Format Agnostic Scene Representation v2

Deliverable D3.1.2

ICoSOLE identifier:	ICoSOLE-D3.1.2-DTO-Format Agnostic Scene Representation_v05.docx
Deliverable number:	D3.1.2
Author:	Werner Bailer, Gert Kienast, Georg Thallinger (JRS), Philippe Bekaert (iMinds), Juergen Schmidt (DTO) David Marston, Richard Day, Chris Pike (BBC)
Internal reviewer:	Werner Bailer (JRS)
Work package / task:	WP3 / T3.1
Document status:	Final
Confidentiality:	Public

Version	Date	Reason of change
0.1	2015-08-27	Initial version based on D3.1.1
0.2	2015-09-28	Integrated Version with contributions from iMinds, JRS, BBC and DTO
0.3	2015-10-05	Added annex, minor updates throughout the document
0.4	2015-10-05	Version for internal review
1.0	2015-10-06	Final version

The work presented in this document was partially supported by the European Community under the 7th framework programme for R&D.

Table of Contents

1 Executive Summary

This deliverable defines the scope of the scene representation and its relation to the overall system architecture.

In the DoW, the task format-agnostic scene representation is defined as follows: *The goal of this task is to define a common site-global framework into which all captured video, audio and metadata are to be registered 3D-spatially as well as in time (through consistent time stamping and synchronisation).*

This scene representation focuses on the sensors involved in capturing data, their parameters, the content and metadata streams they produce, as well as processing components that modify and combine these streams.

The entities in the scene representation are consistent with those defined in the overall system architecture. The definition of the scene representation adds specialisation of some of these entities. It defines attributes of the involved devices/objects, as well as metadata carried in metadata streams of these devices.

The semantic scene creation in WP4 defines the scene used for production, consisting of the video and audio streams and objects in the scene. In order to create this scene, and a spatiotemporally alignment of the data, information about the capture devices and their metadata are needed. The production-side scene has a similar scope as MPEG BIFS, while the capture scene representation is more focused on the capture devices and their metadata.

This document is structured as follows: section 3 lists requirements gathered for the scene representation. Section 4 describes the approach of the generic scene representation model. Section 5 describes actual model structures and values. Section 6 and Section 7 describe the properties of processing devices and entity nodes respectively. The structure is based on the previous deliverable D3.1.1 defining the initial parameter settings with a focus on modifications and extensions, which deemed necessary and relevant due to the knowledge gained in course of the project execution.

The annex specifies the metadata formats, i.e., the representation of the attributes specified in the document using JSON.

2 Introduction

2.1 Purpose of this Document

This document describes the representation of the audio visual scene and its components. It is the reviewed and refined version of the previous edition D3.1.1, taking into account the findings of various technical discussions amongst consortium partners as well as the experience for setting up the first field trial. The main changes with respect to the previous version are revisions of the parameters for audio and video, and adding sections on processing and entity nodes.

2.2 Scope of this Document

This document covers the parameter specifications to be validated by the implementation of the first prototypes.

2.3 Status of this Document

This is the final version of Deliverable D3.1.2.

2.4 Related Documents

This document is based on the analysis of requirements in D2.2 “Use cases and requirements” as well as on the previous version of D3.1.1 “Initial format agnostic scene representation”.

The document is also related to the ICoSOLE system architecture, described in D2.3 “System architecture and interface definitions”.

3 Requirements

The following requirements have been collected from the use case descriptions. Since in the course of the project execution no deviations from these requirements have been identified, they can be considered as being stable and still applicable:

- Represent input metadata streams
- Represent object-based audio streams
- Update position and orientation of objects/devices
- Metadata model describing the spatially outspread event
- Specify semantic and technical metadata describing the professional content sources
- Streams can be linked with a specific event (set) and device location
- Media format, recording parameters, device description, content source, and other parameters can be specified
- Content is supplemented with metadata, including geo-locations, recording parameters, or directional information as well as manual annotations
- For produced audio (i.e., not the raw captured audio, but audio that has been through sound mixing and is ready for streaming) its metadata should use the Audio Definition Model for its technical metadata.
- Represent static content and metadata from location scouting
- Static metadata on event setup
- Metadata streams carried with media data, for instance created by scene creation process (UC-3.4), can be passed to the user endpoint along with the audio and video

4 Scene representation model

4.1 General Structure

The generic data representation scheme follows a hierarchical tree structure describing the spatial and temporal relation of all captured audio-visual data in one common framework. The highest level is called ‘layered scene representation’ and may consist of different scenes. Its representation in the architecture is called ‘set’. Each (sub-)scene describes a closed area (e.g. room, stadium) with several cameras and corresponding audio sources that can be watched or can be navigated through by the editor or the end user. The different scenes might occur simultaneously (different locations of same sport event like different sections of a race or different arenas of the same Olympic Games).

One scene comprises at least one tree, representing sub-scenes, and the audio/video scenes respectively. Figure 1 shows an example of a scene containing one audio tree and one video tree. Those trees consist of individual input and processing objects. The scene represents professional, semi-professional, and user-generated content. The devices and the streams within the scene are controlled and managed by the ICoSOLE system. Its class relations are covered by the system architecture (cf. D2.3). The scene represents an instantiation of different classes of the system architecture and the scene description covers a concrete implementation of the (abstract) classes of the system architecture. As an example, different processing devices within the scene description are specializations of (generic) base classes of the system architecture (Audio Proc 1, Audio Proc 2 in Figure 1). Its functionalities might be something like ‘AudioMix’, ‘HOAEncode’ or ‘NoiseGate’ as examples for audio functionalities or the corresponding video functionalities. Because the class relationships are adequately described within the system architecture, the scene representation can be reduced to a simpler hierarchic description, e.g., known from the scene description of the FascinatE project [FascinatE D2.1.3].

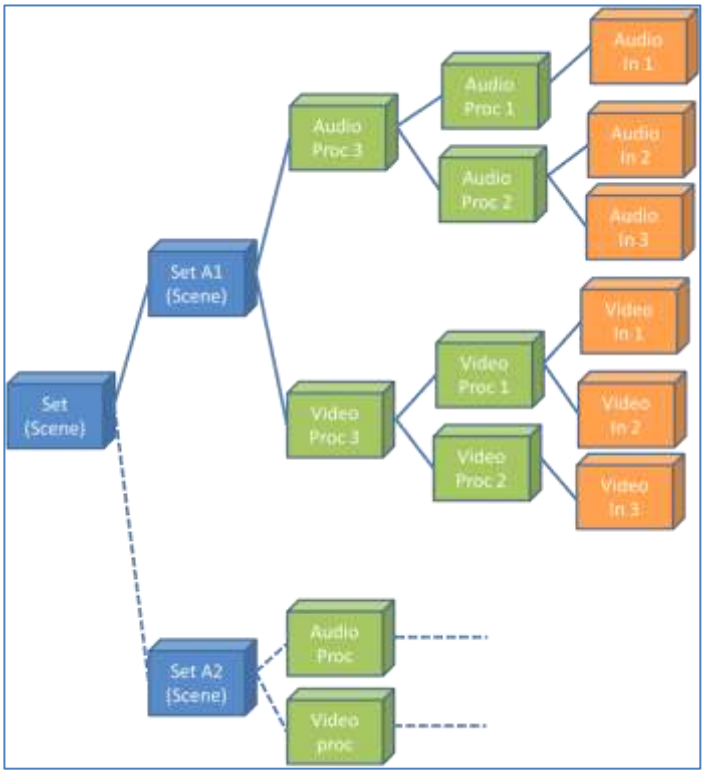

Figure 1: Schematic overview of the components of an audio visual scene. The dashed lines indicate further sub-scenes (not detailed in the diagram).

Beside the evaluation of the scene description of the FascinatE project, we have considered the reuse of entities from MPEG-4 BIFS [BIFS, 2005] for the scene representation. While there are some overlaps in terms of general constructs, the BIFS representation focuses much more on the presentation of

streams and media objects, and lacks the object for representing capture devices and their parameters. Figure 2 shows the relation between some entities of the system architecture to MPEG BIFS entities (shown in blue).

The overlap on the capture side is very limited, thus BIFS does not seem a good candidate for the capture scene representation. It is expected that the overlap with BIFS increases further down the production chain, when the representation evolves from representing capture devices to streams distributed to specific consumers.

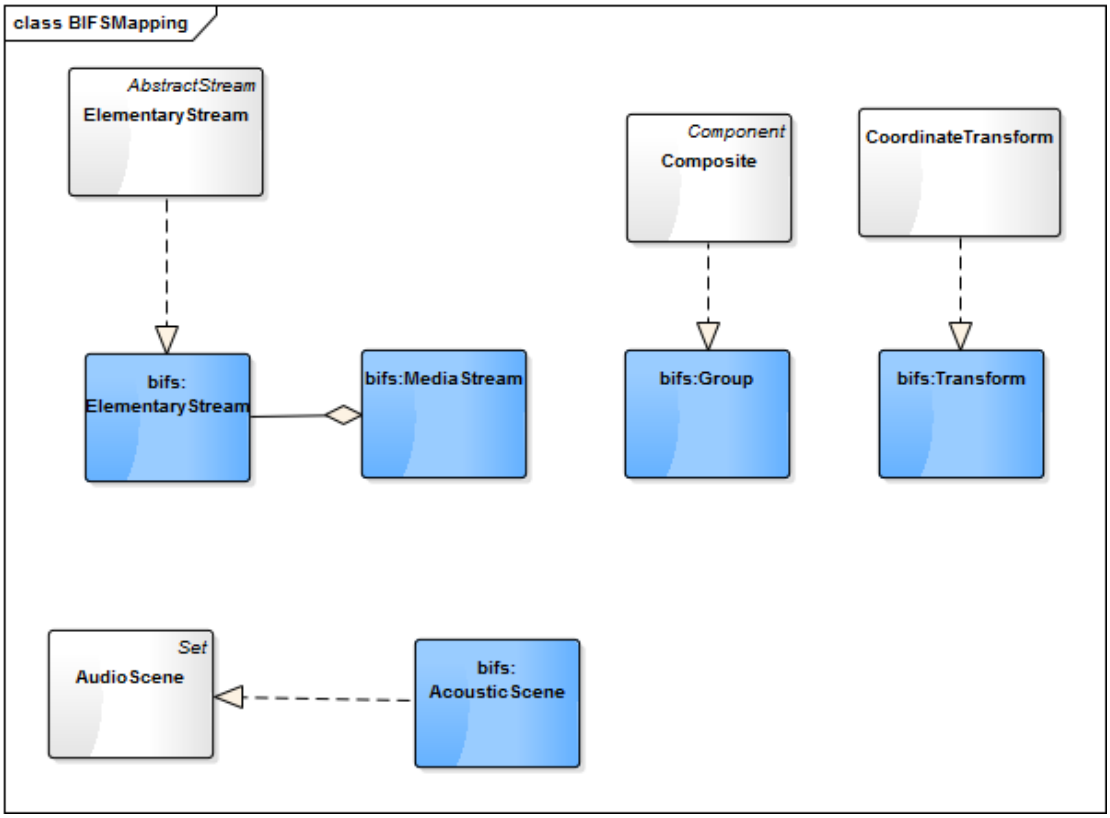

Figure 2: Relation to BIFS objects.

4.2 Nodes

Nodes are concrete implementations of the (abstract) devices in the system architecture. If several nodes have common properties or functionalities, use of object-oriented methods like inheritance should be taken into account for its implementation. To avoid functional limitations and confusion with the system architecture document, these relations are out of scope of this document.

Because the nodes cover the ICoSOLE functionalities, its functions, position within the tree, limitations and especially its properties have to be described. In principle, nodes can be arranged into several functional groups:

- Input nodes
  - Camera
  - Microphone
  - HOA microphone
  - Omni-directional or panoramic camera rig (treated as one special camera)
  - Direct audio input
  - Direct video input
  - …
- Processing nodes
  - Content analysis
  - Content pre-selection

  - Mix
  - Gain
  - Virtual pan-tilt-zoom in omni-directional or panoramic view
  - …
- Entity nodes (representing objects/persons in the scene rather than sensor/processing information)
  - Subject
  - Object

Note: the list above is not considered exhaustive.

4.2.1 Spatiotemporal transformations

In general, nodes are related to one coordinate system (world coordinate system) and a global time reference (master clock) only. That enables consistent and reproducible data and relations. In some foreign systems, e.g., FascinatE, spatiotemporal transformations are used within a scene. Since these transformations are linear, they are redundant with respect to a consistent data model. If such a transformation might be necessary, it can be applied within the application and the transformation information can be managed as private data within a metadata stream or as object attributes.

4.3 Default values

If default values are specified, required parameters do not need to be explicitly contained in data packets, but the receiver can assume the defaults for missing parameters.

5 Attributes of Input Nodes

This section describes the attributes associated with the entities in the scene model. The scene model consists of

- A generic entity (Section 5.1) describing the scene.
- A video scene description consisting of one or more cameras (or other video sources): Sections 5.2 to 5.5.
- An audio scene description consisting of one or more microphones or other sound sources: Sections 5.6 to 5.8.
- Auxiliary data gathered during site survey and possibly during the event (Section 5.9). This auxiliary data serves to aid presenting the audio visual content in a spatiotemporal coherent fashion.

5.1 Scene

Scene Parameters					
Parameter	Symbol	Description	Units	Type	Required = R, optional = O
Scene Description		Text description of the scene content		String	O
WorldOrigin		Text description of 3D location at the set, w.r.t. which spatial coordinates of video and audio sources is defined.		String	O

5.2 Camera

A camera or other video source is described by some generic parameters in the table below, geometric camera properties (Section 5.5), photometric camera properties (Section 5.6) and the encoded video stream parameters (Section 5.7). We further require hat video frames captured by the camera are properly tagged with a time stamp.

General Camera Parameters					
Parameter	Symbol	Description	Units	Type	Required = R, optional = O
CamID	CamID	Camera identification number		UUID	R
Temporal Resolution	Fps	Temporal resolution	Frames/s	Real	R

5.3 Geometric camera properties

Geometric camera properties model the mapping of image coordinates to rays in 3D space and vice versa. They are needed to present video content in a spatially consistent fashion. The exact interpretation of these properties, with formulae describing the mappings involved, is described in an appendix to this document.

There are four kinds of geometric camera properties:

1. Sensor-related properties: these properties are only dependant on the camera sensor and region of interest set. These parameters do not depend on the lens mounted on the camera, nor the cameras position and orientation on the set. These parameters are static during the course of camera operation.
2. Lens-related properties: these properties depend only on the lens mounted on the camera and not on other camera parameters. The properties remain the same regardless of the camera the

lens is mounted on, and quantities are expressed in camera-independent units. Properties such as lens distortion and focal length may vary in time, e.g. on a zoom lens.

3. Sensor-lens assembly parameters: these describe the exact mounting of the lens w.r.t. the sensor. Parameters such as principal point (the optical center of the image) may vary in time, due to zooming e.g.
4. Extrinsic parameters: position of the optical center of the camera (the imaginary point at which light rays captured by the lens would meet if they were not bend to focus on the sensor) and orientation. Position and orientation may of course also vary in time.

The process of determining the actual value of these properties is called geometric camera calibration.

Geometric camera lens properties					
Parameter	Symbol	Description	Units	Type	Required = R, optional = O
Description		Lens manufacturer and model textual description		String	O
Lens model	*L*	Image formation model: rectilinear, equidistant (f-theta fish eye), equirectangular, cylindrical, stereographic, orthographic, equisolidangle)		Enumeration	R
Focal length	*f*	Focal length (rectilinear, equidistant, stereographic, orthographic, equisolidangle models)	Millimetre	Real	R
Hfov	Θ_h	Horizontal field of view (equirectangular or cylindrical model)	Degrees	Real	R
Vfov	Θ_v	Vertical field of view (equirectangular or cylindrical model)	Degrees	Real	R
DistK1(A,B,C)	$\kappa_1(A,B,C)$	1st order distortion coefficient vector for colour components (A,B,C)	1/cm^k	Real	O
DistK2(A,B,C)	$\kappa_2(A,B,C)$	2nd order distortion coefficient vector for colour components (A,B,C)	1/cm^k	Real	O
DistK3(A,B,C)	$\kappa_3(A,B,C)$	3rd order distortion coefficient vector for colour components (A,B,C)	1/cm^k	Real	O
DistK4(A,B,C)	$\kappa_4(A,B,C)$	4th order distortion coefficient vector for colour components (A,B,C)	1/cm^k	Real	O
DistL1(A,B,C)	$L1(A,B,C)$	1st order denumerator distortion coefficient vector for colour components (A,B,C)	1/cm^k	Real	O
DistL2(A,B,C)	$L_2(A,B,C)$	2nd order denumerator distortion coefficient vector for colour components (A,B,C)	1/cm^k	Real	O

Geometric camera sensor properties					
Parameter	Symbol	Description	Units	Type	Required = R, optional = O
Description		Sensor manufacturer and model textual description		String	O
Pixel pitch H	*Sx*	Horizontal pixel pitch (default: 5)	Micrometer	Real	R
Pixel pitch V	*Sy*	Vertical pixel pitch (default: 5)	Micrometer	Real	R
Image width	*Nx*	Number of pixel horizontally in image		Integer	R
Image height	*Ny*	Number of pixels vertically in image		integer	R
Image Origin	*O*	Origin of image coordinate system: top left corner, bottom left corner		Enumeration	R

Sensor-lens assembly parameters					
Parameter	Symbol	Description	Units	Type	Required = R, optional = O
PCX	*Xc*	Principal point x image coordinate	Image coordinate	Real	R
PCY	*Yc*	Principal point y image coordinate	Image coordinate	Real	R
CoDX	*Xd*	Center of distortion x image coordinate	Image coordinate	Real	O
CoDY	*Yd*	Center of distortion y image coordinate	Image coordinate	Real	O
AspectRatio	*a*	Effective pixel aspect ratio		Real	O
Skew	*s*	Effective pixel grid skew angle	Degrees counter clockwise	Real	O

Extrinsics					
Parameter	Symbol	Description	Units	Type	Required = R, optional = O
OCX	*Xw*	Optical center X coordinate w.r.t. parent coordinate transform node or world origin	Meter	Real	R
OCY	*Yw*	Optical center Y coordinate w.r.t. parent coordinate transform node or world origin	Meter	Real	R
OCZ	*Zw*	Optical center Z coordinate w.r.t. parent coordinate transform node or world origin	Meter	Real	R
Pan	α_w	Camera orientation w.r.t. parent coordinate transform node or world origin 1st Euler angle	Degrees counter clockwise	Real	R
Tilt	β_w	Camera orientation w.r.t. parent coordinate transform node or world origin 2nd Euler angle	Degrees counter clockwise	Real	R
Roll	γ_w	Camera orientation w.r.t. parent coordinate transform node or world origin 3rd Euler angle	Degrees counter clockwise	Real	R
EulerOrder	*EulOrd*	Euler order – interpretation of Euler angles (24 possibilities, see [Schoemake, 1994]: XYZr, XYZs, etc...)		Enumeration	R

5.4 Photometric camera properties

Photometric camera properties provide information on how image intensities relate to the intensity of light hitting the camera lens. There are three kind of properties in this class: lens dependent parameters (aperture value, focus distance, and vignetting), camera dependent values (gain, integration time, per-colour channel transforms modelled as a curve in the most generic case, and a colour matrix mixing colour channels captured by the sensor into camera output. The transforms being described here are transforms that took place in the camera or video source being described. Their description allows undoing these transforms in order to homogenize colour output in a semi-automated way if desired.

Camera related photometric parameters					
Parameter	Symbol	Description	Units	Type	Required = R, optional = O
ColourSystem		Colour system of the camera (RGB444, YUV444, YUV422, logC, Rec.709, …)		Enumeration	R
BitDepth	*bpp*	Number of bits per pixel		Integer	R
Primaries	(p_R,p_G,p_B)	Primaries for the colour components		Real	O
WhitePoint	*WP*	White point of the sensor		Real	O
BlackLevel	*BL*	Black level of the sensor		Real	O
Gain	*G(A,B,C)*	Gain factor per colour channel	dB	Real	O
Integration time	T_v	Duration of sensor exposure	1/s	Real	O
Gamma	γ	Gamma curve parameter		Real	O
CurveNrSamples	*Nc(A,B,C)*	Number of samples modelling generic intensity transform per colour channel (piecewise linear approximation)		Real	O
CurveSamples	*U|Vi(A,B,C)*	Curve samples i=0...N_c(A,B,C)-1. 2 coordinates for sample (horizontal axis, vertical axis)		Real	O
ColourMatrix	*CMi,j*	Colour matrix transform, i,j = colour channel indices (#values = nr of colour channels squared)		Real	O

Lens related photometric parameters					
Parameter	Symbol	Description	Units	Type	Required = R, optional = O
Aperture	A_v	Lens aperture f-value	f	Real	O
Vignetting	v_0,v_1,v_2,v_3	Lens vignetting parameters		Real	O

5.5 EncodedVideoStream

Video Coding Parameters					
Parameter	Symbol	Description	Units	Type	Required = R, optional = O
Format	F	Video coding format: e.g. HEVC, H.264, JPEG 2000, Raw		Enumeration	R
Profile	FP	Profile of the video coding format		Enumeration	R
PictureSizeX	P_x	Horizontal size of each frame, if modified by coding scheme	Pixel	Integer	O
PictureSizeY	P_y	Vertical size of each frame, if modified by coding scheme	Pixel	Integer	O
NumTileX	N_x	Number of columns of the tiling (default: 1)		Integer	R
NumTileY	N_y	Number of rows of the tiling (default: 1)		Integer	R
TileSizeX	T_x	Horizontal size of each tile (required if cannot be derived from the syntax of the coding format)	Pixel	Integer	O
TileSizeY	T_y	Vertical size of each tile (required only if $T=1$)	Pixel	Integer	O
CodingColourSys		Colour system, if modified by coding scheme (RGB444, YUV444, YUV422, YUV420,…)		Enumeration	O
CodingBitDepth	Bpp_C	Number of bits per pixel, if modified by coding scheme		Integer	O
CodingOptions		Optional list of parameters describing encoding options and conditions: e.g. encoder description, compression ratio, fidelity (PSNR, SSIM, …), …		Key/ Value	O

5.6 Professional Audio Capture

The parameters for the capture devices are listed below. These could be a microphone, an on-stage device such as a synthesizer or an off-stage feed.[1] This also includes multichannel spatial microphones such as the Eigenmike®.

[1] For microphones, directional parameters refer to rotation from the microphone's frontal direction as specified by the manufacturer.

Audio Capture Device Parameters					
Parameter	Symbol	Description	Units / Enumerations	Type	Required (R),optional (O)
Static Parameters					
Make		Make of device		String	O
Model		Model of device		String	O
DeviceID	DID	Unique ID for the device (e.g. S/N).		String	R
TrackID	TID	Unique ID for track within device.		String	R
NumTracks		Number of tracks (or output signals) emitted by the device.		Integer	R
SampleRate	SR	Sample rate of the device (e.g. 48000Hz)	Hz	Real	O
BitDepth	BD	Bit depth of the device (e.g. 16 bits)	Bits	Integer	O
Coding	CO	Format of the data (e.g. PCM)		String	O
Type		Type of device used	Microphone, DI, Other	Enumeration	R
SignalType	ST	Signal type {A-format / HOA coefficient / …}	Channel, A-format, B-format, HOA, Matrix	Enumeration	R
ArrayCharacteristics	AC	Array characteristics {open sphere / rigid sphere / random placement / Hamasaki square / …}	Mono, OpenSphere, RigidSphere, DummyHead, OCT, SpacedOmni, Fukada, Hamasaki, INA5, IRTCross, ORTF, MS, DoubleMS, Random, Other	Enumeration	O
EulOrd	EulOrd	Euler order – interpretation of Euler angles (24 possibilities: XYZr, XYZs, etc...)	XXX, XXY, …, ZZZ	Enumeration	R
Pattern		Response pattern of a microphone	Omni, Cardioid, Supercardioid, Hypercardioid, Halfcardioid, Shotgun, FigureOfEight, Boundary, Ear, HOA, Other	Enumeration	

Dynamic Parameters					
Pan	*Pan*	Pan direction of device, 1st Euler angle	Degrees, counter clockwise (ccw)	Real	O
Tilt	*Tilt*	Tilt direction of device, 2nd Euler angle	Degrees, ccw	Real	O
Roll	*Roll*	Roll direction of device, 3rd Euler angle	Degrees, ccw	Real	O

5.7 UGC Audio Capture

The audio captured by UGC devices requires a subset of parameters from the professional audio capture devices.

UGC Audio Entity					
Parameter	Symbol	Description	Units / Enumerations	Type	Required = R, optional = O
NumTracks		Number of tracks (or output signals) emitted by the device, even if they are to be used as different sources.		Integer	R
SampleRate	SR	Sample rate of the device (e.g. 48000Hz)	Hz	Real	R
BitDepth	BD	Bit depth of the device (e.g. 16 bits)	Bits	Integer	R
Coding	CO	Format of the data (e.g. PCM)		String	R
SignalType	st	Signal type {loudspeaker channel / microphone signal}	InternalMic, HeadsetMic, BluetoothMic, ExternalMic, Other	Enumeration	R
Preprocessing	pp	String representing the executed pre-processing		String	O
Pattern		Response pattern of a microphone	Omni, Cardioid, Supercardioid, Hypercardioid, Halfcardioid, Shotgun, FigureOfEight, Boundary, Ear, HOA, Other	Enumeration	O

5.8 Composite Audio Scene

The audio from the capture process will not be used directly in the scene description. The capture device signals require signal processing, such as gain and equalisation to provide the appropriate audio signals for producing an audio scene. This would typically be done using a Digital Audio Workstation (DAW). The capture audio also needs to be turned into audio objects, whether they are channel-based, HOA-based or positional objects. This may require some of the capture signals being combined, split or grouped. It will also require new object metadata to be generated, such as the position of the object (which may be different from the capture device position) or the HOA component information.

This compositing stage will take the capture metadata and audio signals and generate audio objects. The set of parameters are shown in the table below. Audio objects can be multichannel so can include traditional channel based audio (such as stereo) and HOA groups of channels, as well as object-based audio with positional information. The audio objects require an adequate set of parameters to allow a suitable rendering of the audio.

The parameters are compatible with a subset of the Audio Definition Model parameters. The 'Type' static parameter indicates the type of audio used: DirectSpeakers (channel-based), Objects (object-based) or HOA (scene-based). The ADM uses a type definition to describe these types hence the names match these, as do the numbers (1, 3 and 4 respectively). The dynamic parameters used depend on the type used, so the numbers in the brackets after each parameter indicate which type the parameter can be used with.

Audio Object Parameters					
Parameter	Symbol	Description	Units / Enumeration	Type	Required = R, optional = O
Static Parameters					
ObjectID	ObjID	Unique ID for the object. Is identical to System Architecture UUID. Needed for referencing of audio pack parameters		UUID	R
SampleRate	SR	Sample rate of the device (e.g. 48000Hz)	Hz	Real	O
BitDepth	BD	Bit depth of the device (e.g. 16 bits)	Bits	Integer	O
Coding	CO	Format of the data (e.g. PCM)	PCM, Coded, Other	Enumeration	O
Type	TY	Type of object: DirectSpeakers (1), Object (3), HOA (4)	DirectSpeaker, Object, HOA	Enumeration	R
Dynamic Parameters					
Xpos (1,3)	*X*	X position of object	metre	Real	R
Ypos (1,3)	*Y*	Y position of object	metre	Real	R
Zpos (1,3)	*Z*	Z position of object	metre	Real	R
Width (3)	*dX*	Size of object in X axis	metre	Real	O
Depth (3)	*dY*	Size of object in Y axis	metre	Real	O
Height (3)	*dZ*	Size of object in Z axis	metre	Real	O
Diffuse (3)	*diff*	1 if fully diffuse, 0 if direct.	0…1	Real	O
Gain (3)	*gain*	Gain to apply to audio signal.	0…inf	Real	O
Interactivity (3)	*int*	Interactivity flag. True allows the object to be altered.	False/true	Bool	O
HOAOrder (4)	*Hord*	If it is an HOA signal, the order of the channel used		Integer	O
HOADegree (4)	*Hdeg*	If it is an HOA signal, the degree of the channel used		Integer	O

Some devices and objects combine multiple channels of audio to represent an entity.

Sometimes several multi-channel devices or objects must be grouped together as an entity in the scene. The parameters covering this are shown in the table below, the grouping is termed a ‘Pack’ as described by the Audio Definition Model [EBU, 2014]. Either the object parameters can be used, or the device ones depending upon the purpose.

Audio Pack Parameters					
Parameter	Symbol	Description	Units	Type	Required = R, optional = O
PackName		Name for the pack		String	R
PackID	ID	Unique ID for the pack.		UUID	R
NumObjects		Number of objects in pack		Integer	R
NumDevices		Number of devices in pack		Integer	R
ObjectIDRef (multiple)		References to ObjectIDs		UUID	R*
DeviceIDRef (multiple)		References to DeviceIDs		UUID	R*

*It must contain at least one ObjectIDRef, or at least one DeviceIDRef, but not contain both.

5.9 Auxiliary data

Additional data gathered during site survey before the captured event, or during the event, helps presentation of captured audio and video content in a spatiotemporally coherent fashion. This includes mesh models of the event site, photographs, image based models, depth maps, and more.

The detailed specification will be added in the revised version of this document.

6 Attributes of Processing Nodes

This chapter lists attributes and parameters for system-controlled audio and video processing modules, used with the scene description. Different dynamic instantiable and parameterizable processing modules can be used and combined for inter-machine application, described in a homogeneous scene. To get a powerful set of processing functionalities, existing processing frameworks can be selected, whereby the desired functionalities are offered as existing or to be developed plugins. Additionally, existing applications can be also integrated, if the interface, as described in the system architecture, will be used.

6.1 Input Parameter Types

This subsection specifies input parameter types used for the different processing nodes. Input parameters are used to control the behaviour of the processing nodes. The parameter types are modelled following the EBU QC[2] data model.

Input Parameter Attributes					
Attribute	Symbol	Description	Units	Type	Required = R, optional = O
Name		Name/Identifier of the parameter		String	R
Value		Value of the parameter. The type of this attribute is string, but the content must be interpreted according to the value of the type attribute. If type is not present, it may be treated as string.		String	R
Type		Data type		String	O
Representation		Additional specification of the representation, if necessary for the type (e.g., values expected as Hex, number of digits, etc.)		String	O
Static		Set to false, if the parameter may be changed after instantiation of the module. Defaults to true if omitted.		Boolean	O
Unit		The unit of the specified value (if applicable).		String	O
Default		The default value for the attribute. If omitted, the default is an empty string for strings, 0 for numerical values and false for boolean types.		String	O
User		Set to true, if the parameter may be presented to a user for interaction. Defaults to false.		Boolean	O
Description		A description of the attribute, e.g., to be used in user interfaces.		String	O
Range		A range of supported values, specified using a minimum and a maximum value separated by a semicolon and enclosed in square or round brackets (to denote inclusion/exclusion of the limits). If min or max are omitted, -/+ infinite is assumed for missing values.		String	O

[2] http://ebu.io/qc

6.2 Tool attributes

Attributes specifying the processing module, which may be used for automatic selection of modules.

Tool Attributes					
Attribute	Symbol	Description	Units	Type	Required = R, optional = O
Name		Name/Identifier of the module		String	R
Version		Version of the module		String	R
Manufacturer		Manufacturer of the module		String	O
InputTypes		Types of input streams supported		String [ ]	R
OutputType		Types of output streams supported		String [ ]	R

6.3 Content analysis

Content Analysis Module Attributes					
Attribute	Symbol	Description	Units	Type	Required = R, optional = O
Tool		Tool attributes.		as specified in Section 6.2	R
InputParameters		List of input parameters.		as specified in Section 6.1.	O
Active		Indicates whether the module is currently active.		Boolean	O

6.4 Content pre-selection

Content Pre-selection Module Attributes					
Attribute	Symbol	Description	Units	Type	Required = R, optional = O
Tool		Tool attributes.		as specified in Section 6.2	R
InputParameters		List of input parameters.		as specified in Section 6.1.	O
Active		Indicates whether the module is currently active.		Boolean	O
FeatureWeights		Specification of the weights of different features used for selection, specified as array of pairs of feature names and weights. It is recommended to normalise weights to [0.0;1.0].		{String, Real} [ ]	O
FeatureSelectors		List of pairs of feature name and selector for each of the features. A selector may be a range (as defined in Section 6.1) or one or a set of (comma separated) expected values.		{String, String} [ ]	O
FeatureCombinations		A list of feature combinations. Each combination is specified by the list of involved features, and a Boolean operator (AND, OR). Features may also be numeric indices to earlier positions in the array to allow for nesting. If omitted,		{String[ ], String } [ ]	O
StreamCombinations		Specification of streams treated as group in the selection, i.e. selection is only performed if all of the streams are selected. The combinations are specified as a list of stream groups. If omitted, or input streams are not listed, they are treated individually.		{UUID[ ]} [ ]	O

7 Attributes of Entity nodes

Entity nodes are used to represent real-world objects, which are visible/audible in the content being captured. The annotation of these entities forms an important link between different streams, as they indicate when semantically related content is covered, and thus options for alternative content or transitions exist.

7.1 Subject

Subject Parameters					
Parameter	Symbol	Description	Units	Type	Required = R, optional = O
Detection type		Specification of the type of detection: person or face		String	O
Name		Label/name of the subject. Multiple names may be provided, and URIs to controlled vocabularies may be used.		String*	O
Email		Email address		String	O
Social media accounts		A list of social media accounts, specifying user ID and an access token (if applicable).		{ID, token [] } []	O
Friends		A list of user IDs known to be friends on any social media platform.		String []	O

7.2 Object

Object Parameters					
Parameter	Symbol	Description	Units	Type	Required = R, optional = O
Class		Identifier of the object class. It is recommended to use one a URI pointing to a controlled vocabulary.		String	R
Name		Label/name of the object. *		String[]	O

7.3 Schedule

An item in a program schedule of one set. Schedule items are defined as specialisations of scenes, and thus support nesting. StartTime and StopTime of the schedule are available via its ancestor Scene.

Schedule item parameters					
Parameter	Symbol	Description	Units	Type	Required = R, optional = O
Title		The title of the item.		String	R
Tags		A tag describing the schedule item. Multiple tags may be provided, and URIs to controlled vocabularies may be used.		String [ ]	O

7.4 File

A file storing the content of one or more streams for a defined time segment. It is an input or output device (node), whereby StartTime and StopTime are controlled from the system architecture device properties. For mutiplexed streams, the requirements from the system architecture have to be considered.

File Parameters					
Parameter	Symbol	Description	Units	Type	Required = R, optional = O
Storage		Name of the storage volume		String	R
Name		Name of the file (relative to the volume root)		String	R
Streams		List of stream IDs contained in the file		UUID [ ]	O

8 Conclusion

This document presents the definition of the ICoSOLE scene representation model and parameters, concerning video, audio, and metadata, resulting from extensive discussions among ICoSOLE project partners. It is a reference that is used for implementation work of modules for capturing, processing, distributing, and displaying audio and video. Based on experience gained from extensive technical discussion in the course of the project execution, and in particular from first field trials, which are upcoming, the representation will be the current version of the deliverable has been refined with respect to the previous one (D3.1.1). The modifications concern revisions of the video and audio attributes (better modelling UGC) and the addition of attributes of processing and entity nodes.

9 References

[BIFS, 2005] ISO/IEC 14496-11:2005 - Information technology -- Coding of audio-visual objects -- Part 11: Scene description and application engine".

[EBU, 2014] Audio Definition Model, Metadata Specification. EBU Tech 3364, v1.0 , Jan. 2014.

[FascinatE D2.1.3] O. Schreer, P. Kauff, J.-F. Macq, P. Rondão Alface, J. Spille, R. Oldfield, G. Thomas, Final Specification of Generic Data Representation and Coding Scheme, FascinatE D2.1.3.

[Schoemake, 1994] Ken Schoemake “Euler Angle Conversion”, Graphics Gems IV, p222, ed Paul Heckbert, Academic Press, 1994.

10 Annex: Metadata Format Specification

10.1 Introduction

This annex specifies the formats to be used for the metadata streams. It defines the package header format, and the formats for specific types of metadata.

All metadata formats are specified using JSON Schema, using a representation that is as close to existing XML schema definition for metadata types where applicable.

10.2 Metadata Packet Format

10.2.1 Assumptions

- Metadata streams are identified by UUID
- A metadata stream may transport multiple types of metadata, the types must be identified, even if a stream happens to provide only one type of metadata
- Metadata are time indexed, specifying the time from when on the metadata is considered valid. An time may be specified, if omitted, the metadata values are assumed to be valid until replaced by an update of the same type of metadata
- Metadata packets are not labelled with IDs in the stream (but are likely to get an identifier in the data store), thus the pair (timestamp, metadata type) is assumed to uniquely identify a packet within a stream, and the triple (stream UUID, timestamp, metadata type) will be globally unique.
- The UUID of the source device can be included.

10.2.2 Metadata Packet

Each metadata update consists of a JSON structure called ICoSOLE metadata packet (*imp*). The packet contains three mandatory attributes, of which the first two usually need to be inspected to handle the packet:

- Timestamp (t): start time of validity
- Metadata type (m): One of a controlled set of short strings identifying the different types of metadata in the system. This type will also define the representation of the payload.
- Source (s): The UUID of the device generating the metadata.
- Payload (p): The metadata in the specified format.

Optionally, the following attributes may be included:

- End timestamp (e): end time of validity
- Absolute location (l) wrt WGS84, specified in latitude, longitude (in decimal degrees) and altitude (in meters)
- Status (st): one of `static`, `dynamic`. If omitted, dynamic is assumed.

Metadata packets are packed in array, in order to allow simultaneous transmission of multiple packets at once. If an array contains multiple packets, the packets MUST be sorted in by ascending time stamps in order to enable a consuming application to process the packets in order.

10.2.3 JSON Schema

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "http://www.icosole.eu/metadata/packets",
  "type": "array",
  "items": {
      "id": "http://www.icosole.eu/metadata/packet",
      "type": "object",
```



```
      "properties": {
        "t": {
          "id": "http://www.icosole.eu/metadata/packet/t",
          "type": "integer"
        },
        "e": {
          "id": "http://www.icosole.eu/metadata/packet/e",
          "type": "integer"
        },
        "m": {
          "id": "http://www.icosole.eu/metadata/packet/m",
          "type": "string"
        },
        "l": { "$ref": "http://www.icosole.eu/metadata/location"},
        "st": {
          "id": "http://www.icosole.eu/metadata/packet/s",
          "type": "string",
          "default": "dynamic",
          "enum": [
          "static",
          "dynamic"
          ]
        },
        "s": {
          "id": "http://www.icosole.eu/metadata/packet/sourceID",
          "type": "string",
          "pattern":     "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
9]{12}$"
        },
        "p": {
          "id": "http://www.icosole.eu/metadata/packet/p",
          "type": "object",
          "oneOf": [
          {"$ref": ""}
          ]
        }
      },
      "required": [
        "t",
        "m",
        "p"
      ]
  }
}
```

10.2.4 Metadata type identifiers

The values to identify the metadata type of the package are defined by the relative part of the IDs of the respective types in the schema.

The first part of the ID after the prefix `http://www.icosole.eu/metadata/` shall be used to identify metadata types in the packet.

Only the top level identifier is valid, unless explicitly defined for a specific type of metadata.

10.2.5 Example

Single packet:

```
[
    {
```



```
        "t": 12345679,
        "e": 12350000,
        "m": "sensor",
        "s": "0000-0000-0000",
        "st": "dynamic",
        "p": {
            "motion": {
                "accelerometer": [
                    0.1,
                    0.2,
                    0.4
                ]
            }
        }
    }
]
```

Multiple packets:

```
[
    {
        "t": 12345679,
        "e": 12350000,
        "m": "sensor",
        "s": "0000-0000-0000",
        "st": "dynamic",
        "p": {
            "motion": {
                "accelerometer": [
                    0.1,
                    0.2,
                    0.4
                ]
            }
        }
    },
    {
        "t": 12345680,
        "m": "sensor",
        "s": "0000-0000-0000",
        "st": " dynamic",
        "p": {
            "motion": {
                "gyroscope": [
                    0.5,
                    0.6,
                    0.7
                ]
            }
        }
    }
]
```


10.3 Basic metadata structures

10.3.1 Introduction

This section defines data structures used for different types of metadata.

This section defines:

- Points: 2D or 3D points in a Cartesian coordination system (e.g, image coordinates, local coordinate system)
- Locations are geographic coordinates in WGS84

10.3.2 JSON Schema

Points

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",

    "definitions": {
        "point2d": {
            "id": "http://www.icosole.eu/metadata/point2d",
            "type": "object",
            "properties": {
              "x": {
                "id": "http://www.icosole.eu/metadata/point2d/x",
                "type": "number"
              },
              "y": {
                "id": "http://www.icosole.eu/metadata/point2d/y",
                "type": "number"
              }
            }
        },
        "point3d": {
            "id": "http://www.icosole.eu/metadata/point3d",
            "type": "object",
            "properties": {
              "x": {
                "id": "http://www.icosole.eu/metadata/point3d/x",
                "type": "number"
              },
              "y": {
                "id": "http://www.icosole.eu/metadata/point3d/y",
                "type": "number"
              },
              "z": {
                "id": "http://www.icosole.eu/metadata/point3d/z",
                "type": "number"
              }
            }
        }
    }


}
```

Location

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
```

```
    "id": "http://www.icosole.eu/metadata/location",
    "type": "object",
    "properties": {
      "lat": {
        "id": "http://www.icosole.eu/metadata/location/lat",
        "type": "number"
      },
      "lon": {
        "id": "http://www.icosole.eu/metadata/location/lon",
        "type": "number"
      },
      "alt": {
        "id": "http://www.icosole.eu/metadata/location/alt",
        "type": "number"
      }
    }
}
```

Scene/Set

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "http://www.icosole.eu/metadata/set",
  "type": "object",
  "properties": {
    "uuid": {
      "id": "http://www.icosole.eu/metadata/set/id",
      "type": "string",
      "pattern":         "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
9]{12}$"
    },
    "parent": {
      "id": "http://www.icosole.eu/metadata/set/parent",
      "type": "string",
      "pattern":         "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
9]{12}$"
    },
    "begin": {
      "id": "http://www.icosole.eu/metadata/set/begin",
      "type": "number"
    },
    "end": {
      "id": "http://www.icosole.eu/metadata/set/end",
      "type": "number"
    },
    "extent": {
      "id": "http://www.icosole.eu/metadata/extent",
      "type": "array",
      "items": {
        "ref": "http://www.icosole.eu/metadata/location"
      }
    },
    "required": [
      "uuid"
    ]
  }
}
```


10.4 Device metadata

10.4.1 Introduction

This section defines general metadata of capture devices (as defined in D4.3). These device properties are considered static metadata.

It also defines structures for specifying file names capturing recordings from devices. It specifies the filename, optionally the name the storage (to which the filename is then assumed to be relative), and the IDs of the essence streams recorded in the file.

10.4.2 JSON Schema

Device properties

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "http://www.icosole.eu/metadata/device",
  "type": "object",
  "properties": {
    "objectID": {
      "id": "http://www.icosole.eu/metadata/device/objectID",
      "type": "string",
      "pattern":         "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}$"
    },
    "parent": {
      "id": "http://www.icosole.eu/metadata/device/parent",
      "type": "string",
      "pattern":         "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}$"
    },
    "name": {
      "id": "http://www.icosole.eu/metadata/device/name",
      "type": "string"
    },
    "type": {
      "id": "http://www.icosole.eu/metadata/device/type",
      "type": "string"
    }
    "version": {
      "id": "http://www.icosole.eu/metadata/device/version",
      "type": "string"
    },
    "startTime": {
      "id": "http://www.icosole.eu/metadata/device/startTime",
      "type": "string"
    },
    "endTime": {
      "id": "http://www.icosole.eu/metadata/device/endTime",
      "type": "string"
    },
    "inputStreams": {
      "id": "http://www.icosole.eu/metadata/device/inputStreams",
      "type": "array",
      "minItems": 1,
      "items": {
      "type": "string",
      "pattern":         "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}$"
```



```
          }
    },
    "outputStreams": {
      "id": "http://www.icosole.eu/metadata/device/outputStreams",
      "type": "array",
          "minItems": 1,
          "items": {
                "type": "string",
                "pattern":    "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-
F0-9]{12}$"
          }
    },
    "parameters": {
      "id": "http://www.icosole.eu/metadata/device/parameters",
      "type": "array",
      "items": {
        "id": "http://www.icosole.eu/metadata/device/parameter",
        "type": "object",
        "properties": {
          "key": {
            "id": "http://www.icosole.eu/metadata/device/parameter/key",
            "type": "string"
          },
          "value": {
            "id": "http://www.icosole.eu/metadata/device/parameter/value",
            "type": "string"
          }
      }
      }
    }
  },
  "required": [
    "objectID",
    "name"
  ]
}
```

File names

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "http://www.icosole.eu/metadata/file",
  "type": "object",
  "properties": {
    "id": {
      "id": "http://www.icosole.eu/metadata/file/id",
      "type": "string"
    },
    "storage": {
      "id": "http://www.icosole.eu/metadata/file/storage",
      "type": "string"
    },
    "name": {
      "id": "http://www.icosole.eu/metadata/file/name",
      "type": "string"
    },
    "streams": {
      "ref": "http://www.icosole.eu/metadata/device/outputStreams"
```



```
    }
  }
}
```

10.4.3 Example instance

Device properties

```
{
    "objectID": "UUID",
    "parent": "UUID",
    "name": "name",
    "version": "ver",
    "startTime": "...",
    "endTime": "...",
    "inputStreams": ["UUID1","UUID2"],
    "outputStreams": ["UUID3","UUID4"],
    "parameters": [{"key": "OSversion", "value": "abc"}]
}
```

File names

```
{
  "id": "012345",
  "storage": "ugc_dranouter",
  "name": "path/file.mp4",
  "streams": [
    "01xxx", "02xxx"
  ]
}
```

10.5 Video metadata

10.5.1 Introduction

The format models the video metadata defined in D3.1.1.

10.5.2 JSON Schema

Camera

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "http://www.icosole.eu/metadata/camera",
  "type": "object",
  "properties": {
    "camID": {
      "id": "http://www.icosole.eu/metadata/camera/camID",
      "type": "string"
      "pattern":         "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
9]{12}$"
    },
    "fps": {
      "id": "http://www.icosole.eu/metadata/camera/fps",
      "type": "number"
    },
    "geometry": {
      "id": "http://www.icosole.eu/metadata/camera/geometry",
      "type": "object",
      "properties": {
```



```
      "description": {
        "id": "http://www.icosole.eu/metadata/camera/geometry/description",
        "type": "string"
      },
      "pixelPitch": {
        "id": "http://www.icosole.eu/metadata/camera/geometry/pixelPitch",
        "type": "array",
        "minItems": 2,
        "maxItems": 2,
        "items": {"type": "number"}
      },
      "imageSize": {
        "id": "http://www.icosole.eu/metadata/camera/geometry/imageSize",
        "type": "array",
         "minItems": 2,
        "maxItems": 2,
        "items": {"type": "integer"}
     },
      "imageOrigin": {
        "id": "http://www.icosole.eu/metadata/camera/geometry/imageOrigin",
        "default": "topLeft",
        "enum": [
          "topLeft",
          "bottomLeft"
        ]
      },
      "required": [
          "imageSize",
          "imageOrigin"
      ]
    },
    "required": [
      "camID",
      "fps"
    ]
  },
  "extrinsic": {
    "id": "http://www.icosole.eu/metadata/camera/extrinsic",
    "type": "object",
    "properties": {
      "opticalCenter": { "$ref": "http://www.icosole.eu/metadata/point3d" },
      "panTiltRoll": {
        "id": "http://www.icosole.eu/metadata/camera/extrinsic/panTiltRoll",
        "type": "array",
        "minItems": 3,
        "maxItems": 3,
        "items": {"type": "number"}
        },
      "eulerAngles": {
        "id": "http://www.icosole.eu/metadata/camera/extrinsic/eulerAngles",
        "type": "array",
        "minItems": 3,
        "maxItems": 3,
        "items": {"type": "number"}

      },
```

```
        "eulerOrder": {
          "id": "http://www.icosole.eu/metadata/camera/extrinsic/eulerOrder",
          "type": "string"
        },
        "required": [
            "opticalCenter"
        ]
      }
    },
    "lens": {
      "id": "http://www.icosole.eu/metadata/camera/lens",
      "type": "object",
      "properties": {
        "geometry": {
          "id": "http://www.icosole.eu/metadata/camera/lens/geometry",
          "type": "object",
          "properties": {
            "description": {
              "id": "http://www.icosole.eu/metadata/camera/lens/geometry/description",
              "type": "string"
            },
            "model": {
              "id": "http://www.icosole.eu/metadata/camera/lens/geometry/model",
              "type": "string"
            },
            "focalLength": {
              "id": "http://www.icosole.eu/metadata/camera/lens/geometry/focalLength",
              "type": "number"
            },
            "fov": {
              "id": "http://www.icosole.eu/metadata/camera/lens/geometry/fov",
              "type": "array",
              "minItems": 2,
              "maxItems": 2,
              "items": {"type": "number"}
            },
            "required": [
                "fov"
            ],
            "distortion": {
              "id": "http://www.icosole.eu/metadata/camera/lens/geometry/distortion",
              "type": "object",
              "properties": {
                "num": {
                  "id": "http://www.icosole.eu/metadata/camera/lens/geometry/distortion/num",
                  "type": "array",
                  "minItems": 4,
                  "maxItems": 4,
                  "items": {"type": "number"}
                },
                "denom": {
                  "id": "http://www.icosole.eu/metadata/camera/lens/geometry/distortion/denom",
                  "type": "array",
                  "minItems": 3,
                  "maxItems": 3,
                  "items": {"type": "number"}
```



```
                }
              }
            }
          }
        },
        "assembly": {
          "id": "http://www.icosole.eu/metadata/camera/lens/assembly",
          "type": "object",
          "properties": {
            "pc": { "$ref": "http://www.icosole.eu/metadata/point2d" },
            "cod": { "$ref": "http://www.icosole.eu/metadata/point2d" },
            "aspectRatio": {
              "id": "http://www.icosole.eu/metadata/camera/lens/assembly/aspectRatio",
              "type": "number"
            },
            "skew": {
              "id": "http://www.icosole.eu/metadata/camera/lens/assembly/skew",
              "type": "number"
            },
            "required": [
                "pc"
            ]
          }
        },
        "photometric": {
          "id": "http://www.icosole.eu/metadata/camera/lens/photometric",
          "type": "object",
          "properties": {
            "aperture": {
              "id": "http://www.icosole.eu/metadata/camera/lens/photometric/aperture",
              "type": "integer"
            },
            "vignetting": {
              "id": "http://www.icosole.eu/metadata/camera/lens/photometric/vignetting",
              "type": "array",
              "minItems": 4,
              "maxItems": 4,
              "items": {"type": "number"}
            }
          }
        }
      }
    },
    "photometric": {
      "id": "http://www.icosole.eu/metadata/camera/photometric",
      "type": "object",
      "properties": {
        "colorSystem": {
          "id": "http://www.icosole.eu/metadata/camera/photometric/colorSystem",
          "type": "string"
        },
        "bitDepth": {
          "id": "http://www.icosole.eu/metadata/camera/photometric/bitDepth",
          "type": "integer"
        },
        "primaries": {
```



```
          "id": "http://www.icosole.eu/metadata/camera/photometric/primaries",
          "type": "array",
          "minItems": 3,
          "maxItems": 3,
          "items": {"type": "number"}
        },
        "whitePoint": {
          "id": "http://www.icosole.eu/metadata/camera/photometric/whitePoint",
          "type": "integer"
        },
        "blackLevel": {
          "id": "http://www.icosole.eu/metadata/camera/photometric/blackLevel",
          "type": "number"
        },
        "gain": {
          "id": "http://www.icosole.eu/metadata/camera/photometric/gain",
          "type": "array",
          "minItems": 3,
          "maxItems": 3,
          "items": {"type": "number"}

        },
        "integrationTime": {
          "id": "http://www.icosole.eu/metadata/camera/photometric/integrationTime",
          "type": "number"
        },
        "gamma": {
          "id": "http://www.icosole.eu/metadata/camera/photometric/gamma",
          "type": "integer"
        },
        "curveSamples": {
          "id": "http://www.icosole.eu/metadata/camera/photometric/curveSamples",
          "type": "object",
          "properties": {}
        },
        "colorMatrix": {
          "id": "http://www.icosole.eu/metadata/camera/photometric/colorMatrix",
          "type": "object",
          "properties": {}
        },
        "required": [
            "colorSystem",
            "bitDepth"
        ]
      }
    }
  }
}
```

Video coding

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "http://www.icosole.eu/metadata/videoCoding",
  "type": "object",
  "properties": {
    "videoCoding": {
      "id": "http://www.icosole.eu/metadata/videoCoding",
```



```
    "type": "object",
    "properties": {
      "format": {
        "id": "http://www.icosole.eu/metadata/videoCoding/format",
        "type": "string"
      },
      "profile": {
        "id": "http://www.icosole.eu/metadata/videoCoding/profile",
        "type": "string"
      },
      "pictureSize": {
        "id": "http://www.icosole.eu/metadata/videoCoding/pictureSize",
        "type": "array",
        "minItems": 2,
        "maxItems": 2,
        "items": {"type": "integer"}
      },
      "numTiles": {
        "id": "http://www.icosole.eu/metadata/videoCoding/numTiles",
        "type": "array",
        "minItems": 2,
        "maxItems": 2,
        "items": {"type": "integer"}
      },
      "tileSize": {
        "id": "http://www.icosole.eu/metadata/videoCoding/tileSize",
        "type": "array",
        "minItems": 2,
        "maxItems": 2,
        "items": {"type": "integer"}
      },
      "colorSystem": {
        "id": "http://www.icosole.eu/metadata/videoCoding/colorSystem",
        "type": "string"
      },
      "bitDepth": {
        "id": "http://www.icosole.eu/metadata/videoCoding/bitDepth",
        "type": "integer"
      },
      "options": {
        "id": "http://www.icosole.eu/metadata/videoCoding/options",
        "type": "array",
        "items": {
          "id": "http://www.icosole.eu/metadata/videoCoding/option",
          "type": "object",
          "properties": {
            "key": {
              "id": "http://www.icosole.eu/metadata/videoCoding/option/key",
              "type": "string"
            },
             "value": {
              "id": "http://www.icosole.eu/metadata/videoCoding/option/value",
              "type": "string"
            }
            }
        }
```



```
        }
      },
      "required": [
        "format",
        "profile"
      ]
    }
  },
  "required": [
    "videoCoding"
  ]
}
```

10.5.3 Example instance

Camera

```
{
    "camID": "uuid",
    "fps": 12.34,
    "geometry":{
        "description": "abc",
        "pixelPitch": [0.1234,0.1234],
        "imageSize": [1920,1080],
        "imageOrigin": "topLeft",
    },
    "extrinsic": {
        "opticalCenter": { "x": 0, "y": 0, "z": 0 },
        "panTiltRoll": [0.1, 0.2, 0.3],
        "eulerAngles": [0.1,0.2,0.3],
        "eulerOrder": "XYZr",
    },
    "lens":{
        "geometry": {
            "description": "string",
            "model": "rectilinear",
            "focalLength": 0.111,
            "fov": [0.111,0.111],
            "distortion": {
                "num": [0.1,0.2,0.3,0.4],
                "denom": [0.1,0.2]
            }
        },
        "assembly": {
            "pc": { "x": 0, "y": 0 },
            "cod": { "x": 0, "y": 0 },
            "aspectRatio": 0.123,
            "skew": 0.1234
        },
        "photometric": {
            "aperture": 1.0,
            "vignetting": [0.1,0.2,0.3,0.4]
        }
    },
    "photometric": {
        "colorSystem": "RGB444",
        "bitDepth": 8,
        "primaries": [700.1,600.2,500.1],
```



```
        "whitePoint": 255.0,
        "blackLevel": 5.2,
        "gain": [0.1,0.2,0.3],
        "integrationTime": 0.001,
        "gamma": 1.0,
        "curveSamples": {},
        "colorMatrix": {},
    }
}
```

Video coding

```
{
    "videoCoding": {
        "format": "mpeg-4",
        "profile": "main",
        "pictureSize": [1920,1080],
        "numTiles": [1,1],
        "tileSize": [1920,1080],
        "colorSystem": "RGB",
        "bitDepth": 8,
        "options": [{"key":"1", "value":"sdds"}],
    }
}
```

10.5.4 Open issues

- Some changes/modelling issues that should be considered when creating the revised scene representation (D3.1.2)
    - IDs should be UUIDs
    - Scene
        - time origin needed?
    - Camera
        - Pixel pitch h/v: should not be required, not known for UGC
        - extrinsic angle: split degrees/euler, and require euler order only when needed
    - Video coding
        - tiles should be optional and default to one
- Camera/curvesamples and colormatrix need to be modelled in detail
- Some strings could be replace by enumerations

10.6 Audio control metadata

10.6.1 Introduction

The JSON schema here is based on the audio device parameters described in D3.1.1. This schema is used to describe the audio capture device parameters, and not the actual object-based (or scene or channel-based for that matter) audio, which requires a human-controlled processing stage. So this schema is intended for the initial raw streams of audio from the capture devices.

10.6.2 JSON Schema

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "http://www.icosole.eu/metadata/audio",
  "type": "object",
  "properties": {
    "audioDeviceID": {
```



```
      "id": "http://www.icosole.eu/metadata/audio/audioDeviceID",
      "type": "string",
      "pattern":         "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
9]{12}$"
    },
    "make": {
      "id": "http://www.icosole.eu/metadata/audio/make",
      "type": "string"
    },
    "model": {
      "id": "http://www.icosole.eu/metadata/audio/model",
      "type": "string"
    },
    "deviceType": {
      "id": "http://www.icosole.eu/metadata/audio/deviceType",
      "default": "microphone",
      "enum": [
        "microphone",
        "onStage",
        "offStage",
        "other"
      ]
    },
    "subject" : {
      "id": "http://www.icosole.eu/metadata/audio/subject",
      "type": "array",
      "minItems": 0,
      "maxItems": 255,
      "items": {"type": "string"}
    },
    "micParams": {
      "id": "http://www.icosole.eu/metadata/audio/micParams",
      "type": "object",
      "properties": {
        "pattern": {
          "id": "http://www.icosole.eu/metadata/audio/micParamas/pattern",
          "type": "string"
        },
        "panTiltRoll": {
          "id": "http://www.icosole.eu/metadata/audio/micParams/panTiltRoll",
          "type": "array",
          "minItems": 3,
          "maxItems": 3,
          "items": {"type": "number"}
        },
        "eulerOrder": {
          "id": "http://www.icosole.eu/metadata/audio/micParams/eulerOrder",
          "type": "string"
        }
      }
    },
    "signals": {
      "id": "http://www.icosole.eu/metadata/audio/signals",
      "type": "object",
      "properties": {
        "coding" : {
```



```
          "id": "http://www.icosole.eu/metadata/audio/signals/coding",
          "default": "PCM",
          "type": "string"
        },
        "bitDepth" : {
          "id": "http://www.icosole.eu/metadata/audio/signals/bitDepth",
          "default": 24,
          "type": "integer"
        },
        "sampleRate" : {
          "id": "http://www.icosole.eu/metadata/audio/signals/sampleRate",
          "default": 48000,
          "type": "integer"
        },
        "numTracks": {
          "id": "http://www.icosole.eu/metadata/audio/signals/numTracks",
          "default": 1,
          "type": "integer"
        }
      }
    }
  },
  "required": [
    "audioDeviceID",
    "signals"
  ]
}
```

10.6.3 Example

```
{
  "audioDeviceID": "12345678-1234-1234-1234-1234567890ab",
  "make": "Sennheiser",
  "model": "MKH-20",
  "deviceType": "microphone",
  "subject": ["cello"],
  "micParams": {
    "pattern": "Omni",
    "panTiltRoll": [-90.0, 0.0, 0.0],
    "eulerOrder": "PTR"
  },
  "signals": {
    "numTracks": 1
  }
}
```

10.7 Sensor metadata

10.7.1 Introduction

Sensor metadata of mobile devices are modelled according to the Android list of sensor types (they translate to IDs in the Android API by prefixing TYPE_ and changing the name to upper case).

10.7.2 JSON Schema

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
```



```
  "id": "http://www.icosole.eu/metadata/sensor",
  "type": "object",
  "properties": {
    "motion": {
      "id": "http://www.icosole.eu/metadata/sensor/motion",
      "type": "object",
      "properties": {
        "accelerometer": {
          "id": "http://www.icosole.eu/metadata/sensor/motion/accelerometer",
          "type": "array",
          "minItems": 3,
          "maxItems": 3,
          "items": {"type": "number"}
        },
        "gravity": {
          "id": "http://www.icosole.eu/metadata/sensor/motion/gravity",
          "type": "array",
          "minItems": 3,
          "maxItems": 3,
          "items": {"type": "number"}
        },
        "gyroscope": {
          "id": "http://www.icosole.eu/metadata/sensor/motion/gyroscope",
          "type": "array",
          "minItems": 3,
          "maxItems": 3,
          "items": {"type": "number"}
        },
        "gyroscope_uncalibrated": {
          "id": "http://www.icosole.eu/metadata/sensor/motion/gyroscope_uncalibrated",
          "type": "array",
          "minItems": 6,
          "maxItems": 6,
          "items": {"type": "number"}
        },
        "rotation": {
          "id": "http://www.icosole.eu/metadata/sensor/motion/rotation",
          "type": "array",
          "minItems": 4,
          "maxItems": 4,
          "items": {"type": "number"}
        },
        "step_counter": {
          "id": "http://www.icosole.eu/metadata/sensor/motion/step_counter",
          "type": "integer"
        }
      }
    },
    "environment": {
      "id": "http://www.icosole.eu/metadata/sensor/environment",
      "type": "object",
      "properties": {
        "ambient_temperature": {
          "id": "http://www.icosole.eu/metadata/sensor/environment/ambient_temperature",
          "type": "integer"
        },
```



```
        "light": {
          "id": "http://www.icosole.eu/metadata/sensor/environment/light",
          "type": "integer"
        },
        "pressure": {
          "id": "http://www.icosole.eu/metadata/sensor/environment/pressure",
          "type": "integer"
        },
        "relative_humidity": {
          "id": "http://www.icosole.eu/metadata/sensor/environment/relative_humidity",
          "type": "integer"
        }
      }
    },
    "position": {
      "id": "http://www.icosole.eu/metadata/sensor/position",
      "type": "object",
      "properties": {
        "game_rotation_vector": {
          "id": "http://www.icosole.eu/metadata/sensor/position/game_rotation_vector",
          "type": "array",
          "minItems": 3,
          "maxItems": 3,
          "items": {"type": "number"}
        },
        "geomagnetic_rotation_vector": {
          "id": "http://www.icosole.eu/metadata/sensor/position/geomagnetic_rotation_vector",
          "type": "array",
          "minItems": 3,
          "maxItems": 3,
          "items": {"type": "number"}
        },
        "magnetic_field": {
          "id": "http://www.icosole.eu/metadata/sensor/position/magnetic_field",
          "type": "array",
          "minItems": 3,
          "maxItems": 3,
          "items": {"type": "number"}
        },
        "magnetic_field_uncalibrated": {
          "id": "http://www.icosole.eu/metadata/sensor/position/magnetic_field_uncalibrated",
          "type": "array",
          "minItems": 6,
          "maxItems": 6,
          "items": {"type": "number"}
        },
        "proximity": {
          "id": "http://www.icosole.eu/metadata/sensor/position/proximity",
          "type": "integer"
        }
      }
    }
  }
}
```


10.7.3 Example

```
{
    "motion": {
        "accelerometer": [0.1,0.2,0.3],
        "gravity": [0.1,0.2,0.3],
        "gyroscope": [0.1,0.2,0.3],
        "gyroscope_uncalibrated": [0.1,0.2,0.3,0.4,0.5,0.6],
        "linear_acceleration": [0.1,0.2,0.3],
        "rotation": [0.1,0.2,0.3,0.0],
        "step_counter": 1
    },
    "environment": {
        "ambient_temperature": 0.0,
        "light": 0.0,
        "pressure": 0.0,
        "relative_humidity": 0.0
    },
    "position": {
        "game_rotation_vector": [0.1,0.2,0.3],
        "geomagnetic_rotation_vector": [0.1,0.2,0.3],
        "magnetic_field": [0.1,0.2,0.3],
        "magnetic_field_uncalibrated": [0.1,0.2,0.3,1.1,1.2,1.3],
        "proximity": 0.0
    }
}
```

10.8 Persons, Faces, Links

10.8.1 Introduction

This section defines metadata for visual analysis results, in particular detection of objects, persons and faces, and linking between views, by temporal, spatial and visual overlap.

Image coordinates are normalised to image width and height, specified as number in the interval [0,1].

The coordinates of a box are specified as upper left x, upper left y, lower right x, lower right y

The coordinates of a polygon are specified as a list of points x and y coordinates in counter-clockwise order.

10.8.2 JSON Schema

Object, Person, Face

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "http://www.icosole.eu/metadata/object",
  "type": "object",
  "properties": {
    "@type": {
      "id": "http://www.icosole.eu/metadata/object/@type",
      "type": "string",
      "enum": [
          "object",
          "face",
          "person"
      ]
    },
    "@id": {
      "id": "http://www.icosole.eu/metadata/object/@id",
      "type": "string"
```



```
    },
    "Box": {
      "id": "http://www.icosole.eu/metadata/object/Box",
      "type": "array",
      "minItems": 4,
      "maxItems": 4,
      "items": { "type": "number" }
    }
  },
  "required": [
    "@type",
    "@id",
    "Box"
  ]
}
```

Links

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "http://www.icosole.eu/metadata/link",
  "type": "object",
  "properties": {
    "Relations": {
          "id": "http://www.icosole.eu/metadata/links/Relations",
      "type": "array",
      "items": {
        "id": "http://www.icosole.eu/metadata/links/Relation",
        "type": "object",
        "properties": {
          "@target": {
            "id": "http://www.icosole.eu/metadata/links/Relation/@type",
            "type": "string",
            "enum": [
              "time",
              "location",
              "visual"
            ]
          },
          "@target": {
            "id": "http://www.icosole.eu/metadata/links/Relation/@target",
            "type": "string",
            "pattern":   "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
9]{12}$"
          },
          "@similarity": {
            "id": "http://www.icosole.eu/metadata/links/Relation/@similarity",
            "type": "number"
          },
          "@confidence": {
            "id": "http://www.icosole.eu/metadata/links/Relation/@confidence",
            "type": "number"
          },
          "SourcePolygon": {
            "id": "http://www.icosole.eu/metadata/links/Relation/SourcePolygon",
            "type": "array",
            "minItems": 6,
            "items": { "type": "number" }
```



```
          },
          "TargetPolygon": {
            "id": "http://www.icosole.eu/metadata/links/Relation/TargetPolygon",
            "type": "array",
            "minItems": 6,
            "items": { "type": "number" }
          }
        },
        "required": [
          "@type",
          "@target"
        ]
      }
    },
    "required": [
        "Relations"
    ]
  }
}
```

10.8.3 Example

Object, person, face

```
{
    "@type": "person",
    "@id": "http://some.object",
    "Box": [0, 0, 0.2, 0.41]
}
```

Links

```
{
    "Relations": [
        {
            "@type": "visual",            "@target": "UUID",
            "@similarity": 0.3,
            "@confidence": 0.8,
            "SourcePolygon": [0, 0, 123, -122, 340, -12],
            "TargetPolygon": [0, 0, 123, -122, 340, -12]
        }
    ]

}
```

10.9 Quality metadata

10.9.1 Introduction

The format is based on MPEG-7 Part 5 Amd5, implementing the EBU QC output data model[3]. It defines a subset of QCItemResultType.

10.9.2 JSON Schema

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "http://www.icosole.eu/metadata/quality",
```

[3] http://www.ebu.io/qc/


```
"type": "object",
"properties": {
  "Results": {
    "id": "http://www.icosole.eu/metadata/qualityItems",
    "type": "array",
    "items": {
      "id": "http://www.icosole.eu/metadata/qualityItem",
      "type": "object",
      "properties": {
        "Name": {
          "id": "http://www.icosole.eu/metadata/quality/Name",
          "type": "object",
          "properties": {
            "@href": {
              "id": "http://www.icosole.eu/metadata/quality/Name/@href",
              "type": "string"
            },
            "#text": {
              "id": "http://www.icosole.eu/metadata/quality/Name/#text",
              "type": "string"
            }
          },
          "required": [
            "@href"
          ]
        },
        "CheckResult": {
          "id": "http://www.icosole.eu/metadata/quality/CheckResult",
          "type": "string"
        },
        "Output": {
          "id": "http://www.icosole.eu/metadata/quality/Outputs",
          "type": "array",
          "items": {
            "id": "http://www.icosole.eu/metadata/quality/Output",
            "type": "object",
            "properties": {
              "@name": {
                "id": "http://www.icosole.eu/metadata/quality/Output/@name",
                "type": "string"
              },
              "@unit": {
                "id": "http://www.icosole.eu/metadata/quality/Output/@unit",
                "type": "string"
              },
              "@track": {
                "id": "http://www.icosole.eu/metadata/quality/Output/@track",
                "type": "string"
              },
              "@valueRange": {
                "id": "http://www.icosole.eu/metadata/quality/Output/@valueRange",
                "type": "string"
              },
              "@type": {
                "id": "http://www.icosole.eu/metadata/quality/Output/@type",
                "type": "string"
```



```
            },
            "@representation": {
              "id": "http://www.icosole.eu/metadata/quality/Output/@representation",
              "type": "string"
            },
            "@annotation": {
              "id": "http://www.icosole.eu/metadata/quality/Output/@annotation",
            "type": "string"
            },
            "@confidence": {
              "id": "http://www.icosole.eu/metadata/quality/Output/@confidence",
              "type": "number"
            },
            "@severity": {
              "id": "http://www.icosole.eu/metadata/quality/Output/@severity",
              "type": "integer"
            },
            "#text": {
              "id": "http://www.icosole.eu/metadata/quality/Output/#text",
              "type": "string"
            },
            "required": [
                "@name",
                "#text"
            ]
          }
        }
      },
      "Relevance": {
        "id": "http://www.icosole.eu/metadata/quality/Relevance",
        "type": "integer"
      },
      "ExecutionStatus": {
        "id": "http://www.icosole.eu/metadata/quality/ExecutionStatus",
        "type": "string"
      },
      "ResultDescription": {
        "id": "http://www.icosole.eu/metadata/quality/ResultDescription",
        "type": "string"
      },
      "Annotation": {
        "id": "http://www.icosole.eu/metadata/quality/Annotation",
        "type": "string"
      },
      "MaxSeverity": {
        "id": "http://www.icosole.eu/metadata/quality/MaxSeverity",
        "type": "integer"
      },
      "DetectionMethod": {
        "id": "http://www.icosole.eu/metadata/quality/DetectionMethod",
        "type": "string",
        "enum": [
            "human",
            "automatic",
            "assisted"
        ]
```



```
          }
        },
        "required": [
          "Name"
        ]
      }
    }
  },
  "required": [
    "Results"
  ]
}
```

10.9.3 Example

```
{
  "Results":
    [
      {
        "Name": {
            "@href": "http://www.ebu.io/id",
            "#text": "Name"
        },
        "CheckResult": "true",
        "Output": [
            {
             "@name": "param",
             "@unit": "dB",
             "@track": "video",
             "@valueRange": "[0,100]",
             "@type": "real",
             "@representation": "",
             "@annotation": "",
             "@confidence": 0.7,
             "@severity": 2,
             "#text": ""
            }
            ],
        "Relevance": 5,
        "ExecutionStatus": "complete",
        "ResultDescription": {},
        "Annotation": {},
        "MaxSeverity": 10,
        "DetectionMethod": "automatic"
      }
    ]
}
```

10.10 Tagging, ratings, annotations

10.10.1 Introduction

This section describes a framework for annotating tags, likes, ratings etc. This mechanism can be used for different purposes, such as

- Automatic analysis results, e.g., key frames, detected objects
- Content selection by production team
- User annotations

- Social media feedback

All user IDs are internal to the system. External user IDs (e.g., from social media platform) are mapped to internal IDs before metadata are inserted.

Tags can be described by one of the following means:

- label: a free text label for the tag
- userlabel: A free text label of a user created tag. User labels are indexed and can be accessed for tag completion.
- link: an URI for a tag, to be used in addition or alternatively to the label

An annotation has an optional *about* element, referring to a metadata stream UUID, a start time and optionally an end time. The semantics are as follows:

- If *about* is not present, the annotation is inserted in the metadata stream as indicated in the packet header, and is valid for the time specified.
- If *about* is present, the packet describes the annotation event in the metadata, and the metadata is additionally inserted retrospectively in the stream indicated in *about*, valid for the time indicated there. This time MUST be before the time of the annotation.

10.10.2 JSON Schema

Annotations

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "http://www.icosole.eu/metadata/annot",
  "type": "object",
  "properties": {
    "tags": {
      "id": "http://www.icosole.eu/metadata/annot/tags",
      "type": "array",
      "items": {
        "id": "http://www.icosole.eu/metadata/annot/tag",
        "type": "object",
        "properties": {
          "link": {
            "id": "http://www.icosole.eu/metadata/annot/tag/link",
            "type": "string"
          },
          "label": {
            "id": "http://www.icosole.eu/metadata/annot/tag/label",
            "type": "string"
          },
          "userlabel": {
            "id": "http://www.icosole.eu/metadata/annot/tag/userlabel",
            "type": "string"
          }
        }
      }
    },
    "about": {
       "id": "http://www.icosole.eu/metadata/annot/about",
       "type": "object",
       "properties": {
           "stream": {
               "id": "http://www.icosole.eu/metadata/about/stream",
               "type": "integer",
```



```
                "pattern":   "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}$"
            },
            "t": {
               "id": "http://www.icosole.eu/metadata/about/t",
               "type": "integer"
            },
            "e": {
               "id": "http://www.icosole.eu/metadata/about/e",
               "type": "integer"
            }
       },
       "required": [
            "stream",
            "t"
       ]
    },
    "annotation": {
      "id": "http://www.icosole.eu/metadata/annot/annotation",
      "type": "string"
    },
    "rating": {
      "id": "http://www.icosole.eu/metadata/annot/rating",
      "type": "number"
    },
    "source": {
      "id": "http://www.icosole.eu/metadata/annot/source",
      "type": "string"
    }
  }
}
```

Social media user data

Social media user data is defined to be always in a static metadata package (at least for the time being).

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "http://www.icosole.eu/metadata/user",
  "type": "object",
  "properties": {
    "id": {
      "id": "http://www.icosole.eu/metadata/user/id",
      "type": "string"
    },
    "socialmediaaccounts": {
      "id": "http://www.icosole.eu/metadata/user/socialmediaaccounts",
      "type": "array",
      "items": {
        "id": "http://www.icosole.eu/metadata/user/socialmediaccount",
        "type": "object",
        "properties": {
          "userid": {
            "id": "http://www.icosole.eu/metadata/user/socialmediaccount/userid",
            "type": "string"
          },
          "token": {
            "id": "http://www.icosole.eu/metadata/user/socialmediaccount/token",
            "type": "string"
```



```
          }
        }
      }
    },
    "email": {
      "id": "http://www.icosole.eu/metadata/user/email",
      "type": "string"
    },
    "name": {
      "id": "http://www.icosole.eu/metadata/user/name",
      "type": "string"
    },
    "friends": {
      "id": "http://www.icosole.eu/metadata/user/friends",
      "type": "array",
      "items": {
        "id": "http://www.icosole.eu/metadata/user/friend",
        "type": "string"
      }
    },
    "required": [
        "id"
    ]
  }
}
```

10.10.3 Examples

A frame annotated as key frame by the system.

```
{
  "tags": [
    {
      "link": "http://icosole.eu/tag/keyframe"
    }
  ]
}
```

Some semantic annotations and hash tags.

```
{
  "tags": [
    {
      "link": "http://dbpedia.org/page/AC/DC"
    },
    {
      "link": "http://dbpedia.org/page/Red_Bull_Ring"
    },
    {
      "label": "#acdc2015"
    },
    {
      "label": "#RockOrBustWorldTour"
    },
    {
      "userlabel": "ACDC"
    }
```



```
  ]
}
```

A like on Facebook.

```
{
  "rating": 1.0,
  "source": "http://www.facebook.com/JohnDoe"
}
```

A related tweet.

```
{
  "annotation": "#Angus #ACDC #RockOrBustWorldTour  TV interview on French TV @frenchopen2015
Enjoy !!  https://m.youtube.com/watch?t=12&v=ROBQcLylDH8 …",
  "source": "https://twitter.com/ACDCtwitta"
}
```

Tweet about previous video content

```
{
  "annotation": "#Angus #ACDC #RockOrBustWorldTour  TV interview on French TV @frenchopen2015
Enjoy !!  https://m.youtube.com/watch?t=12&v=ROBQcLylDH8 …",
  "source": "https://twitter.com/ACDCtwitta",
  "about": {
    "stream": "01234-abcd-1243453-def",
    "t": "123040"
  }
}
```

Social media user data

```
{
    "id": "user123",
    "name": "John Doe",
    "socialmediaaccounts": [ {"userid": "johndoe71", "token": "01234545678"} ],
    "friends": [ "user456" ]
}
```

10.11 Schedule

10.11.1 Introduction

This section describes metadata for representing schedules (e.g. playlists). A metadata packet shall contain a single session/performance, and the start/end time shall describe the start/end time of the session. Schedule metadata is considered not to be updateable, i.e. in case of changes the old data must be deleted.

In order to provide further metadata, the tagging framework can be used.

10.11.2 JSON Schema

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "http://www.icosole.eu/metadata/schedule",
  "type": "object",
```



```
  "properties": {
    "title": {
      "id": "http://www.icosole.eu/metadata/schedule/tags",
      "type": "string"
    },
    "tags": {
      "ref": "http://www.icosole.eu/metadata/annot/tags"
    }
  }
}
```

10.11.3 Example

```
{
  "title": "Great performance",
  "tags": [
    {
      "label": "John Doe"
    }
  ]
}
```


11 Glossary

Terms used within the ICoSOLE project, sorted alphabetically:

ADM	Audio Definition Model
BIFS	Binary Format for Scenes (specified in MPEG)
DoW	Description of Work
EBU	European Broadcast Union
HOA	Higher Order Ambisonics
MPEG	Moving Pictures Expert Group
UC	Use Case
UGC	User Generated Content
WPx	Work Package x

Partner Acronyms

BBC	British Broadcasting Corporation, UK
BIT	Bitmovin Softwareentwicklung OG, AT
DTO	Deutsche Thomson OHG (Technicolor), DE
iMinds	iMinds vzw, BE
JRS	JOANNEUM RESEARCH Forschungsgesellschaft mbH, AT
TaW	Tools at Work Hard + Soft Vertriebsges.m.b.H, AT
VRT	De Vlaamse Radio en Televisieomroeporganisatie NV, BE

Acknowledgement: The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 610370.

