INTERNATIONAL ORGANISATION FOR STANDARDIZATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND ASSOCIATED AUDIO

ISO/IEC JTC1/SC29/WG11
MPEG97/2133
April 1997, Bristol

Title: APIs for MPEG-4 Systems

Source: A. Puri, R. L. Schmidt (AT&T),
A. Eleftheriadis, H. Kalva, Y. Fang (Columbia University)

Status: Proposal

1. Overview

The focus of current MPEG-4 Systems work on standardization of a fixed binary format stream (BIFS) representation is a step in the right direction. Moreover, the specification of (either normative or informative) a set of useful API's may further serve to highlight and illustrate the potential advanced capabilities provided by MPEG-4 and the demands they may impose on resources. For example, the interaction between Decoding and Composition is typically hidden, however, in applications with user interactivity, and limited computational/memory resources, a clearer interface may be the key between being able to effectively provide certain functionalities versus not.

In this document, examples of a few basic as well as advanced functionalities are presented in the form of example API's. The API specification is draft and
requires lot more work. However, it is introduced to illustrate sometimes simple as well as sometimes complex relationship between Decoding and Composition functions.

2. Organization of MPEG-4 Systems

The organization of MPEG-4 Systems into the following categories (of classes/interfaces) appears adequate.

- Basic (includes AVObject)
- Control and Events

- Demux
- Decoding
- Composition
- Functionalities
- Presentation

It is suggested that all the classes/interfaces be initially placed into **mpeg4.system** package. Perhaps later, they can be paritioned into individual packages such as **mpeg4.system.decoding** etc, depending on the specific category. The example APIs discussed in the remaining part of this document are assumed to be part of Functionalities category. Java language is used for API specification.

3. List of APIs

Table 1 A List of Proposed APIs

No	Functionality Categories and API	Explanation
	<u>**Session**</u>	
1.	Stream Editing API	MPEG-4 stream editing
	<u>**Aural/Visual**</u>	
1.	Progressive API	Progressive decoding and composition of an AV object under user control
2.	HotObject API	Decoding, enhancement and composition of an AV object based on user control
3.	Directional API	Decoding of AV object with viewpoint (or accoustic) directionality selected by used
4.	TrickMode API	Decoding of portions of AV object and composition of an AV object under user control
5.	Transparency API	Decoding, refinement and composition of an AV object based on transparency and user control

4. Session Functionality APIs

The following API specifies session related functionality. A particular example is that of editability of MPEG-4 bitstreams without complete decoding.

4.1 Stream Editing API

Class mpeg4.system.StreamEdit

public class **StreamEdit**

This class allows determination of contents as well as modification of MPEG-4 systems streams. Operations such as access, copy, add, replace, delete and others are supported.

Constructors

```
public StreamEdit()
```

Methods

```
public int[]      getObjectList(Mp4Stream srcstrm)
```

Returns the list of objects in the srcstrm. The returned object is the cumulative table of objects in the bitstream.

```
public boolean   replaceObject (Mp4Stream srcstrm, ulong srcobjid, Mp4Stream deststrm, ulong destobjid)
```

Replaces the occurrence of objects with object id destobjid in deststrm with corresponding occurences of object with object id srcobjid in the srcstrm. The object tables are updated accordingly. The operation returns true on successful replace whereas false indicates a failure to replace.

`public boolean    replaceObjectAt (Mp4Stream srcstrm, ulong srcobjid, ulong m, Mp4Stream deststrm, ulong destobjid, ulong n)`

Same semantics as replaceObject(), except that the position to start to replace is specified. Replaces the destination object from nth occurences of destobjid with source objects from the mth occurrence of srcobjid. For m=n=0, it performs identically to replaceObject().

`public boolean    containsObjectType (MP4Stream srcstrm, ulong objtype)`

Returns true if srcstrm contains an object of objtype, else returns false.

`public boolean    addObjects (Mp4Stream srcstrm, ulong srcobjId, Mp4Stream deststrm)`

Adds objects of srcobjid from srcstrm to deststrm. Returns true if successful, else returns false.

`public boolean    addObjectsAt (Mp4Stream srcstrm, ulong srcobjid, Mp4Stream deststrm, ulong destobjid, ulong n)`

Adds objects of srcobjid from srcstrm to deststrm starting after nth occurrence of destobjid. Returns true if successful, else returns false.

`public boolean    copyObjects (Mp4Stream srcstrm, ulong srcobjid, Mp4Stream deststream, destobjid)`

Copies objects with srcobjid in srcstrm to deststream with new object id, destobjid. If deststrm does not exist, it is created. If it exists it is overwritten. This operation can be used to create elementary stream objects from multiplexed strams for subsequent operations. Returns true if successful, else returns false.

`public boolean    deleteObjects (Mp4Stream deststrm, ulong destobjid)`

Delete all objects with destobjid in deststrm. Also remove all composition information. Returns true if successful, else returns false.

`public boolean    spliceAt (Mp4Stream deststrm, ulong destobjid, ulong n, Mp4Stream srcstrm)`

Splice deststrm after nth occurrence of destobjid and paste the srcstrm. Returns true if successful, else returns false.

5. Aural/Visual Functionality APIs

The following APIs specify example aural/visual functionalities, some requiring decoder and composition interaction. The work is very preliminary; it is mainly intended to solicit comments on potential usefulness of specifying functionalities in this manner.

A core class common to the APIs and used for video decoding is Mp4Vdecoder and is listed as follows. It is assumed that a similar class called Mp4ADecoder exists for audio decoding.

Class mpeg4.systems.Mp4VDecoder

public class **Mp4VDecoder**
extends VideoDecoder

This class extends VideoDecoder , an abstract class. It contains methods to decode various types of visual bitstreams.

Constructors

`public Mp4VDecoder()`

Methods

`public VObject    baseDecode(Mp4Stream basestrm)`

Decodes a base MPEG-4 video stream, basestrm, and returns a decoded visual object ,VObject.

```
public VObject    sptEnhDecode(Mp4Stream enhstrm)
```

Decodes a spatial enhancement MPEG-4 video stream, enhstream, and returns a decoded visual object, VObject.

```
public VObject    tmpEnhDecode(Mp4Stream enhstrm)
```

Decodes a temporal enhancement MPEG-4 video stream, enhstrm, and returns a decoded visual object ,VObject.

```
public VObject    snrEnhDecode(Mp4Stream enhstrm, int level)
```

Depending on the level, decodes a snr enhancement MPEG-4 video stream, enhstrm, and returns a decoded visual object ,VObject.

```
public VObject    datapartDecode(Mp4Stream enhstrm, int level)
```

Depending on the level, decodes a data partitioned MPEG-4 video stream, enhstrm, and returns a decoded visual object, Vobject.

```
public VObject    trickDecode(Mp4Stream trkstrm, int mode)
```

Depending on the mode, skip and decode trick stream, trkstrm, and returns a decoded visual object, Vobject.

```
public MeshObject meshAuxDecode(Mp4Stream auxstrm)
```

Decodes an MPEG-4 auxillary video stream, auxstrm, and returns a mesh object, MeshObject, which includes mesh geometry and motion vectors.

5.1 Progressive API

Class mpeg4.systems.ProgAVObject

public class **ProgAVObject**
extends AVObject

A ProgAVObject allows progressive refinement of quality of an AV object under user control. Currently, visual objects are assumed to be static (still image vops).

Constructors

```
public ProgAVObject()
```

Methods

```
public void startDec ()
```

Start decoding of data.

```
public void stopDec ()
```

Stop decoding of data.

```
public void pauseDec ()
```

Temporarily suspend decoding of data.

```
public void resumeDec ()
```

Restart decoding of data from current state of pause.

```
public int  selectProgLevel ()
```

Select level up to which decoding of transform (DCT or wavelet) coefficients will take place. A level constitutes coefficients up to a certain position in scan order.

```
public void attachDecoder (Mp4Stream srcstrm, int proglvl)
```

Attach a decoder to srcstrm in preparation to decode a valid MPEG-4 stream and specifies the prog level up to which decoding is to take place

```
public void offsetStream (Mp4Stream srcstrm, ulong offset)
```

Allow an offset into the srcstrm as the target where the decoding may start. In reality, the actual target location may be beyond the required target and depends on the location of valid entry point in the stream.

5.2 Hot Object/Region API

Currently this API allows interaction with hot (active) AV objects. It may be extended in future to allow interaction with hot regions within an object. This API is intended to allow one or more advanced functionalities such as spatial resolution enhancement, quality enhancement, temporal quality enhancement of an AV object. The actual enhancement that occurs is dependent on user interaction (via mouse clicks/menu) and the enhancement streams locally/remotely as well as enhancement decoders available.

Class mpeg4.systems.HotAVObject

public class **HotAVObject**
extends BaseAVObject

HotAVObject is a class that triggers the action of enhancement of an AVObject provided that the object is a hot object. Thus hot objects have some enhancement streams associated with them that are triggered when needed. This class extends BaseAVObject, which is used primarily to decode base (layer) streams. Further, the definition of hot objects will be extended in future to include regions when KeyRegions are supported by Video.

Constructors

```
public HotAVObject()
```

Methods

```
public void startDec ()
```

Start decoding of data.

```
public void stopDec ()
```

Stop decoding of data.

```
public void pauseDec ()
```

Temporarily suspend decoding of data.

```
public void resumeDec ()
```

Restart decoding of data from current state of pause.

```
public int  selectHotType ()
```

Select type of enhancement (spatial, quality, temporal etc).

```
public Mp4Stream  enhanceObject (int type)
```

Use selected enhancement type to obtain needed enhancement stream.

```
public void attachDecoder (Mp4Stream srcstrm, int type)
```

Attach a decoder to srcstrm in preparation to decode a valid MPEG-4 stream and specifies the type of decoding is to take place

```
public void offsetStream (Mp4Stream srcstrm, ulong offset)
```

Allow an offset into the srcstrm as the target where the decoding may start. In reality, the actual target location may be beyond the required target and depends on the location of valid entry point in the stream.

Class mpeg4.systems.BaseAVObject

public class **BaseAVObject**

This is a basic class allowing decoding of base AV object stream.

Constructors

```
public BaseAVObject()
```

Methods

```
public void startDec ()
```

Start decoding of data.

```
public void stopDec ()
```

Stop decoding of data.

```
public void attachDecoder (Mp4Stream basestrm)
```

Attach a decoder to basestrm in preparation to decode a valid MPEG-4 stream to whose decoding is to take place.

5.3 Directional API

This API allows interaction with directionally sensitive AV objects. It supports static visual objects (still vops), dynamic visual objects (moving vops), as well as directional speech and audio. For visual objects it permits view-point to be selected and only the corresponding bitstreams are decoded and decoded data forwarded to compositor. For sural objects an analogous operation takes place depending on desired auralpoint. At the present time, a number of predefined directional choices are assumed.

Class mpeg4.systems.DirecAVObject

public class **DirecAVObject**
extends BaseAVObject

DirecAVObject is a class that allows creation of objects that respond to x-y-z location in space (in the form of prequantized directions for now). This class is most easily explained by assuming a bitstream composed of a number of static visual vops coded as an AV object such that depending on the user interaction, vops corresponding to one or more viewpoint are decoded as needed. There class is equally suitable to decoding dynamic AVObjects.

Constructors

```
public DirecAVObject()
```

Methods

```
public void startDec ()
```

Start decoding of data.

```
public void stopDec ()
```

Stop decoding of data.

`public void pauseDec ()`

Temporarily suspend decoding of data.

`public void resumeDec ()`

Restart decoding of data from current state of pause.

`public void loopDec ()`

This method allows user interactive decoding of a dynamic visual object as a defined sequence of static vops forming a closed loop. Similar analogy may be applicable to audio as well. User selection occurs via mouse clicks or menus.

`public int  selectDirec ()`

Select the direction (scene orientation). A number of prespecified directions are allowed and selection takes place by clicking a mouse on hot points on the object or via a menu.

`public Mp4Stream  enhanceObject (int orient)`

Use selected scene orientation to obtain needed temporal auxillary (enhancement) stream.

`public void attachDecoder (Mp4Stream srcstrm, int orient)`

Attach temporal auxillary (enhancement) decoder to srcstrm in preparation to decode a valid MPEG-4 stream and specifies the selected scene direction of AV object.

`public void offsetStream (Mp4Stream srcstrm, ulong offset)`

Allow an offset into the srcstrm as the target where the decoding may start. In reality, the actual target location may be beyond the required target and depends on the location of valid entry point in the stream.

Class mpeg4.systems.BaseAVObject

This is same as that presented earlier.

5.4 Trick Mode API

Trick Mode API supports conditional decoding under user control for enhanced trick play capabilities. Enhanced trick play can be defined as enabling of VCR/CDPlayer like functions such as different speeds for FF or FR, Freeze Frame, Random Access as well others such as reverse play etc, however with the difference that, MPEG-4 can allow these capabilities on individual AV object basis in addition to that on composited scene basis.

Class mpeg4.visual.TrickAVObject

public class **TrickAVObject**
extends AVObject

TrickAVObject is a class that can be used to form objects that allow decoding suitable for trick play.

Constructors

`public TrickAVObject()`

Methods

`public void startDec ()`

Start decoding of data.

`public void stopDec ()`

Stop decoding of data.

```
public void pauseDec ()
```

Temporarily suspend decoding of data.

```
public void resumeDec ()
```

Restart decoding of data from current state of pause.

```
public void loopDec ()
```

This method allows user interactive decoding of selected portions of the srcstream for forward or reverse playback at a variety of speeds.

```
public boolean    selectDirec ()
```

Select the direction of decoding. Returns true when trick decoding is done in (normal) forward direction, else it returns false when reverse direction for trick decoding is selected.

```
public Mp4Stream  enhanceObject (boolean decdirec)
```

Obtain the MPEG-4 stream to be decoded in direction specified by decdirec

```
public void attachDecoder (Mp4Stream srcstrm, int decdirec)
```

Attach trick decoder to srcstrm in preparation to decode a valid trick mode MPEG-4 stream and specifies the direction of decoding.

```
public void offsetStream (Mp4Stream srcstrm, ulong offset)
```

Allow an offset into the srcstrm as the target where the decoding may start. In reality, the actual target location may be beyond the required target and depends on the location of valid entry point in the stream.

5.5 Transparency API

Transparency API supports selective decoding of regions of an object under user control. In case of visual objects, it is assumed that encoding is done in a manner where a large object is segmented into a few smaller regions by changing the transparency of other pixels in the object. The pixels not belonging to region of interest are coded by assigning them a selected key color not present in the region being coded. This API allows decoding under user control such that a few or all of the regions may be coded. Further, for a region of interest, enhancement bitstream may be requested to improve the spatial or temporal quality. The key color for each region is identified to compositor. The user may not need to decode all regions either due to limited bandwidth/computing resources, portions of object are hidden and are thus not needed, or a much higher quality is needed for a specific region at the cost of no image or poor image in other regions. The process of using a key color is similar to “chroma key” in broadcast applications.

<u>Class mpeg4.systems.TranspAVObject</u>

public class **TranspAVObject**
extends BaseAVObject

TranspAVObject is a class that can be used to form objects with transparency information. Both aural and visual object types are handled.

Constructors

```
public TranspAVObject()
```

Methods

```
public void startDec ()
```

Start decoding of data.

```
public void stopDec ()
```

Stop decoding of data.

```
public void pauseDec ()
```

Temporarily suspend decoding of data.

```
public void resumeDec ()
```

Restart decoding of data from current state of pause.

```
public int  getRegion ()
```

Select the region by number in a listed menu or by clicking on hotpoints (also translates to a number).

```
public Mp4Stream  enhanceObject (int type, int regnum)
```

Use selected enhancement type to obtain needed enhancement stream for the region regnum.

```
public void attachDecoder (Mp4Stream srcstrm, int type, int regnum)
```

Attach decoder to srcstrm in preparation to decode a region and its key color.

```
public void offsetStream (Mp4Stream srcstrm, ulong offset)
```

Allow an offset into the srcstrm as the target where the decoding may start. In reality, the actual target location may be beyond the required target and depends on the location of valid entry point in the stream.

Class mpeg4.systems.BaseAVObject

This is same as that presented earlier.

6. Summary

We have discussed the potential organization of MPEG-4 Systems. Further a new category called Functionalities is introduced and a few example APIs are introduced to show simpler as well as more complicated interactions between Decoding and Composition. Feedback on potential for inclusion of such APIs (after more work) in systems specification is sought.