124th MPEG Macau SAR, China, 8 - 12 October 2018, Meeting Report

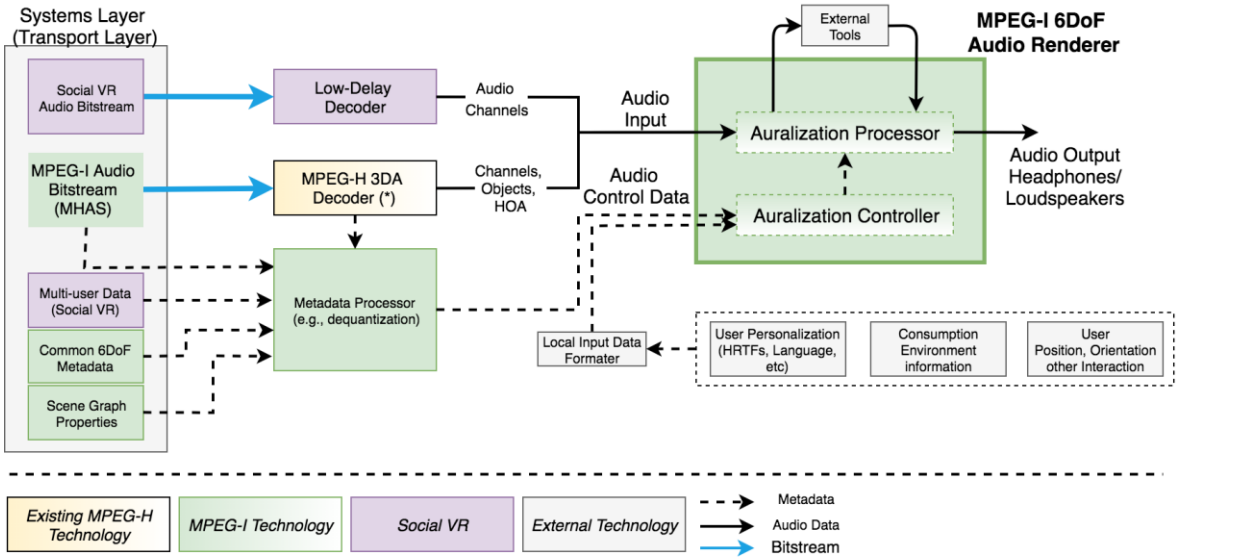
Panos Kudumakis
qMedia, Queen Mary University of London

Contents

1 Draft MPEG-I Audio Requirements 1
2 Internet of Media Things (IoMT) - Architecture 4
3 Use Cases and Requirements for Compressed Representation of Neural Networks 5
4 Network-based Media Processing 14
5 Smart Contracts for Fair Trade of Music 15

1 Draft MPEG-I Audio Requirements

This document provides the draft of MPEG-I Audio Requirements as currently discussed in the Audio Subgroup.



*(*) **MPEG-H 3DADecoder** is defined as the core decoder of the MPEG-H 3D Audio Low Complexity (LC) Profile receiving as input in the form of an MHAS stream and providing as output decoded PCM audio (channels, objects HOA) together with all metadata available in the MHAS packets.*

Figure 1 - MPEG-I Audio Reference Architecture.

P – "Primary" Requirements
S – "Secondary" Requirements
SYS- "System" Requirements

Shall support – The specification shall make this happen
Shall enable – The specification shall permit other agents to make this happen

General Requirements

1. [P] The specification shall support user 6DoF (Degrees of Freedom) so that the user perceives an experience consistent with user's movement in the environment (e.g., low and non-perceivable motion-to-sound latency).
2. [P] The specification shall support efficient representation and compression of media and metadata.
   2.1. [P] Media coding shall be done according to the MPEG-H 3D Audio Low Complexity Profile (including 3D Audio metadata). The specification shall support any combination of channel-based, object-based and HOA content types.

Note: Multiple HOA streams (e.g., sampled at different locations) may be supported simultaneously in one Audio Scene.

2.2. [P]The specification shall support Loudness and Dynamic Range Control using MPEG-H 3D Audio tools.

2.3. [P] The specification shall support additional metadata as needed to support user 6DoF.

2.4. [P] The specification shall support delivery of the audio elements in multiple audio streams.

Note: This may already be fulfilled by MPEG-H 3D Audio.

3. [P] The specification shall support rendering of the audio scene, including its acoustic elements and acoustic environments, resulting in a user experience consistent with the scene.

3.1. [SYS] Audio elements shall be rendered consistently with their corresponding visual elements, if such visual elements exist.

3.2. [S] The specification shall support signalling of audio elements that have a fixed position relative to the user orientation and position (e.g., non-diegetic content).

Note: This may already be fulfilled by MPEG-H 3D Audio.

3.3. [S] The specification shall support earcons.

Note: This may already be fulfilled by MPEG-H 3D Audio.

4. [P] The specification shall support dynamic inclusion of audio elements in a sub-scene based on their relevance, e.g., audibility relative to the user location, orientation, direction and speed of movement or any other audio scene change.

4.1. [P] The specification shall support metadata to allow fetching of relevant sub-scenes, e.g., depending on the user location, orientation or direction and speed of movement.

Note: *A complete audio scene may be divided into a number of audio sub-scenes, defined as a set of audio elements, acoustic elements and acoustic environments. Each audio sub-scene could be created statically or dynamically.*

Requirements on Audio Renderer

5. [P] The specification shall support metadata describing the audio scene.

6. [P] The specification shall support metadata for controlling and restricting audio scene parameters.

Note: This may already be fulfilled by MPEG-H 3D Audio.

7. [S] The specification shall support control (e.g., via metadata or interface) of the audio rendering parameters (e.g., consumption space, player capabilities, etc.).

8. [P] The specification shall support random-access in time (e.g. every 0.5 seconds) and space (e.g. jump within a sub-scene or to a new sub-scene).

9. [S] The specification shall support metadata for enabling transition effects on audio rendering during user jumps between two different listener positions in the audio scene (e.g., fade-out fade-in).

10. [S] The specification shall support metadata for enabling audio zooming (e.g., adjustment of prominence, dialog enhancement, simulation of depth-of-field effect, etc.).

Note: This may already be fulfilled by MPEG-H 3D Audio.

11. [P] The specification shall support 3D spatial extent for audio objects, supported by metadata.

12. [P] The specification shall support rendering of the radiation pattern of audio objects, supported by metadata.

13. [P] The specification shall support occlusion of audio elements, supported by metadata.

14. [S] The specification shall support metadata for including locally captured audio (e.g., user's own voice) in the audio scene.

15. [P] The specification shall support accurate 3D spatial localization of audio objects (sound sources). Differences in localization are with respect to what is perceivable.

Interfaces and extensions

16. [P] The specification shall support input interfaces for changing the audio scene.

    Note: MPEG-H 3D Audio already provides interfaces enabling basic functionality that could be enhanced for MPEG-I Audio.

17. [P] The specification shall enable future extension of the rendering functionality by means of extension mechanisms (e.g., interfaces to external rendering tools, extension payloads, reserved bit fields etc.).

18. [P] The specification shall enable support for personal HRTF into the audio renderer, including an interface for providing these filters.

Presentation Modes

19. [P] The specification shall support 6DoF head-tracked binaural rendering for headphone reproduction.

20. [P] The specification shall support 6DoF head-tracked rendering to loudspeakers (e.g. to immersive configurations such as 7.1 + 4H).

    Note: The user explores the scene by moving in the listening area and the audio is rendered accordingly over loudspeakers based on his position.

21. [P] The specification shall support 6DoF rendering to loudspeakers for the use case that the user's consumption position is fixed, while the virtual position changes.

    Note: The user is located in the sweet-spot and navigates the scene for example using a joystick.

22. [S] The specification shall support rendering to a combination of 6DoF head-tracked binaural headphones reproduction and loudspeaker reproduction.

Social VR

23. [P] The specification shall support rendering of speech and audio from other users in the virtual environment. The speech and audio may be immersive.

    23.1. [P] The specification shall support low-latency conversation between users within a given virtual environment.

    23.2. [S] The specification shall support low-latency conversation between a user within the given virtual environment and a user outside the given virtual environment.

    23.3. [SYS] The specification shall enable synchronization of audio and video of users and the scene *[Comment: Needs refinement in the joint session with systems and visual groups]*.

    23.4. [P] The specification shall support metadata specifying restrictions and recommendations for rendering of speech/audio from the other users (e.g. on placement and sound level).

Interoperability between 3DoF and 6DoF platforms

24. [P] The specification shall support decoding and presentation of MPEG-H 3D Audio Low Complexity Profile content on an MPEG-I 6DoF platform with an experience as with an MPEG-H 3D Audio Low Complexity Profile decoder.

25. [P] The specification shall enable consumption of MPEG-I Audio content on MPEG-H 3D Audio Low Complexity Profile (3DoF).

26. [P] The specification shall enable consumption of MPEG-I Audio content on MPEG-I Audio platforms with reduced degrees of freedom e.g., 3DoF+, 3DoF, 0DoF platforms.

Output Documents
N18085 - Draft MPEG-I Audio Requirements
N18086 - MPEG-I Audio Test Material
N18087 - Workplan on MPEG-I Audio

2 Internet of Media Things (IoMT) - Architecture

The global IoMT architecture is presented in the Figure below, which identifies a set of interfaces, protocols and associated media-related information representations related to:

- User commands (setup info.) between a system manager and an MThing, cf. Interface 1.
- User commands (setup info.) forwarded by an MThing to another MThing, possibly in a modified form (e.g., subset of 1), cf. Interface 1'.
- Sensed data (raw or processed data) (compressed or semantic extraction) and actuation information, cf. Interface 2.
- Wrapped interface 2 (e.g., for transmission), cf. Interface 2'
- MThing characteristics, discovery, cf. Interface 3.

Furthermore, the mapping of components between SC29 IoMT and SC41 IoT Reference Architecture is straightforward as the interfaces encompassed by the SC29 IoMT Architecture ensure data exchanges between the various domains considered by SC41 IoT Reference Architecture. Specifically, the following mapping is established:

- Interface 1 & 1' of IoMT relate to the User Domain interaction with the things in SC41 IoT Reference Architectures
- Interface 2 & 2' of IoMT relate to the functionalities of the Sensing & Controlling Domain and to Application & Service Domain
- Interfaces 3 of IoMT relates to Resource Access & Interchange Domain.

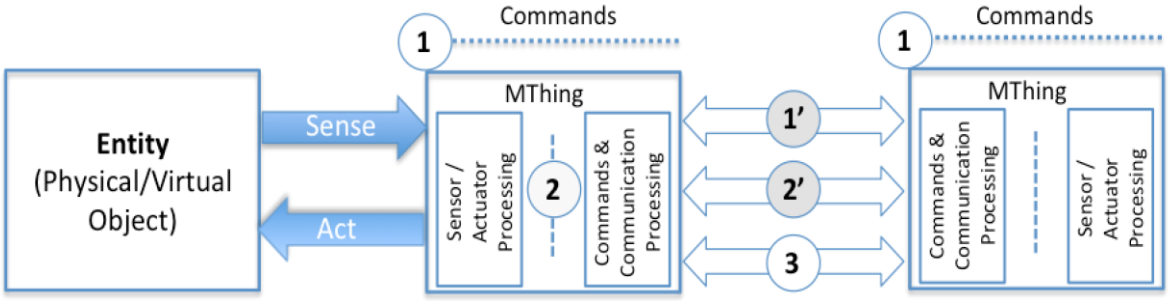


Figure 2 - IoMT Architecture.

Use cases

MPEG identified 26 use-cases for IoMT; they are are structured in five main categories, as follows:

- **Smart spaces: Monitoring and control with network of audio-video cameras**
    - Human tracking with multiple network cameras
    - Automatic title generation
    - Intelligent firefighting with IP surveillance cameras
    - Networked digital signs for customized advertisement
    - Digital signage and second screen use
    - Self-adaptive quality of experience for multimedia applications
    - Ultra wide viewing video composition
    - Face recognition to evoke sensorial actuations
    - Automatic video clip generation by detecting event information
    - Temporal synchronization of multiple videos for creating 360° or multiple view video
    - Intelligent similar contents recommendation using information from IoMT devices
- **Smart spaces: Multi-modal guided navigation**
    - Blind person assistant system
    - Personalized navigation by visual communication
    - Personalized tourist navigation with natural language functionalities
    - Smart identifier: face Recognition on Smart Glasses
    - Smart advertisement: QR code recognition on smart glasses

- **Audio/video smart environments in smart cities**
    - Smart factory: Car maintenance assistance A/V system using smart glasses
    - Smart museum: Augmented visit museum using smart glasses
    - Smart house: Light control, vibrating subtitle, olfaction media content consumption
    - Smart car: Head-light adjustment and speed monitoring to provide automatic volume control
- **Multi-modal smart collaborative health**
    - Increasing patient autonomy by remote control of left-ventricular assisted devices
    - Diabetic coma prevention by monitoring networks of in-body / near body sensors
    - Enhanced physical activity with smart fabrics networks
    - Medical assistance with smart glasses
    - Managing healthcare information for smartglass
- **Blockchain usage for IoMT transactions authentication and monetizing**
    - Reward function in IoMT by using blockchains
    - Content authentication with blockchains

Output Documents
N18045 - Text of ISO/IEC DIS 23093-1 IoMT Architecture
N18090 - DoC on ISO/IEC CD 23093-1 IoMT Part 1 Architecture
N18048 - Text of ISO/IEC DIS 23093-2 IoMT Discovery and Communication API
N18091 - DoC on ISO/IEC CD 23093-2 IoMT Part 2 Discovery and communication API
N18049 - Text of ISO/IEC DIS 23093-3 IoMT Media Data Formats
N18092 - DoC on ISO/IEC CD 23093-3 IoMT Part 3 Media data formats and API

3 Use Cases and Requirements for Compressed Representation of Neural Networks

Artificial neural networks (NNs) have been adopted for a broad range of tasks in multimedia analysis and processing, media coding, data analytics and many other fields. While the underlying technology has been known for decades, the recent success is based on two main factors: (1) the ability to process much larger and complex neural networks (deep neural networks, DNNs) than in the past, and (2) the availability and capacity of large-scale training data sets. These two aspects not only make trained networks powerful, but also mean that they contain a large number of parameters (weights), resulting in quite large sizes of the trained NNs (e.g., several hundred MBs).

In many cases, these NNs are used to replace certain components in a processing workflow, that have previously relied on handcrafted approaches, e.g. for feature extraction or filtering, and NNs have been shown to outperform these handcrafted approaches. The correct execution of the workflow thus requires the deployment of a particular trained network instance, which is then evaluated as part of the workflow (a step called inference). Inference may be performed on a large number of devices (e.g., in consumer applications), thus requiring to transmit the trained NN over a network connection, and possibly these devices may have limitations in terms of processing power and memory (e.g., on mobile devices, smart cameras or at edge nodes).

The NNs used in an application can be improved incrementally (e.g., training on more data, including feedback from validation of results), so that updates of already deployed networks may be necessary. In addition, the NNs for many applications (e.g., classification) start from an NN that has been pretrained on a general dataset, and then adapted and retrained for the specific problem. Thus, different applications may use NNs that share large parts among them.

In existing work on neural network compression, it has been shown that significant compression is feasible, with no or only small impact on the performance of the NN in a particular use case. As the description of the network topology is rather small compared to the parameters/weights, compression technology will in particular address compression of weights, e.g., by reducing their number, quantising them, representing them more compactly etc.

Any use case, in which a trained neural network (and its updates) needs to deployed to a number of devices, which potentially run on different platforms or in applications of different manufacturers, could benefit from a

standard for the compressed representation of NNs. Compression will enable an application to have smaller representations of NNs sent across network connections, and potentially also NNs having smaller memory footprint during inference. While exchange formats for NNs exist (e.g., ONNX, NNEF), they do not yet address compression and incremental updates. What is currently missing is a representation of the compressed parameters/weights of a trained network, complementing the description of the network structure/architecture in existing (exchange) formats for neural networks. A standard for the compressed representation of NNs will ensure interoperability with inference environments on different platforms.

Scope

MPEG aims to define a compressed, interpretable and interoperable representation for trained neural networks. The representation shall be able to

- represent different artificial neural network types (e.g., feedforward networks such as CNN and autoencoder, recurrent networks such as LSTM, etc.)
- enable scalability, e.g. trading off compression rate vs. performance of the NN
- enable efficient incremental updates of compressed representations of NNs
- allow inference without performing full reconstruction of the original network, in order to enable faster inference than with the original network
- enable use under resource limitations (computation, memory, power, bandwidth)

A possible processing chain for neural network compression is shown in Figure 3. Some of the steps change (simplify) the model structure, while others affect only the weights.

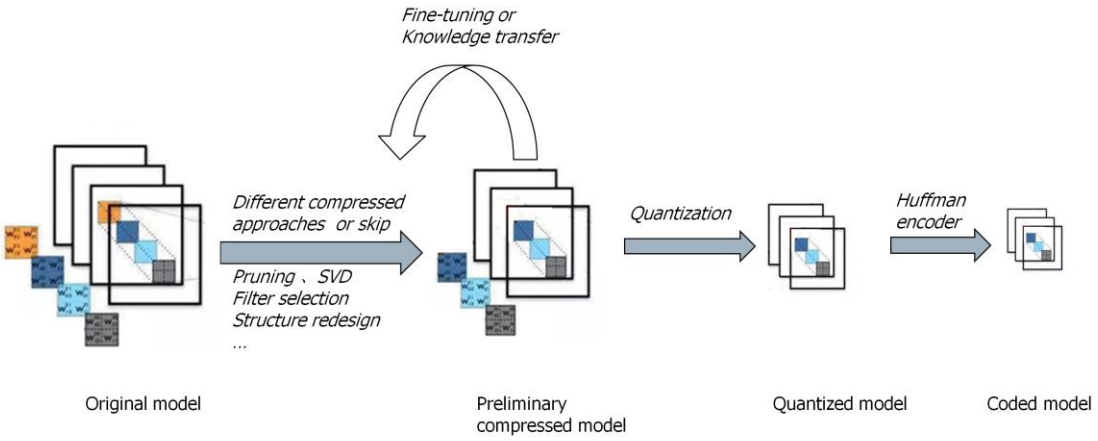


Figure 3 - Framework for neural network compression.

Use cases on NN distribution/deployment

- **UC1 Installing NN-based applications**
  In an extension of the current app model, users install Neural Network-based applications on their devices. The NN-based core of the application can be compressed.

- **UC2 Camera application with object recognition**
  Recent smart phones include cameras, that adjust their automatic mode based on scene/object recognition results. For example, the Huawei Mate 10's camera app classifies 13 scene/object types.

  In order to add/improve classification, the trained NN needs to be updated in the app. There are probably also cases where users are interested in adding types of objects/scenes, training some on their own and transferring models from one phone to another.

- **UC3 Translation application**
  Some translation apps (such as e.g. Microsoft Translator) make use of NNs for language recognition and/or synthesis. Those apps can be extended with a set of languages, that can be downloaded within the app. The data for a language contains the trained networks for a language, but also other data (e.g. dictionaries).

- **UC4 Large-scale public surveillance**
  In recent years, surveillance cameras in public area are increasing and it is becoming a challenging problem to realize automatic surveillance systems for public areas such as stations, large-scale parks,

exhibition halls, stadiums and shopping malls. Video analysis using deep neural network is becoming one of essential core functions for automatic surveillance systems.

Figure 4 shows one of potential automatic surveillance systems. Area 1 to area M have local automatic surveillance system composed of surveillance cameras which have a function to extract metadata like the output of object detection, object tracking, action recognition, etc. by deep neural network and a local server as "surveillance AI" to recognize the situation of the corresponding area by the metadata from the cameras, respectively. A cloud server periodically re-trains the neural network for the metadata extraction on the surveillance cameras and provide it to the cameras. Since the automatic surveillance system can be connected to a huge number of cameras , it will be desirable to compress the neural network when it is transferred to the cameras. Note that it may be necessary to transmit different networks to different areas, e.g., depending on the relevant tasks for each area.

Moreover, since it is expected that it will be realized to collect/generate the training dataset automatically for re-training of the neural network in the future, the update cycle of the neural network will become shorter. In consideration of such situation, the compression of the neural network will be more important to reduce the traffic over the network.

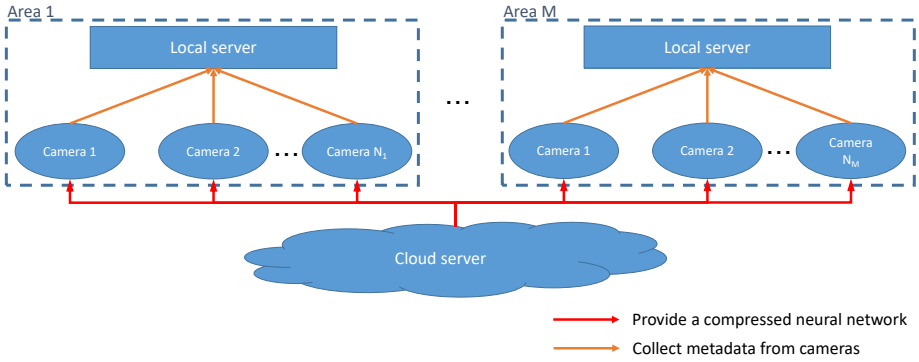


Figure 4 - An example of future public surveillance system.

- **UC5 Visual pattern recognition (VPR)**
  Visual pattern recognition (VPR) techniques are an important component of an intelligent system and are used for many application domains. Visual pattern recognition is one of the most effective applications to which deep neural network can be applied, and excellent recognition results have been reported in many recent studies. In particular, visual analysis and pattern recognition applications in mobile and IoT devices are becoming more popular. In order to apply DNN in a mobile and IoT environment in which computing capability is limited, a system configuration cooperating with a PC or server is considered as a practical way. In other words, a network is trained in a PC or a server, then pattern recognition is performed using the trained network in a mobile/IoT device.

  Figure 5 shows an example such case of VPR systems using DNN in which object detection is performed. A set of neural networks with different algorithms are trained and provided by PC/Server. A user can select an appropriate network for the given application of pattern recognition, then the network model and the trained results of the selected network are delivered to a mobile/IoT device in which pattern recognition is performed.

  In addition, a user can also optionally report the evaluation of the pattern recognition result, which is feedback to the server in which network will be retrained/updated.

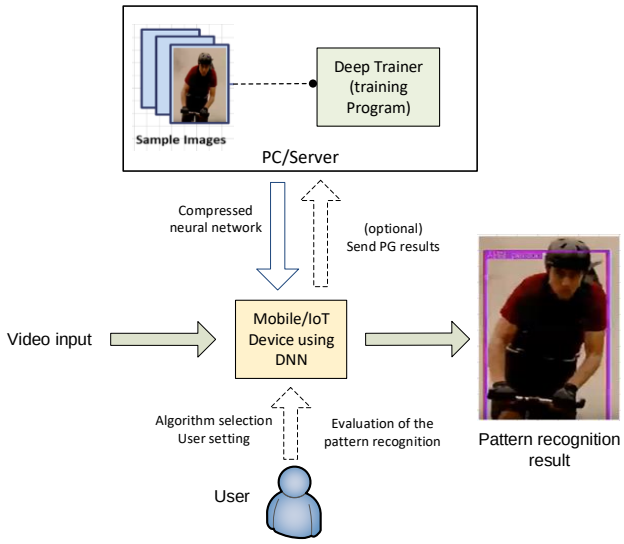


Figure 5 - An example of pattern recognition using DNN in an mobile/IoT device.

- **UC6 NN representation for devices with limited memory and bandwidth**
  This use case covers a scenario where the device that will run a neural network has very limited memory, computational capability and bandwidth. An example of such an application is object recognition in memory, computation and bandwidth limited IoT devices which are capable of media processing.

  It is challenging for such devices to load large neural networks and do inference. It is also not always possible to compress a neural network so that it fits into the memory of the IoT device. Moreover, such a demanding compression usually has an impact on the inference performance (i.e. significant loss of accuracy). Hence, it is required that a mechanism is defined to handle loading and inference of such neural networks that run on devices with very limited memory, computation capability and connectivity bandwidth.

  A group of such devices, connected together, form an processing chain of neural network layers, in which each devices may process one or more layers of computation. This group of devices can be used at both the training stage and the inference stage. Another need is to make up for layer-wise NN computation, in case some of NN layer communications are interrupted due to various reasons e.g. communication errors, inaccessible devices etc.

- **UC9 Efficient re-use of neural networks among different media applications**
  Many applications utilize neural networks derived from the same neural network. This is the case for example for mobile phone apps. The common procedure consists of taking a neural network pre-trained on a large dataset (such as ImageNet), which is able of extracting high-quality generic visual features, and deriving from it a new model for a specific down-stream task. The derivation may consist of one or more of the following options:
  - fine-tuning all layers;
  - freezing some layers/weights and fine-tuning only the remaining layers/weights;
  - freezing or fine-tuning some layer, and adding new layers and branches which are trained.

  This means that several apps would need to store on the device the same base neural network multiple times. This is a waste of both storage and inference speed. In fact, inference speed may become very low if multiple models need to be run on the same device at the same time, e.g., when taking a picture or video the camera may run a camera parameter tuning neural net, a person detection neural net, a style-transfer neural net, etc.

  There is a need to efficiently share & reuse a network model among multiple applications and tasks. Another need related to the use case is the update (versioning) of the neural network that an app use. The versioning can either be needed to upgrade the network to a better performing one (tested in the server side). Still another need is the layer-wise or filter-wise updating of the neural network that an app use in each stream, a network layer (or convolutional filter) is sent to the device to perform designated computation on devices. Another need is to make up for layer-wise NN computation, in case some of NN layer communications are interrupted due to various reasons e.g. communication errors, inaccessible devices etc.

- **UC14 Electronic Health Record and Genomic Data**
  The development of deep learning network (DNN) technology and the accumulation of big data have led to active research to utilize deep learning about medical information. The technology of diagnosing diseases by applying deep learning to medical images has been developed much, but recently, there is a growing interest in exploiting health record and genomic data for better healthcare by use of DNN technologies on individual health record and genomic data to better predict the correlation between disease and individual trait.

  The efforts of life sciences researchers have accumulated various levels of information, from genomic data to medical information data, and open Databases are being provided. As an example, it is possible to use this data to develop algorithms for predicting disease risk by collecting and processing medical information big data centered on genome information of open databases such as GTEx, NCI-60, ENCODE, ICGC, 1000 Genome, NIH Epigenomics Project and GIANT.

  Generally, genetic information and medical information data cannot be easily shared because of privacy violation. Therefore, it is necessary to define an intermediate neural network data exchange mechanism that can guarantee the privacy protection.

- **UC15 Dynamic adaptive media streaming**
  Universal Media Access (UMA), as proposed in the late 1990s and early 2000s, is now reality. It is very easy to generate, distribute, share, and consume any media content, anywhere, anytime, on any device. These kinds of real-time entertainment services -- specifically, streaming audio and video -- are typically deployed over the open, unmanaged Internet and account for the majority of the Internet traffic. A major technical breakthrough and enabler was certainly the HTTP Adaptive Streaming (HAS) technique resulting in the standardization of MPEG Dynamic Adaptive Streaming over HTTP (DASH). According to Cisco's Visual Networking Index (VNI), the global IP video traffic will be 82% of all IP traffic by 2021 (up from 73% in 2016). Nielsen's Law of Internet bandwidth states that the users' bandwidth grows by 50% per year (i.e., 10% less than Moore's Law for computer speed), which roughly fits data from 1983 to 2018. Thus, the users' bandwidth will reach approximately 1 Gbps by 2021. Therefore, as media rates and network bandwidth continuously increase there is a continuous need to provide support for dynamic adaptive media streaming in order to address the heterogeneity of today's and future devices and networking infrastructure.

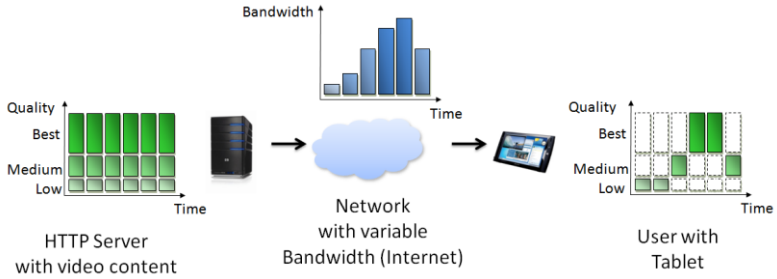


Figure 6 - HAS in a nutshell.

HTTP Adaptive Streaming (HAS) including but not limited to Dynamic Adaptive Streaming over HTTP (DASH) allows for seamless delivery of media within heterogeneous environments over the top of existing infrastructure to enable high Quality of Experience (QoE). The QoE is mainly determined by factors like media bitrate/throughput, (start up) delay, stalls, switches, etc. and AI-based methods can help to improve the QoE as pointed out recently. As a consequence, we see a need to support dynamic adaptive media streaming within this new activity on coded representation of neural networks. In particular, the coded representation of neural networks needs to be made available to such clients (i.e., initially prior to streaming and updates during the streaming session) to enable better adaptation decisions and, thus, increase QoE.

- **UC16A Audio Classification / Acoustic Scene Classification**
  The goal of acoustic scene classification is to classify sound into one of the providing predefined classes. Those predefined classes are characterizing the environment in which it was recorded for example "bus", 'home", "office", "street" as illustrated in Figure 7.

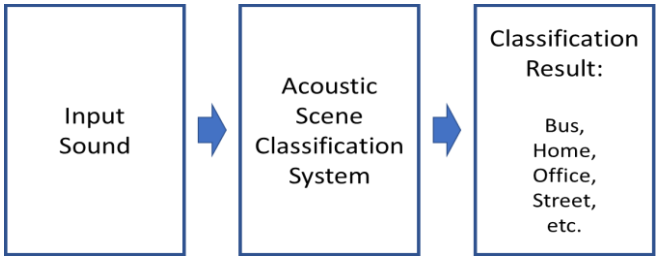


Figure 7 - A schematic illustration of the Acoustic Scene Classification.

This Acoustic Scene Classification can be used in data compression and ambient sound reduction in Active Noise Cancellation Application.

- **UC16A Audio Classification / Sound Event Detection**
  The goal of sound event detection is to detect the presence of predefined sound in multisource conditions like our everyday life. Those predefined sounds may be characterizing speech, object sound like bird singing sound and environment sound of car passing by.

  The Sound Event Detection can be used in data compression and home security applications.

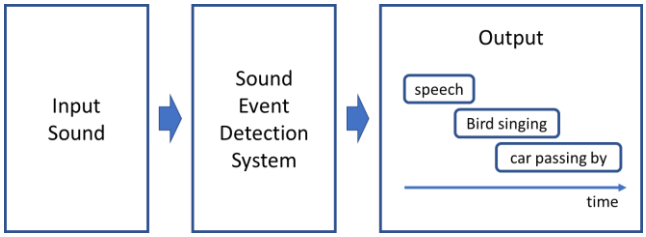


Figure 8 - A schematic illustration of Sound Event Detection.

Use cases on NN (re)training

- **UC7 Deep NN Factory**
  The scientific and technical advancements witnessed in the last 5 years in the field of machine learning based on deep neural networks is making a whole new business area grow at an incredible pace. Main cloud providers are offering increasingly engineered services that can be used by their customers to train specialized networks on demand. If on the one hand training such complex networks in a reasonable amount of time requires the availability of a non-trivial amount of dedicated hardware, which can be affordable only by said providers, on the other hand the execution (inference) phase can be accommodated on much simpler infrastructure, in most cases affordable by the customers themselves. Thus, the opportunity to "download" pre-trained networks and use them on site starts to be a meaningful use case in this domain. The necessity to retrain/refine networks as long as the statistics of the data evolve or new classes are needed is an additional element making this scenario even more realistic. A "Deep NN Factory" is therefore a system implementing such training/refinement service and producing pre-trained (or refined) deep networks on demand of its customers. Delivered NN by the Factory can be compressed using a lossless or lossy technique, depending on the requested delivery latency. This use case is limited to those factories operating in the multimedia domain, and in particular specialized in multimedia classification tasks.

- **UC8A Personalized machine reading comprehension (MRC) application**
  MRC is an advanced deep learning technique in natural language processing. For the user's question, it finds the answer in the natural language text using the neural network (NN). In order to answer user's questions based on personal e-mail text and SMS messages, the MRC NN model should be learned from the personal questions and the personal texts. However, because of the privacy protection, it is difficult to collect and learn the personal data to send to the central cloud server. This use-case describes the case where the user's edge device (i.e., smart phone) learns personalized MRC NN model using the personal MRC usage record. The personalized MRC NN model learned at the user's edge device is sent to the central cloud server and used to build an updated version of shared MRC NN model. Figure 8 shows an overall architecture for personalized MRC application.

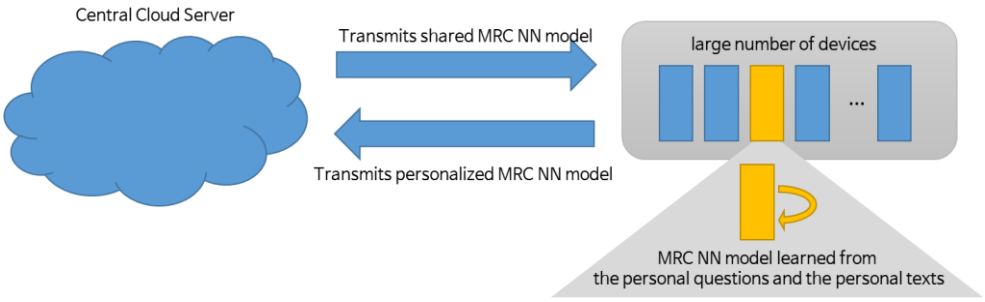


Figure 9 - An overall architecture for personalized machine reading comprehension application.

- **UC8B Machine Translation application**

  In recent years, deep learning technique is applied in machine translation and shows a good performance enhancement. In speech translation, speech recognition and speech synthesis is used for processing input and output data and translation engine is used between speech recognition and speech synthesis modules. For all three modules, the neural network (NN) is applied successfully and provides good translation results. For a better translation results, new words and information on new adjacent words should be learned every time they were created and used. In fact, these new language resources of different languages are created and learned in distributed locations, sites, applications, devices or services. To apply new learned language resources to the translation system affects the translation accuracy and performances. This use-case describes the case where the on-line language resources such as new words and adjacency information are learned and applied to NN model incrementally. The learned translation NN model is sent to the central cloud server and used to build an updated version of shared translation NN model. Figure 10 shows an overall architecture for a translation application with incremental learning of new language information.

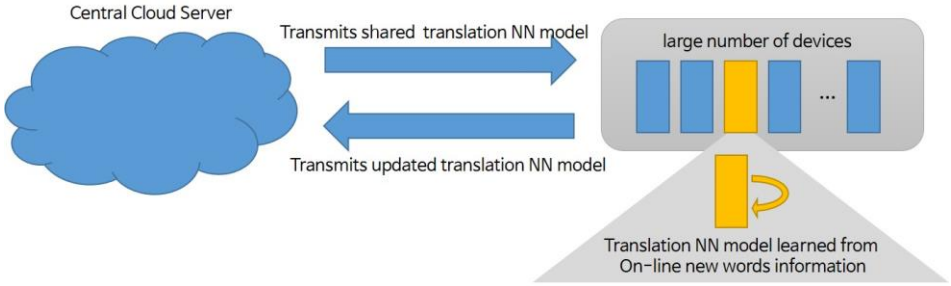


Figure 10 - An overall architecture for machine translation application.

- **UC10 Distributed training and evaluation of neural networks for media content analysis**

  This use-case concerns the distributed training and evaluation of neural networks for applications such as object detection and data generation performed on mobile devices and other devices, or at the edge. The open issues related to centralized solutions: 1) power in-efficiency for centralized training; 2) data communication; and 3) the deployment of the learned models.

  It is good to have continuous training and evaluation capabilities, where the NN representation can be generated and shared in such distributed use cases. The validated model updates are shared with a central entity and/or other devices to improve the media analysis performance. Also, the training devices are assumed to have media capture capabilities, or other means for utilizing content for training (e.g., edge nodes receiving content from devices being served).

  In distributed (or federated) training, there is a central server which orchestrates the distributed training on several end devices (training devices) for the same (or similar) task. Each training device uses captured or other media to perform a local model update on the local version of the neural network. The estimated model update is not communicated to the central entity and/or to the other devices until it is validated. In fact, for each training-device performing a partial training, it will be extremely important to evaluate the *quality* and *integrity* of the partially-trained model, in order to decide whether the model-update estimated by a certain training-device should be taken into account (i.e., communicated to a central entity and/or to all other training devices) or be temporarily ignored. If it is successfully validated, the model update is then taken into account. This may mean that the model update is readily shared with all other training devices or it will be further combined with model updates from other training devices and then shared after a centralized validation.

Examples of aspects to be evaluated will be robustness to adversarial examples, performance based on the specific task (e.g., object detection, data-generation), etc. The goal is to standardize the signaling content and format needed to evaluate neural networks. Signaling content may include data sent by a central entity to be input to the neural network, optionally also the ground-truth output (if the evaluation is done on the training device), feedback from the training device (neural network's output if evaluation is done on a central entity, otherwise only the evaluation results), proposed weight updates in case of federated training. To implement a model resembling with distributed training to exploit the computational power of many distributed devices, in which training methods should be able to ensure the diversity among the model architectures and the training data across individual training devices.

Case A:
The server needs to signal, to different device, different model types & architectures & (optionally) initialization parameters & training datasets. This signaling is from server to each device in P2P mode.

Case B:
The server needs to signal, to different devices, the same model type & architecture & (optionally) initialization parameter & training datasets. This signaling is in broadcast mode. The server sends indication to each devices signaling which connections weights need to be masked (i.e. set to zero). This part of signaling is P2P as in case A.

Case C:
The server needs to signal, to different devices, the same model type & architecture & (optionally) initialization parameter & training datasets. This signaling is in broadcast mode.

The dropout is achieved by each device autonomously. For all cases, devices send back the trained model, and meta-data (e.g. device identifier). For some life-critical systems e.g., autonomous driving or medical diagnosis, the analysis of media contents has to be transparent and explainable to human understanding.

Use cases on image/video processing and coding

- **UC11 Compact Descriptors for Video Analysis (CDVA)**
  MPEG CDVA specifies a compact descriptor for video segments, aiming to support search for specific objects across video collections. The standard describes the bitstream representation of such a descriptor, as well as those parts of the extraction process that are required to ensure interoperability of the resulting descriptors. Features extracted from deep convolutional neural networks (often named deep features) have been shown to be applicable for a wide range of computer vision tasks. CDVA makes use of features extracted from CNNs, which have been shown to provide good performance and are complementary to traditional features such as those included in CDVS.

  There are two levels of conformance: With strict conformance, the neural network is exactly defined, and thus the issue of transmitting the network is only one of the initial deployment of the application. With loose conformance the network may be exchanged, but the same network representation needs to be shared between all parties to establish interoperability. In this case, an update of the neural network may need to be updated more frequently. Intended use cases of CDVA include descriptor extraction on devices such as smart phones or set top boxes.

- **UC12 Image/Video Compression**
  Recently, studies on still image and video compression based on deep neural network (DNN) have been actively conducted, and it has been reported that its performance can be comparable to the conventional image compression standard. Image/video compression using DNN is widely thought to be a new compression method. RNN (Recurrent Neural Network) is mainly applied for image/video compression. In particular, for video compression, different approaches have been attempted to apply deep learning techniques on a tool-by-tool basis or on the entire codec.

  DNN based image/video compression requires coded representation of neural network in the following two aspects. Image/video encoders and decoders are used at different locations in general applications codec. In other words, a feature vector, which is the output of the encoder to the input image/video, is stored or transmitted, and the original input image/video is reconstructed at the decoder. Therefore, the coded representation of the trained neural network of a codec is needed to be delivered to decoders. In addition, re-trained network is desirable to be updated periodically.

  In addition, NNR may be needed in a specific application environment of the image/video codec. For example, there may be available different NNs customized to categories of image/video to be encoded. In this case, the coded representation of NN associated with a given input image/video may be required to support such application.

- **UC13 Distribution of neural networks for content processing**
  Recently, analysis and processing of media such as images and videos require the application of neural networks. One example of content processing is applying neural network based super resolution (NSR) on a video at the client side, so that the video can be transmitted to that client at reduced resolution and thus save bandwidth.

  The neural network performing NSR may be trained to increase the resolution by a specific factor (e.g., 2), thus the sender needs to down-sample the content by the same factor and signal to the client the neural network to be used. This may consist of either signaling the neural network type (e.g., the super resolution factor) if the client already has that neural network or the actual neural network weights and topology.

  Furthermore, the neural net may be specifically trained (or fine-tuned) on the content which is being transmitted, for example on a video. In this case, the neural net needs to be transmitted to the client either before the video transmission, during the video transmission in case different networks are trained for different temporal portions of the video. This can be extended to other content processing techniques apart from super resolution, such as other content enhancing neural networks. In this use case proposal, we use neural network and model interchangeably.

The following section provides the list of requirements that shall fulfilled by responses to the Call for Proposals closing in March 2019.

Requirement	Description	Comment
Efficient representation of the network	The size needed to represent the compressed network should be lower than 30% of the size of the original network.	
Support representation of different types of artificial neural networks	The compression method shall be applicable to any type/architecture of neural network, and not specific to particular types (e.g., CNNs).	
Self-contained representation of parameters and weights	The representation of the compressed neural network shall contain all required information for decoding the parameters and weights (i.e., not require external information for their interpretation).	The structure/architecture of the network may be contained in external information.
Performance of reconstructed network comparable to original network	The use of the reconstructed network after decoding shall result in a performance comparable to the original uncompressed network in the specified use cases.	A method may support lossy compression, which allows trading off performance against compression efficiency.
Inference with compressed network	It is desirable that the compressed network can be directly used for inference without complete reconstruction of the original network. Some methods (e.g., pruning, quantisation) can result in a reduced network that can still be directly used for inference, while others may require decoding/reconstruction of the network in order to perform inference.	Avoiding the reconstruction of the original network is expected to result in lower runtime and memory requirements during inference. Some decoding steps could be necessary to use the compressed network for inference, even if complete reconstruction is not performed.
Encoding without original training data	The proposed method must be able to work without access to the original training data. Optional modes of using the method, which improve the performance when the original training data is available, may be supported.	
Low computational complexity and memory consumption of decoding	The computational complexity and memory consumption of the decoding process needs to be suitable to support use on devices with limited capabilities (e.g., mobile phones, smart cameras).	

Output Documents
N17924 - Use cases and requirements for Compressed Representation of Neural Networks
N17929 - Evaluation Framework for Compressed Representation of Neural Networks
N17931 - Report on the CfE on Compressed Representation of Neural Networks
N17934 - Call for Proposals on Neural Network Compression

4 Network-based Media Processing

The Network-Based Media Processing (NBMP) framework enables the creators, the service providers and the consumers of the digital media to describe media processing operations that are to be performed by the media processing entities connected through the digital networks as shown in Figure 1. It provides a method to describe a workflow of a media processing service as a composition of a set of Processing Functions provided by a Media Processing Entity which are accessible through the NBMP Application Programming Interfaces (APIs). A Media Processing Entity applies Processing Functions on the media data and the related metadata received from a Media Source or other Media Processing Entity. A Media Processing Entity also provides Control Functions that are used to compose and configure the Processing Functions. A Media Processing Entity produces media data and the related metadata to be consumed by a Media Sink or other Media Processing Entity.

The NBMP framework supports any format of media content, including the existing MPEG codecs and MPEG formats such as ISO/IEC 13818-1, ISO/IEC 14496-12, ISO/IEC 23008-1 and ISO/IEC 23009-1.

The NBMP framework supports the delivery over IP-based networks using the different transport protocols. (e.g., TCP, UDP, RTP and HTTP).

The NBMP framework support the existing delivery methods such as streaming, file delivery, push-based progressive download, hybrid delivery, multipath and heterogeneous network environments.

The following diagram depicts the NBMP architecture that will be used as a reference architecture to scope the NBMP work:

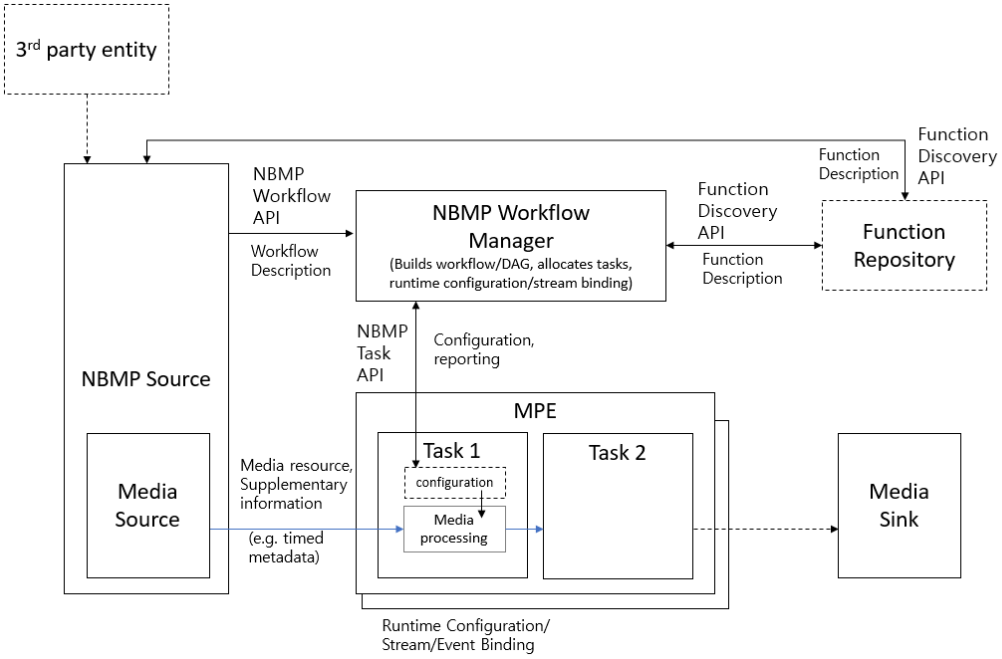


Figure 11 - NBMP Reference Architecture.

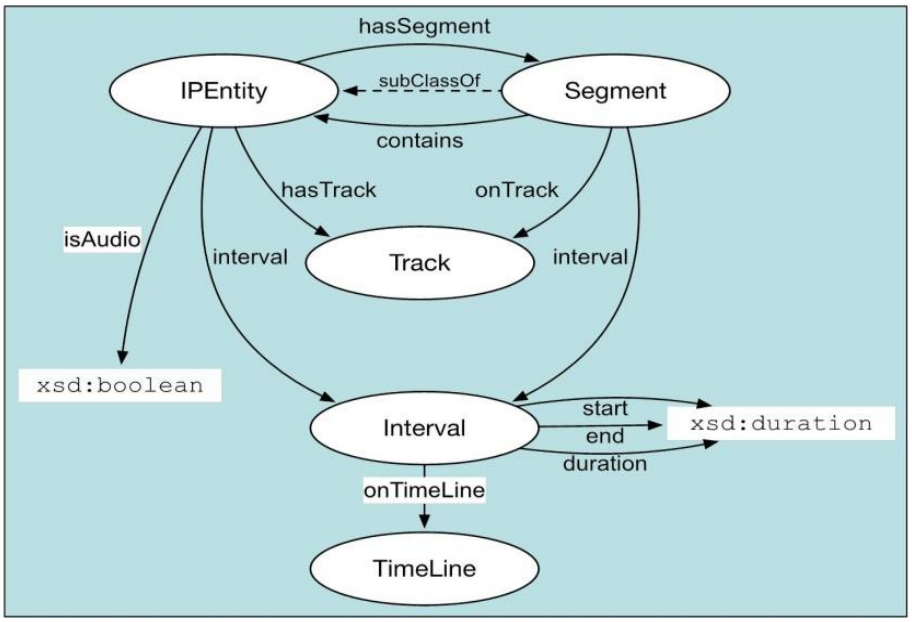
An input contribution to NBMP group submitted describing MPEG-M and its relevance to NBMP, as follows:
Xin Wang (MediaTek), Panos Kudumakis (QMUL), Leonardo Chiariglione (CEDEO) and Jaime Delgado (UPC), 'Overview of MPEG-M for NBMP', Ref. as ISO/IEC JTC1/SC29/WG11/M44558, Macau SAR, China, Oct. 2018.

Output Documents
N17984 - WD of ISO/IEC 23090-8 Network-based Media Processing
N17985 - Description of Core Experiments on Network-based Media Processing

5 Smart Contracts for Fair Trade of Music

Media Value Chain Ontology (ISO/IEC 21000-19) facilitates IP rights tracking for fair and transparent royalties payment by capturing user roles and their permissible actions on a particular IP entity. However, widespread adoption of interactive music services, e.g., remixing, karaoke and collaborative music creation, thanks to Interactive Music Application Format (ISO/IEC 23000-12), raises the issue of rights monitoring when reuse of audio IP entities is involved, such as, tracks or even segments of them in new derivative works.

Audio Value Chain Ontology (ISO/IEC 21000-19 AMD1), addresses the aforementioned issue by extending MVCO functionality related to description of composite IP entities in the audio domain, whereby the components of a given IP entity can be located in time, and for the case of multi-track audio, associated with specific tracks. The introduction of an additional 'reuse' action enables querying and granting permissions for the reuse of existing IP entities in order to create new derivative composite IP entities.



Audio Value Chain Ontology (ISO/IEC 21000-19 AMD1) conceptualisation.

Furthermore, MVCO/AVCO smart contracts by facilitating machine readable deontic expressions for permissions, obligations and prohibitions, with respect to particular users and IP entities, could be used in conjunction with distributed ledgers, e.g., blockchain, enabling both transparency and interoperability towards fair trade of music.

For further info please visit MPEG Developments.

Resources

- ISO/IEC Information Technology – Multimedia Framework (MPEG-21) – Part 19: Media Value Chain Ontology AMENDMENT 1: Extensions on Time-Segments & Multi-Track Audio (ISO/IEC 21000-19/AMD 1:2018). *Published 1 June 2018.*

- ISO/IEC Information Technology – Multimedia Framework (MPEG-21) – Part 8: Reference Software AMENDMENT 4: Media Value Chain Ontology Extensions on Time-Segments & Multi-Track Audio (ISO/IEC 21000-8/AMD 4:2018). *Published 12 October 2018.*