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

ISO/IEC JTC1/SC29/WG11 MPEG2019/M50899
October 2019, Geneva, CH

Source	**Telecom Paris**
Status	**For discussion during MPEG #128**
Title	Towards GPAC 1.0
Author	Jean Le Feuvre

1 Introduction

We are pleased to inform the MPEG Systems subgroup that the GPAC team has released the development branch of the future 1.0 version of the GPAC open-source framework. Given the potential interest of MPEG Systems experts in GPAC, we would like to present an overview of the new features that we hope will be of interest.

2 Features

2.1 *What's really new?*

Without going into the details, what is really new in the upcoming version is a complete rewrite of the underlying stream management system used in GPAC, dating from 2002 (and mostly inspired by the MPEG-4 works)!
The new architecture is based on a highly modular approach, where media processing blocks can be reused in various media pipelines for many purposes. For MPEG terminology, think of it as "Reconfigurable Systems Processing", not very far from NBMP.
Do not worry, we have hidden most of the complexity of the block connection logic from the end user through built-in dynamic graph resolution. And of course, current applications (MP4Box MP4Client/Osmo4) have been ported to the new framework in a backward-compatible way, guaranteeing a seamless transition for end-users.

For more details on the rationale for this re-architecture and its impact on your workflows, check our wiki.

2.2 *Data IO*

GPAC now supports input and output data streams other than files: pipes (IO), UDP and TCP sockets (IO) and HTTP(S) (input only for now).
This implies you no longer need to go through local files to create your MP4 files in GPAC, you can now take inputs from any program outputting to pipes or socket.
Similarly, you can also export most GPAC outputs to pipes or sockets.

In the process, we also added a simple RTP/RTSP server to GPAC.
For more info, see:

https://github.com/gpac/gpac/wiki/pipes
https://github.com/gpac/gpac/wiki/sockets

2.3 Encoder support

That's right, we finally have encoders in GPAC! We are currently based on FFMPEG but we are planning support to other encoders as well.
This means that you can now encode and dash a source in one pass:

```
MP4Box -dash 2000 -profile live -out session.mpd source.mp4:@@enc:c=avc:fintra=2
```

The above command will invoke an encoding filter chain in AVC|H264 format with a forced intra period of 2 seconds.

For more info, see:
https://github.com/gpac/gpac/wiki/encoding
https://github.com/gpac/gpac/wiki/mp4box-filters

2.4 Raw formats support

GPAC can now extract your audio and video streams to raw PCM or RGB/YUV formats, including format conversion. It can also use such formats as input to many operations, including storage in ISOBMFF/QT.
We also added various filters for raw formats, allowing you to crop/flip videos and rewind audio and video sources.
We are planning to add support for FFMPEG libavfilter for GPAC 1.0.

For more information, see
https://github.com/gpac/gpac/wiki/raw-formats

2.5 Graphics composition

As some of you may know, GPAC started as an MPEG-4 Systems implementation and includes advanced support for 2D and 3D graphics rendering (BIFS/SVG/LASeR/VRML/X3D) in its media player.
GPAC 1.0 finally gives you access to all these tools outside of the media player. This means that you can now use the 2D and 3D graphics engine in a regular encoding or streaming session.
When working with 2D graphics, you now even have the possibility to render directly over the input frame, in RGB or YUV space !

For more info, check:
https://github.com/gpac/gpac/wiki/olay-composition

2.6 Distributed Processing

GPAC being no longer tied to local files, we added an internal format used to serialize all streams to and from pipes and sockets. This format represents the exact state of the media session, and is not subject to the limitations of a given container format or transmission protocol (supported codecs, meta-data signaling).

This allows distributed processing of a media session in two or more sessions, handling data transfer through pipes or sockets (UDP still experimental).

The internal format supports AES-128 CBC encryption (full or pattern-mode) for the transfer of data, and is overall as costly as ISOBMFF encapsulation.

2.7 *Scripting*

We have replaced the very old SpiderMonkey Javascript engine with the brand new QuickJS engine giving full support of ECMAScript 2020 !
With this in mind, we have added binding to the core streaming engine to let you develop your own filters in JavaScript. The JS API currently supports all filters functionality and XMLHttpRequest.
This means you can now write your own DASH client or segmenter in JavaScript :)

This is a work in progress, we are currently adding a 2D Canvas-like API ("buildbot-qjs" branch, still under testing but quite stable) and a 3D WebGL API in the near future.

2.8 *User Friendliness*

We have not just been busy coding!

- Improved documentation:
  - automatic generation of documentation to ensure up-to-date man pages and public doc
  - better doxygen for libgpac : https://doxygen.gpac.io/
  - HowTo-s (work in progress): https://github.com/gpac/gpac/wiki/Howtos
- Improved testing:
  - test suite and coverage: we now cover 90%+ of the functions in GPAC, 75%+ LOC
  - fuzzbot: we permanently fuzz content against the latest builds (work in progress)
  - publicly available for each build:
    - build results: https://buildbot.gpac.io/#/
    - tests results: https://tests.gpac.io/
    - binaries: https://gpac.wp.imt.fr/downloads/gpac-filters-branch/

2.9 *Other goodies*

You may also be interested in the new support for:
- on the fly encryption and decryption (CENC, ISMA)
- reduced disk space requirement modes for DASH onDemand and ISOBMFF/QT Fast-Start
- HEVC tile splitting and merging with slice header rewrite
- advanced stream concatenation (codecA->codecB->RAW->codecC)
- inspection and media analyzing on any source
- more HEIF tools
- HLS generation, dual HLS/DASH generation
- DASH segment encryption
- live grabbers (webcam, microphone) through FFMPEG
- and all the things supported in GPAC 0.8.0, including ATSC3/ROUTE demux and AV1/VP9 tools!

3 Conclusion

The code is currently hosted as the "filters" branch on github, and its version is 0.9.0. We plan on merging before the end of the year, and release 1.0 in early 2020.

We encourage MPEG Systems experts to play with this new version of GPAC and give us feedback and feature requests. We sincerely hope this new version can be useful in many contexts, from standardization, R&D projects, academic/students works to industrial deployments.

For more information, see:
- http://gpac.io/
- https://github.com/gpac/gpac/
- https://www.gpac-licensing.com/