AUDIO TOOLS IN MPEG-4 FOR INTERACTIVE APPLICATIONS

Nicolas Déflache
Formation Supérieure aux Métiers du Son
Conservatoire National Supérieur de Musique et de Danse
Paris – France
ndeflache@mageos.com

The MPEG-4 standard introduces innovative tools designed for modern multimedia applications. It normalizes audio and visual compression schemes, but also ways to represent synthetic objects: the Synthetic/Natural Hybrid Coding capabilities make it possible to choose a suitable codage for each source. On decoding, sources of different origins are mixed. The powerful SAOL programming language gives a concrete expression to the Structured Audio concepts. The object-oriented structure of MPEG-4 makes it very suitable for Interactive applications: this paper gives the basic knowledge for this usage.

In Interactive applications, the final sound-image is not permanently set by the soundtrack's author during production. Some characteristics of the image are let to be modified dynamically at the time of the reproduction. As a new tool for audio-visual transmission, MPEG-4 takes up the challenge of bringing dynamic controls to modern multimedia products.

After a brief presentation of the basic organization of the MPEG-4 standard, we will progressively focus on 2 of its innovative capabilities. First, with Structured Audio, a new interesting way to describe sounds is explained. Second, we will present BIFS and its sub-part AudioBIFS, the MPEG-4 powerful scene description languages.

Most of the concepts presented here came out of a review of Eric Scheirer's and other MIT researchers' publications: important ideas leading to the integration of Audio among the MPEG-4 object-oriented principles were indeed exposed by this group.

1. MPEG-4 Overview

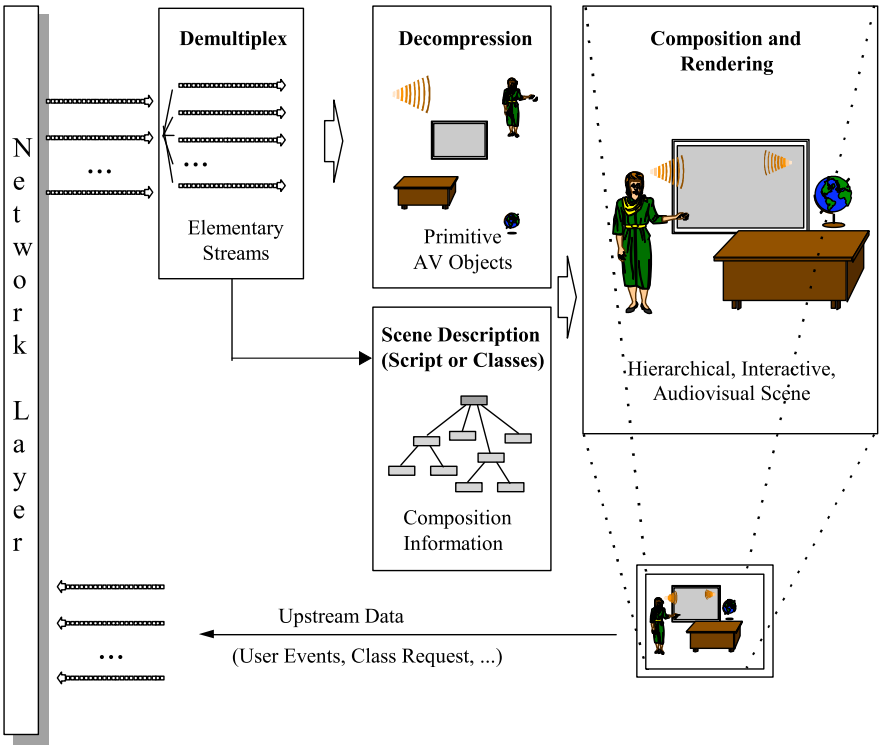
1.1- Basics [1]

MPEG-4 is an ISO/IEC standard developed by MPEG (Moving Picture Experts Group). It is the result of an international effort of hundreds of researchers and engineers from all over the world.

	Final Comittee Draft	ISO standard
Version 1	October 1998	March 1999
Version 2	December 1999	Early 2000

MPEG-4 standardizes ways to:

- Represent units of aural, visual or audiovisual content, called *Media Objects*. These media objects can be of natural or synthetic origin;
- Describe the *composition of* these *objects* to create compound media objects, that form *audiovisual scenes*;
- *Multiplex* and *synchronize* the data associated with media objects, so that they can be transported over network channels;
- *Interact* with the audiovisual scene generated at the receiver's end.

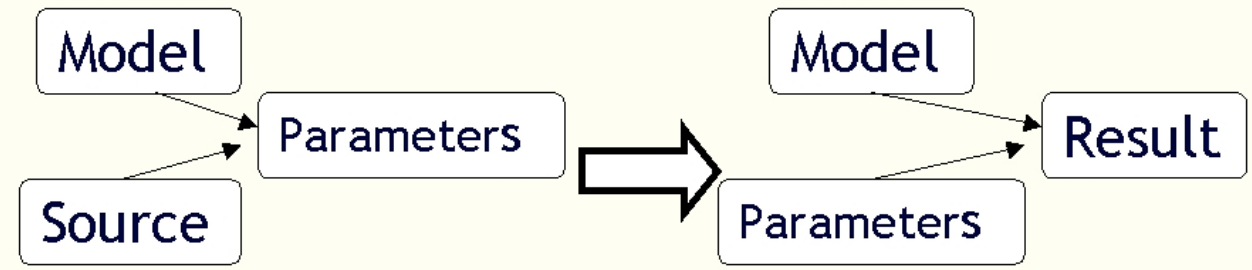


1.2- Principles for Compression of Media Objects [3]

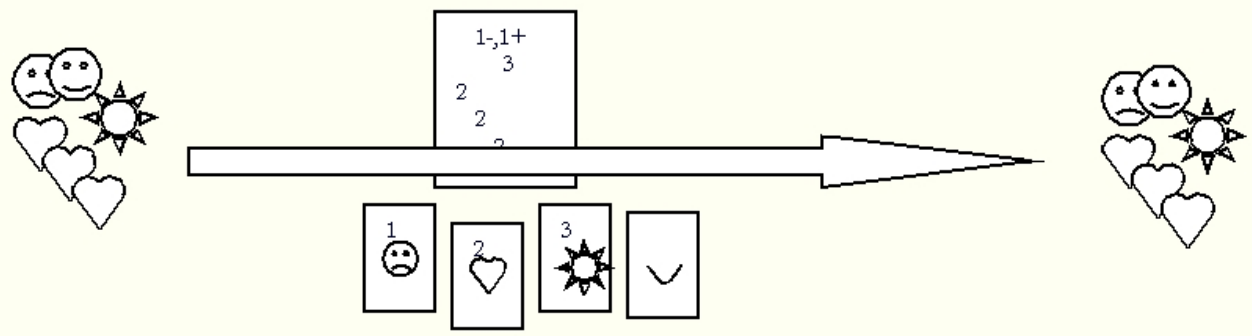
Entropic encoding : with a lossless compression scheme, we use fewer bits to represent the sequences that statistically occur more often.

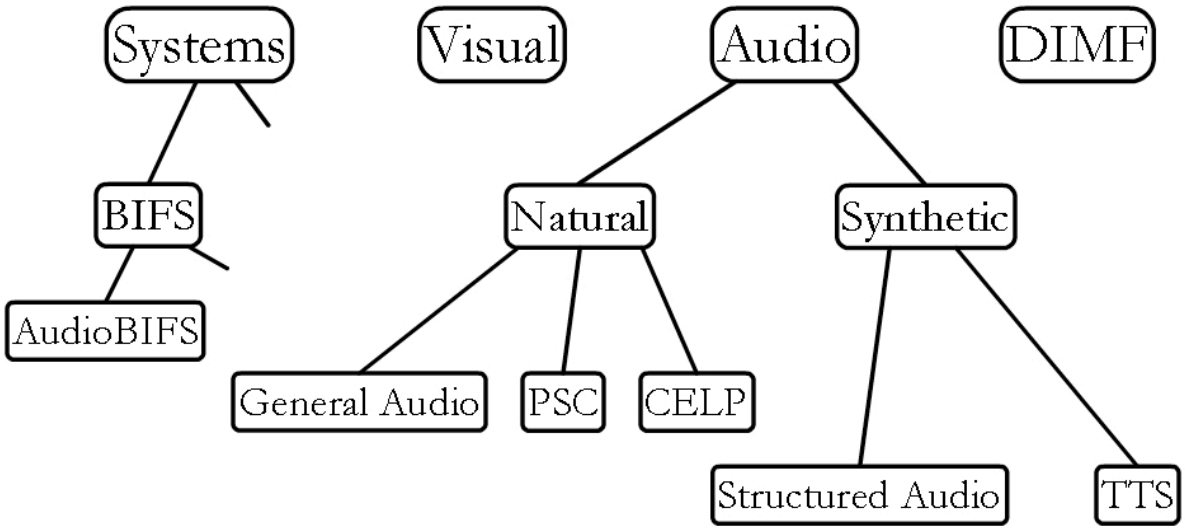
Perceptual compression : this lossy coding exploits the perceptual redundancies of objects. A psychoacoustic model detects the informations that may be omitted with no perceptible change for humans.

Process redundancies elimination : the physical creation of the object is approximated using a model of the generator. Then, linear-predictive coding permits to represent the object with only a few parameters. At the receiving side of the transmission, a synthesis algorithm using the same model, is used to reconstruct it from the transmitted parameters.



Structural redundancies : in the organized human-world, very similar sounds or images appear a great number of times. Any occurance of a particular object has the same characteristics, except for small differences. Each object is represented once, while the differences are described by an algorithm and a few parameters.



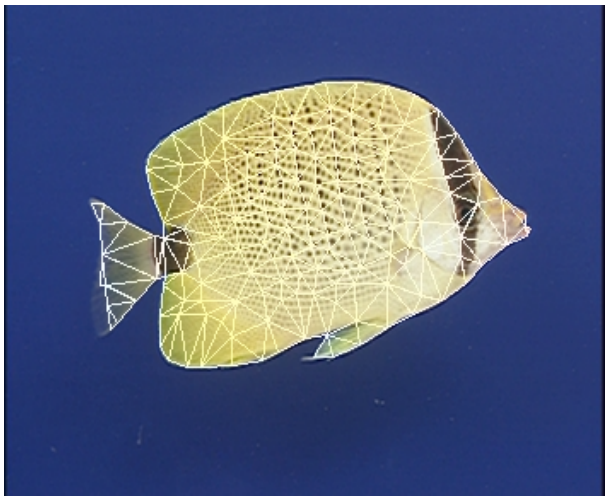


MPEG-4 General Organisation

1.3- MPEG-4 Visual [1]

Representation tools provide standardized technologies for storage, transmission, but also manipulation of textures, images and video data.

MPEG-4 provides *natural video* compression schemes, but also means of describing *synthetic* visual objects. The tools in the standard include parametric description of face and body animations, Mesh coding and view-dependant scalability.



A Mesh is a partition of a region into polygonal patches.
By deforming the mesh, the fish can be animated very efficiently, and be made to 'swim'.

1.4- MPEG-4 Audio [2]

1.4.1- Main Audio coding tools

coding method	GA	CELP	PSC	TTS	SA
average bitrate	16 to 64 kbs	16 to 24 kbs	6 kbs	from 0.2 kbs	2 kbs
applications	broadband (music)	speech	speech	speech synthesis	synthesis and effect processing
origins	MPEG-2 AAC		harmonic vector excitation		Music-N

General Audio (GA)
Based on the MPEG-2 Advanced Audio Coding (AAC) method, with extensions that permit better scalability and better performance at low bit rates.

Codebook Excited Linear Prediction (CELP)
Coding of speech signals.

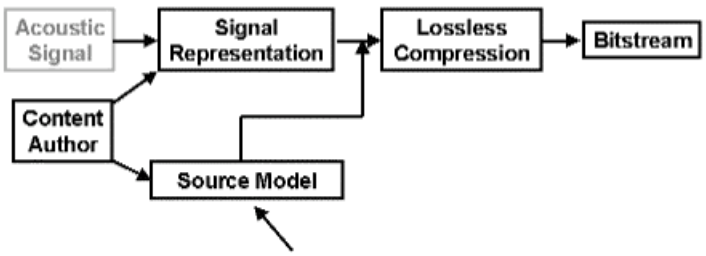
Parametric Speech Coder
Uses Harmonic Vector eXcitation Coding (HVXC).

1.4.2- Synthetic Audio Coders [4]

In coding synthetic sound, perceptual models are not used; rather, parametric models are used to transmit sound *descriptions*. The descriptions are received at the decoding terminal and converted into sound through real-time sound *synthesis*.

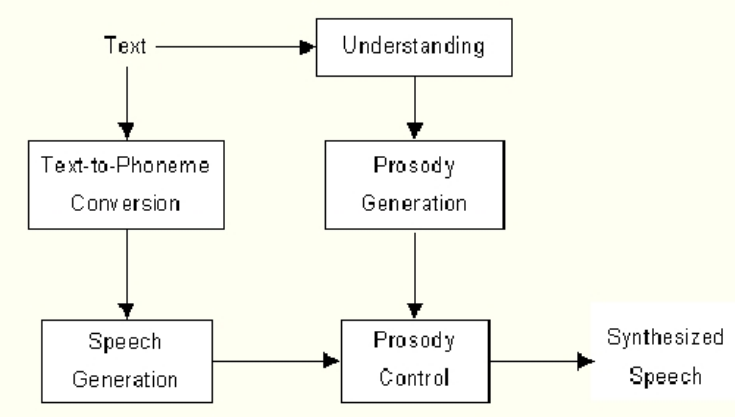
Structured Audio (SA)
Algorithmic Structured Audio techniques use a general-purpose software-synthesis language to represent sound for transmission. SA may eliminate process redundancies by transmitting a model of the source, used by the decoder to produce the final sound. But it also eliminates structural redundancies by transmitting only once the sounds that are assumed to be similar, and a set of parameters to differenciate them.

Structured Audio	Synthesis language / Programming language	enables	Source Modelisation / Structure Description	Content Description



The model may be a synthesis algorithm, or the definition of a perceptual coder. Thus, although SA is well adapted to coding synthetic sound, it is also an innovative tool for coding natural sound. In MPEG-4, the distinction between decompression and synthesis is getting blurred.

Text-To-Speech (TTS)
It should be considered as a restrictive particular case of Structured Audio. TTS standardizes a bitstream format, and specifies a joint coding-method for phonemic information and facial animation parameters. No method of speech synthesis is normalized.



Immediate applications are email reading, voice response systems or visually handicaped devices. Integration in MPEG-4 should also help the development of interactive applications: on-demand storytelling, motion picture dubbing (lip shape following), and "talking head" synthetic videoconferencing.

Other existing Speech Synthesis systems: DecTalk(83), CNET's PSOLA(92) and ATR's CHATR(96). Current research: synthesis of a particular speaker's voice, multi-language support, naturalness.

1.4.3- Generalized Audio Coding

The overall philosophy of MPEG-4 Audio is to divide a single soundtrack into components, each being coded using a different model, to optimise the coding process. Indeed, a *lossy perceptual coder* is actually a *lossless coder* for a small set of sounds (Scheirer and Kim, 1999[5]).

As an extension of this idea, a coder may transmit its model along with the data : no fixed model would be used for the coding, and the content-provider would decide what model is the best for coding his sounds.

Although it was not originally developped for this purpose, MPEG-4 Structured Audio may be used as a *model-description framework*. Scheirer and Kim [5] prooved it is a universal model capable of emulating any other sound model, and termed it a *generalized audio coder*.

1.5- MPEG-4 Systems [6]

1.5.1- Authoring Tools for Scene Description

A Scene description tool should provide means to:

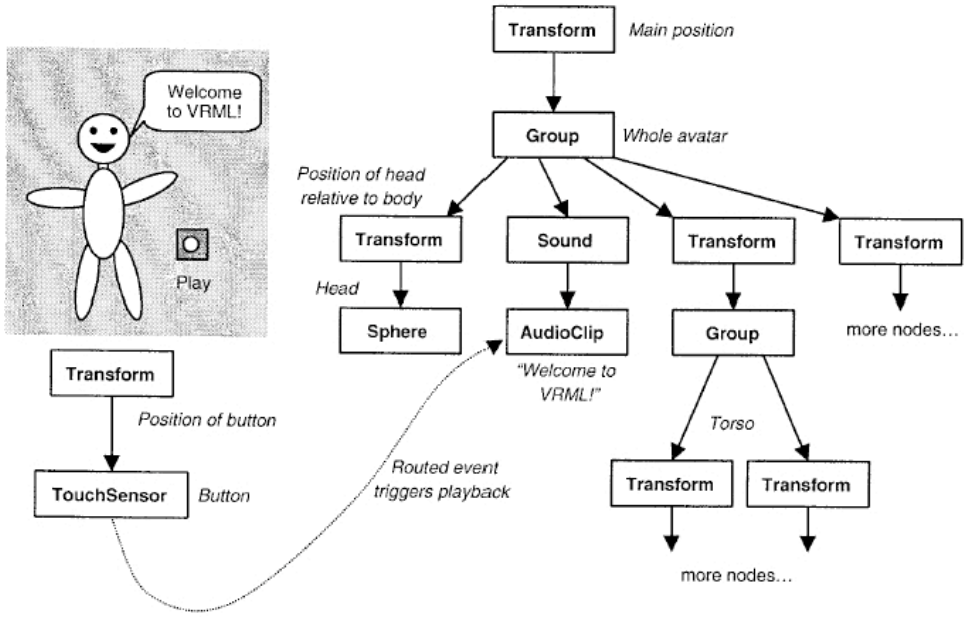
- Place media objects anywhere in a given coordinate system;
- Apply transforms to change the appearance of a media object;
- Group primitive media objects in order to form compound media objects;
- Apply streamed data to media objects, in order to modify their attributes
- Change, interactively, the user's viewing and listening points.

The first powerful file format capable of describing interactive worlds is the *Virtual Reality Modeling Language* (VRML). A standardization was achieved with VRML 1.0.

VRML 97 is capable of representing static and animated dynamic 3D and multimedia objects [7]. Designed to be used on the Internet with browsers available on different platforms, it also permits the embedding of sounds and movies.

In VRML, scenes are defined through a set of *nodes*, organized in a *scene graph*.

A node may represent :
- an object
- a property
- a transformation that may affect certain nodes (rotation, scaling).



Messages may be sent between nodes along *routes*.

The scene description framework used in MPEG-4, termed *BInary Format for Scenes* (BIFS) is based largely on VRML [6]. But VRML only implements 2 audio nodes named Sound an AudioClip, whereas BIFS provides mulitichannel, direct intensity mixing, and many more capabilities.

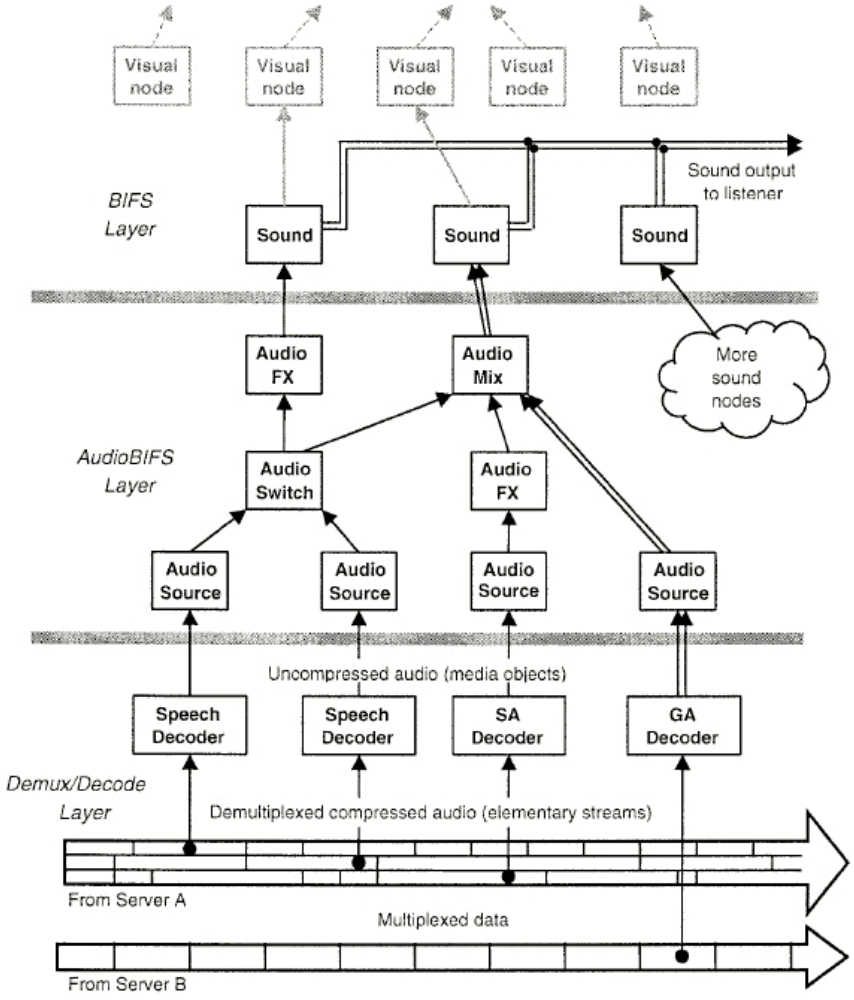
A media object may be completely described within the BIFS information, or may also require elementary stream data from one or more audiovisual objects.

Any kind of natural or synthetic object from the MPEG-4 stream may be included in the scene. This capability of integrating in the same scene *natural* and *abstract* objects or treatments is termed *Synthetic/Natural Hybrid Coding* (SNHC).

1.5.2- Integration of Audio Objects

The AudioBIFS scheme [8] is a sub-part of BIFS, implementing a set of AudioBIFS specific nodes. It attaches sound to scenes through an *Audio Subgraph*.

Sound is introduced in the subgraph through the AudioSource node, and is played back to the listener through the Sound node.



Some other nodes are placed between AudioSource and Sound. They may transform the sounds or modify the objects parameters.

A special node named TermCap allows the scene graph to detect what are the hardware and performance properties of the terminal, or the playback conditions. These informations aren't necessarily normative.

The scene graph could specify that a compressor should be applied in a noisy environment.

1.5.3- Technical viewpoint

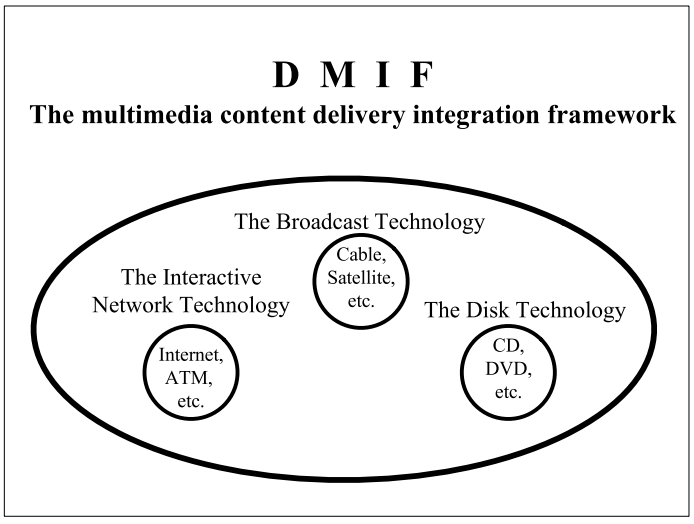
Media Objects created with Audio or Video Tools, as well as scene description commands, are transmitted separately as *elementary streams*. Synchronization is achieved through *time stamping* of individual *access units* within elementary streams. Time stamping is independant of the media type.

As the decoding process uses widely the client-side hardware, MPEG-4 includes a simulation tool, used to determine the decoder level of performance needed. *Levels* of the standard are set with regard to computational-complexity measured with this tool.

In order to permit partial implementations of the standard, MPEG has defined BIFS *profiles*: Complete (audio+visual...), Complete 2D, Simple 2D (equivalent to MPEG2), Audio (for radios...).

1.6- MPEG-4 DMIF

The *Delivery Multimedia Integration Framework* (DMIF) specifies the first steps of the multiplexing process and an interface with generic delivery technologies [1].



2. Structured Audio

2.1- Software Synthesis Languages

Historically, the first solutions that were thought of to produce sound on a computer were using software-synthesis.

As early as 1969, Max Mathews proposed to decompose synthesis algorithms in basic primitives, to describe organized sounds as a network of interacting oscillators and envelope functions. He gave birth to the “Music N” languages, that inspired several other similar languages, each of them featuring a particular way of modeling sound-synthesis, adapted to a particular usage:

- Csound (Vercoe 1995)
- SuperCollider (McCartney 1996) employs an *object-oriented* model
- Nyquist (Dannenberg 1997) employs a *functionnal programming* model.

2.2- Structured Audio principles

The idea of transmitting descriptive algorithms to reduce bitrates was suggested by Julius O’Smith in 1991. In 1996, NetSound (Casey and Smaragdis) was constructed on this concept, using Csound as a synthesis engine, and allowing transmission on the Internet. In 1998, Vercoe, Gardner and Scheirer termed this idea Structured Audio.

Structured Audio provides capabilities that would not be possible through natural sound, making it well adapted to Interactive Audio Applications.

- Low bitrate
- Dynamic control of individual sounds
- Accurate sound description : not limited to one method of synthesis or to the wavetable included in a Sound-card
- Description of Effect Processors
- *Downloadable* decoder (flexibility).

2.3- The MPEG-4 Structured Audio Languages [2,9,10]

The language provides general access to any method of structuring sound. It is capable of emulating any type of sound synthesis.

It features a distinction between an Orchestra Language (SAOL), describing the processing tasks, and a Score Language (SASL), giving their temporal succession.

This *declarative unit-generator-based* language has been developped especially for an MPEG-4 Audio usage. Its syntax has similarities with the C syntax, although SAOL adds features from other general-purpose software-synthesis languages.

Indeed, SAOL is based on a fully recursive expression grammar, that permits an easy description of the temporal structure of sound. Management of the 3 time scales of the musical sound (notes succession, envelopes and samples) is achieved by the use of 3 types of *pass*:

- instantiation-pass: occurs at the beginning of each note
- control-pass, more frequent (+- 100Hz): description of sound modulations
- audio-pass, for the calculation of each sample.

There are 3 corresponding types of variables, whose content is computed at different rates: i-rate, k-rate and a-rate.

Modern languages such as Nyquist and SuperCollider have adopted the Block-based Processing. In this mechanism, each line is evaluated for an entire block of samples before the next line is evaluated. This permits an efficient calculation of orchestras, but keeping strictly this idea, some algorithms cannot be constructed.
SAOL implements a sample-by-sample model, used earlier. The authors assume that it is possible to develop SAOL compilers that will optimize the execution and automatically use the block processing method when the orchestra permits it.

SAOL *instruments* are some kind of routines, describing a synthesis procedure. A sound is produced by the instantiation of an instrument, i.e. the initialization of its parameters and its declaration to the *scheduler*.

Values may be passed through SAOL instruments by *global* variables. The instruments' behaviour may also be controlled by the SASL using the import function.

Any compliant SAOL implementation contains a full set of *Core Opcodes*, that are sound-treatment oriented fonctions used in the language: Math functions, Pitch converters, Signal Generators, Filters, Effects...

To these default unit-generators the user may add some *user-defined unit generators*. Once the new opcode defined, its names becomes a standard name that is used exactly the same as any other unit-generator. SAOL being modular, any user-defined opcode may be re-used easily without changing anything in the program.

A MIDI implementation makes it possible to control a SA orchestra through this interface or the *MIDI file* format. Although, MIDI is clearly designed to be a *real-time* transmission language, whereas SASL acts more like a score, defining a note with its duration.

Other existing control formats: ZIPI (1994) and OpenSound Control (1997).

2.4- SAOL as a processing tool for AudioBIFS

According to the principle of transmitting audio in a structured way, *effect processors* affecting the audio objects present in the scene should also be considered as objects. SAOL's elegant integration among other MPEG-4 Tools permits to describe these processors as well.

The implementation of an easy routing system permits to send signals to SAOL instruments the same natural way as in a usual audio system.

Processing algorithms may be written with the language, possibly with the help of core opcodes like spectral analysers, delays or filters. It was prooved [9] that any algorithm for digital sound manipulation can be written with SAOL. Using the same tools, algorithms may be used to describe any type of audio-coder, capable of eliminating any type of entropic, perceptual, process or structural redundancies.

2.5- Structured Audio Sample-Bank Format

SASBF, designed in collaboration with the MIDI Manufacturers Association, allows banks of sampled PCM sounds for use in wavetable synthesis to be transmitted. Wavetable datas may also be used for describing resonance filters and envelopes to be applied to the samples using a SAOL-described algorithm.

2.6- Streaming format and Decoding process [2,4]

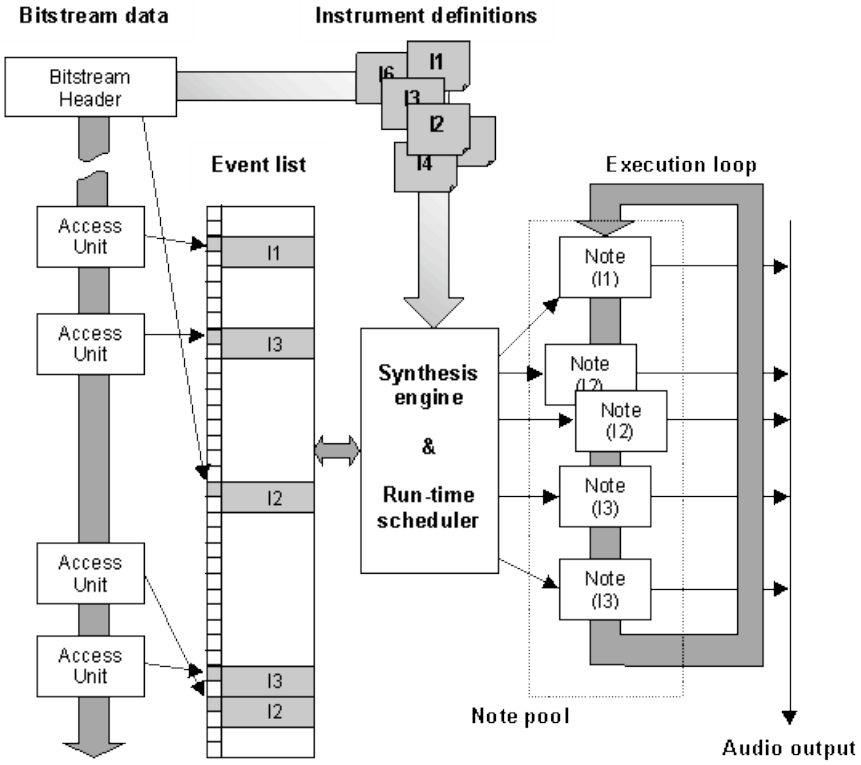
The bitstream is divided in a *decoder configuration header*, in which instruments and auxiliary data are defined, and *bitstream access units* containing compressed data. Score data may be included in the bitstream or in the header, what permits a better ressource allocation and a real time tempo control during playback.

The instructions transmitted are tokenized. To get back to a plain-text human-readable representation, an optional symbol table may be included in the header.

Decoding consists in de-multiplexing the bitstream by finding the corresponding Audio access units, recreating SAOL instruments and SASL score, and make them available for the Run-time Scheduler in an event-list.

The *scheduler semantics* describes how control events, SASL scores, MIDI data and the SAOL orchestra interact to produce sound.

The Run-time Scheduler reads event lines and interacts with the synthesis engine to modify instruments parameters, keep track of note instances in a note pool and execute the corresponding algorithms at each time step.



3. MPEG-4 Dynamic Tools

In the first part of this paper, was presented a way to describe sound scenes including any type of media objects. In interactive applications, the user should be able to react to the events so that the scene progress may be influenced by his actions. Let's now examine how the author may introduce dynamic responses affecting the media objects.

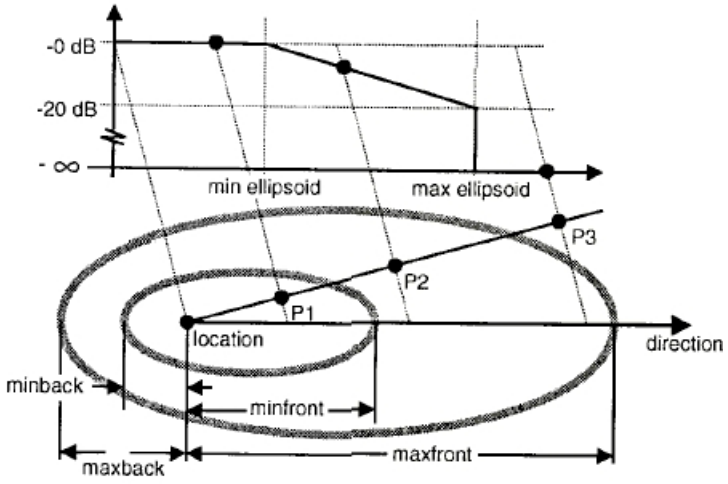
3.1- General-purpose AudioBIFS nodes [8]

Sound in the scene may be spatialized and physically modeled according to the parameters of a virtual world, to re-create the true response of an existing acoustical situation. It may also be post-processed with arbitrary filters, reverberators or other digital effects, in order to express the virtual designer's creativity. These 2 modes are called *virtual reality* compositing and *abstract effects* compositing.

First version of AudioBIFS implements Core nodes for audio integration, abstract effects compositing and virtual spaces description.

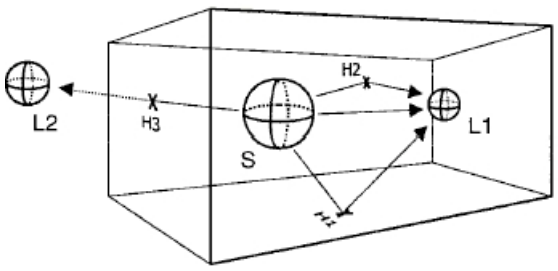
Node	Purpose
AudioBuffer	Interactively caches short sound samples
AudioClip	Insert an audio clip to scene
AudioDelay	Insert delay to sound
AudioMix	Mix sounds
AudioSource	Define audio source input to scene
AudioFX	Attach structured audio objects to sound
AudioSwitch	Switching of audio sources in scene
Group, Group2D	Grouping of nodes and subtrees in a scene
ListeningPoint	Define listening point in a scene
Sound, Sound2D	Define properties of sound

The Sound node is similar to the one of VRML. It uses the same ellipsoïdal sound-attenuation model for positionning. This model is not normative: it may be achived by amplitude panning as well as HRTFs.



An advanced version of BIFS adds 3 audio nodes for virtual acoustics physical modeling. A precise model may be built by determining successively the *source* and *environment* acoustical properties, and the virtual listener's positionning.

Name	Function
AcousticScene	Group sounds together in an auralization process.
AcousticalMaterial	Specify reflection and transmission impulse responses for an object in a scene.
DirectiveSound	Specify frequency-dependent directivity modeling for a sound.



Room geometry (walls disposition, their characteristics...) as well as object attenuation, air transmission or directivity of instruments may be specified.

Concerning the perceptual approach, a 4th node is also implemented in Advanced AudioBIFS. In the *perceptual parameters* node, sound description is independant of visual "reality". Subjective testing of acoustical room qualities allow high level perceptual parameters to be defined: brilliance, heavyness, liveness, envelopment, reverberance... It may be used when no link is needed between sound and image, or when an imaginary effect is intended.

Parameter name	Relation to room acoustic response
sourcePresence	Energy of the direct sound and early room effect.
sourceWarmth and sourceBrilliance	Variation of early energy with frequency.
roomPresence	Energy of late room effect.
runningReverberance	Early decay time.
envelopment	Energy of early room effect relative to direct sound.
lateReverberance	Late decay time.
heavyness and liveness	Variation of decay time with frequency.
omniDirectivity	Diffuse field spectrum of the sound source.
directFilterGains	Filter definition for the sound arriving directly from the source to the listener.
inputFilterGains	Filter definition for the sound fed to the system. May be used, e.g., for producing occlusion effects.
refDistance	Reference distance at which these parameters apply; For distances other than this a new sourcePresence value is computed as a function of the actual distance, the reference distance, and the sourcePresence value at the reference distance.
modalDensity	theoretical density of modes in the response in the frequency domain.

3.2- Interactivity with AudioBIFS

In BIFS like in VRML, nodes may communicate through messages. Thus, if a sensor node detects an action from the user, it may send a parameter to another node.

The user interface permits to manipulate the *exposed fields* of some nodes: for instance the matrix field of an AudioMix node, the params field of an AudioFX node, or the start, stop, pause and rewind fields of AudioSource. This node also permits pitch and speed changes for decoders that allow this fonction (SA and HVXC).

The Advanced AudioBIFS nodes also provide great dynamic fonctionalities, as any of their parameters are exposed fields.

Sound presence may increase while the user moves through the scene towards the saxophone soloist.

The scene-graph itself may be dynamically modified by a special stream called *BIFS update*.

3.3- The AudioFX node : SAOL integration

To extend AudioBIFS capabilities, BIFS may take some help from SAOL for effects definitions. As explained in section 2.4, any digital-processing algorithm may be written in SAOL, possibly with the help of Core opcodes or wavetables transmitted with SASBF.

The AudioFX node makes the link between the SA orchestra and the scene graph. It sends sounds and parameters to the SAOL instruments used to transform the audio scene. Moreover, time varying effects can be controlled using SASL, handled exactly the same way as the controls over an instrument instantiation.

Employing *user-defined effect-processing opcodes* guarantees a precise control over the desired algorithm : treatment will sound exactly the same, no matter what terminal achieves it.

Exposed fields of AudioFX are children (sets affected nodes), orch (specifies the SAOL orchestra), score (specifies a SASL script), and params (interactive control of effects from the scene graph : parameters sent to the orchestra).

3.4- Structured Audio interactive usage

Extending the use of AudioFX, one can imagine using BIFS as a remote for any type of SAOL instrument.

One great advantage of representing sounds on a Structured form is that it permits to handle them as parametric objects. Consequently, a great variety of parameters will be available for a Virtual World Designer to build freely a more creative sound-image, and convincing auditive and visual parralels within a multimedia product. The final user will have more possibilities to interact with the Audio document.

Due to SAOL's programming language basis, algorithmic or stochastic compositions, as well as dynamic response to interactions can be written in SAOL with no need of an external language interface.

The intimidated bassist may change the way he is playing on the listener's approach. Importing a variable giving the listener's position in a SAOL instrument, the author may allow the bass line to be played with more dynamics or may introduce mistakes in the chords.

3.4- Perspectives and other systems

One shouldn't forget that MPEG-4 is still only a normalization. An important criteria to classify the few implementations that have already been developed is the fact to be of *real-time* execution.

Most of the software available right now is demo software, like IM-1, demonstrating the capabilities of the *MPEG-4 Systems* norm, and giving directions for future implementations.

Advances of the current research in this domain are found on the Internet.

About Structured Audio, the Dutch web-site www.saol.net presents an overview of existing implementations and tries to gather sets of SAOL User-defined Opcodes, as well as instruments and music.

The Machine Listening Group at the MIT, where Structured Audio was originated, diffuses *saolc*, that is the reference software for SAOL. It produces .wav files, so that it is possible quite easily to ear the result right now.

At UC Berkeley, John Lazzaro and John Wawrzynek developped a non-real time implementation called *sfront*. It permits to translate SA datas into a C program that generates audio.

The Structured Audio Open Group (SAOG) has a first project called SAINT, also a Structured Audio INTerpreter [11].

Bert Schiettecatte, from the Vrije University in Brussels, is currently developping Qorchestra [12], a visual authoring tool for SA, using the C++ Qt class library for writing the Graphical User Interface.

With MPEG-4, we discovered an innovative tool perfectly adapted to future multimedia interactive applications, driving it to a Virtual Reality and Internet usage. One good reason for this is the fact that MPEG-4 has an object-oriented basis. Indeed, partial transformations of a scene are only possible if its content is represented as separate objects. Moreover, the flexibility of the integrated SAOL programming language pushes away the limits of the normalization.

Some of the techniques described in this paper are not new and exist under different forms. Many of these were already disseminated in the Industrial world. The point is that MPEG-4 is a standardization, and that this document describes the way these techniques will be *used* and *combined* in a near future.

References

Most references may be found on the internet. Good starting points would be http://sound.media.mit.edu/~eds/papers.html and http://www.saol.net/.

[1] Overview of the MPEG-4 standard, ISO/IEC JTC1/SC29/WG11 Document N2725, March 1999

[2] MPEG-4 Final Committee Draft – Part 3: Audio, ISO/IEC SC29/WG11 Document N2503, March 1999 (section 5: Structured Audio)

[3] MPEG-4 Structured Audio technical overview, http://sound.media.mit.edu/mpeg4/sa-tech.html , 1996

[4] Scheirer, Eric D., and Youngjik Lee and Jae-Woo Yang (in press),
Synthetic and SNHC Audio in MPEG-4.
Signal Processing: Image Communication **15**:4-5 (Jan. 2000), pp. 445-461.

[5] Scheirer, Eric D., and Youngmoo E. Kim
Generalized Audio Coding with MPEG-4 Structured Audio.

AES 17th International Conference on High-Quality Audio Coding, Florence IT, Sept. 1999.

[6] MPEG-4 Final Comittee Draft – Part 1: Systems, ISO/IEC SC29/WG11 Document N2201, May 1998

[7] Virtual Reality Modeling Language (VRML 97) ISO/IEC JTC/SC24 IS 14472-1, 1997

[8] Scheirer, Eric D., and Riitta Väänänen and Jyri Huopaniemi (1999).
AudioBIFS: Describing Audio Scenes with the MPEG-4 Multimedia Standard.
IEEE Transactions on Multimedia **1**:3 (Sept. 1999), pp. 237-250.

[9] Scheirer, Eric D., and Barry L. Vercoe (1999).
SAOL: The MPEG-4 Structured Audio Orchestra Language.
Computer Music Journal **23**:2 (Summer 1999), pp 31-51.

[10] Lazzaro, John and Wawrzynek, John: "The MPEG-4 Structured Audio Book", http://www.cs.berkeley.edu/~lazzaro/sa/book/index.html,1999

[11] The Structured Audio Open Group, http://lsiwww.epfl.ch/saog

[12] Schiettecatte, Bert, A visual authoring tool for SAOL – Software design (draft), January 2000