rate
BIFS-Anim stream
BIFS Decoder
Apply animation to object
coded animation values
Initial OD
OD Decoder
Animation Mask from BIFS config
Adaptive arithmetic decoder

Figure 12: SMIL ANIMATION USING ANIMATION ELEMENT AND TIMING ATTRIBUTES

```
<img src="myimage.jpg">
    <animateMotion dur="8s" accelerate="0.25s" decelerate="0.25s"/>
</img>
```

Figure 13: SMIL ANIMATION BY SPECIFYING A SET OF VALUES FOR THE ANIMATING ATTRIBUTE

```
<rect >
    <animate attributeName="width" values="40;100;40" dur="10s"/>
</rect>
```

Figure 14: SMIL ANIMATION WITH UNEVEN PACING

```
<rect >
    <animate attributeName="x" values="0;50;100" dur="10s"
keyTimes="0;0.8;1" calcMode="linear"/> </rect>
```