Format definitions for integrated content and scene

Deliverable D4.3

ICoSOLE identifier:	ICoSOLE-D4.3-VRT-FormatDefinitionsForIntegratedContentAndScene_v06.doc
Deliverable number:	D4.3
Main author of Deliverable:	Rik Bauwens (VRT), Mike Matton (VRT), Jürgen Schmidt (DTO), Werner Bailer (JRS)
Internal reviewer:	Ingo Randolf (TaW)
Work package / task:	WP4 / T 3,4
Document status:	Final
Confidentiality:	Public

Version	Date	Reason of change
1	2014-09-11	Document created, initial structure
2	2014-10-07	Slightly modified structure, added placeholder for serialization
3	2014-10-21	Added introduction, executive summary, scope, related, calendar data model, serialization, conclusion, glossary
4	2014-10-31	Complete draft for internal review
5	2014-11-14	Final version

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

This document does not represent the opinion of the European Community, and the European Community is not responsible for any use that might be made of its content.

Neither the ICoSOLE consortium as a whole, nor a certain party of the ICoSOLE consortium warrant that the information contained in this document is capable of use, nor that use of the information is free from risk, and does not accept any liability for loss or damage suffered by any person using this information.

Table of Contents

1 Introduction 1

List of Figures 2

List of Tables 3

2 Executive Summary 4

3 Introduction 5
3.1 Purpose of this Document 5
3.2 Scope of this Document 5
3.3 Status of this Document 5
3.4 Related Documents 5

4 Semantic Scene Model 6
4.1 Transformation of capture scene to semantic scene 6
4.2 Scene data model 6
4.3 Serialization of semantic scene data model 10

5 Format specifications 13
5.1 Audio format specifications 13
5.2 Video format specifications 13
5.3 Metadata format specifications 14

6 Conclusions 15

7 References 16

8 Glossary 17

9 Appendix A – Example comparison of capture and semantic scene representations 18
9.1 Capture Scene 18
9.2 Semantic Scene 20

1 Introduction

In the ICoSOLE project, data and metadata originating from various devices and other sources need to be stored and accessible. For good interoperability, it is crucial that standard formats are defined in a meaningful way. This document describes audio and video format specifications, as well as a semantic scene model that organizes this information in a logical way, and how the capture scene representation (D3.1.1) can be transformed into the semantic scene representation.

Furthermore, (parts of) this semantic scene model will be used to exchange information between different applications. To achieve this, an example in JSON format, which specifies the serialization of this semantic model, has been created (see Annex).

List of Figures

Figure 1: Overview of the model. Coloured entities are defined in the system architecture, the gray entities in the bottom row are specialisations added by the scene data model. 7
Figure 2: Specialised entities for capture devices 8
Figure 3: Specialised entities for objects. 8
Figure 4: Specialised entities for streams. 9
Figure 5: Specialised entities for processing devices. 9
Figure 6: Calendar datamodel. 10
Figure 7: Scene used for example. 18
Figure 8: Capture scene representation examples (created using json-discoverer). 19

List of Tables

Table 1: Parameter description for serialization via REST 9

2 Executive Summary

This document defines some standards regarding data and metadata formats that will be used throughout the production systems of the ICoSOLE project.

All captured content is collected and represented in a capture scene model (D3.1.1). This capture scene model provides input for the rest of the production process (e.g. used devices).

In this document, the basis for a semantic scene model is described. It represents the output of the production process (e.g. produced A/V streams). Both the capture and semantic scene models are based on the same data model and have been aligned with the system architecture (D2.3).

Next, the transformation of this capture scene model to the semantic scene model is outlined. It provides guidelines on how to represent the necessary information from the capture scene model in its semantic counterpart for further use in production.

This document also provides a detailed schematic overview of the semantic scene model, as well as a JSON format for serializing this model. This serialization is required to transfer (parts of) this model in several (client) applications.

Lastly, various audio and video formats which will be used in the project are determined, both for production and UGC content.

In the appendix, a comparison of the capture scene model and semantic scene model is described. The data and metadata collected during the Salford test shoot is hereby used as an example.

3 Introduction

3.1 Purpose of this Document

Format definitions for integrated content and scene: Specify formats specifications needed for professional content and UGC. This covers professional audio content in an object based description and sound field based descriptions, professional video content, audio-UGC and video-UGC."

3.2 Scope of this Document

This document covers a description of the semantic scene model, the serialization format of this model and audio/video format specifications. The transformation from the capture scene model into the semantic scene is also described.

3.3 Status of this Document

This is the final version before review.

3.4 Related Documents

Before reading this document it is recommended to be familiar with the following documents:

- D2.3 System architecture and interface definitions
- D3.1.1 Initial scene representation

4 Semantic Scene Model

4.1 Transformation of capture scene to semantic scene

The capture scene representation provides the input to the production process. It represents the devices used to capture an event, including cameras and microphones. The semantic scene representation is the output of the production process, representing the produced event content ready for distribution to the audience, with the meta-data required to create rich immersive and interactive user experiences.

The capture scene representation and the semantic scene model are based on the same data model. The processing chain from capture to distribution performs a gradual transformation of the model, including

- Moving from capture devices and their parameters to streams and metadata streams
- The model starts being structured by devices and turns into being structured by streams
- Objects are being added to the model, enhancing/replacing streams
- Extracting descriptive metadata and representing it explicitly
- Providing control meta-data to define spatial rendering of streams
- Providing control meta-data to define interactivity in rendering of content
- Filtering content
- Putting streams in relation and adding transitions
- Distribution nodes are gradually added and connected to the streams

It is expected that different tools along the processing chain have their own instantiation of the model, representing their view on the entire scene. These instances of the model are connected by the video, audio and metadata streams used as inputs and outputs.

4.2 Scene data model

The scene data model is based on the entities of the system architecture (see D2.3). The most relevant entities for the scene data model are specialisations of devices and streams. In particular, two classes of devices, CaptureDevice and PlayoutDevice are defined. Another general entity added is Object, representing an object in the scene being captured by at least one of the devices (i.e., being visible or audible in at least one of the streams). A Pack is a set of streams that are logically related (cf. the definition of AudioPack in EBU Tech 3364). The relation of the entities to the entities of the system architecture is shown in Figure 1. The coloured entities in the top rows are defined as part of the system architecture, the gray entities in the bottom row are added by scene data model.

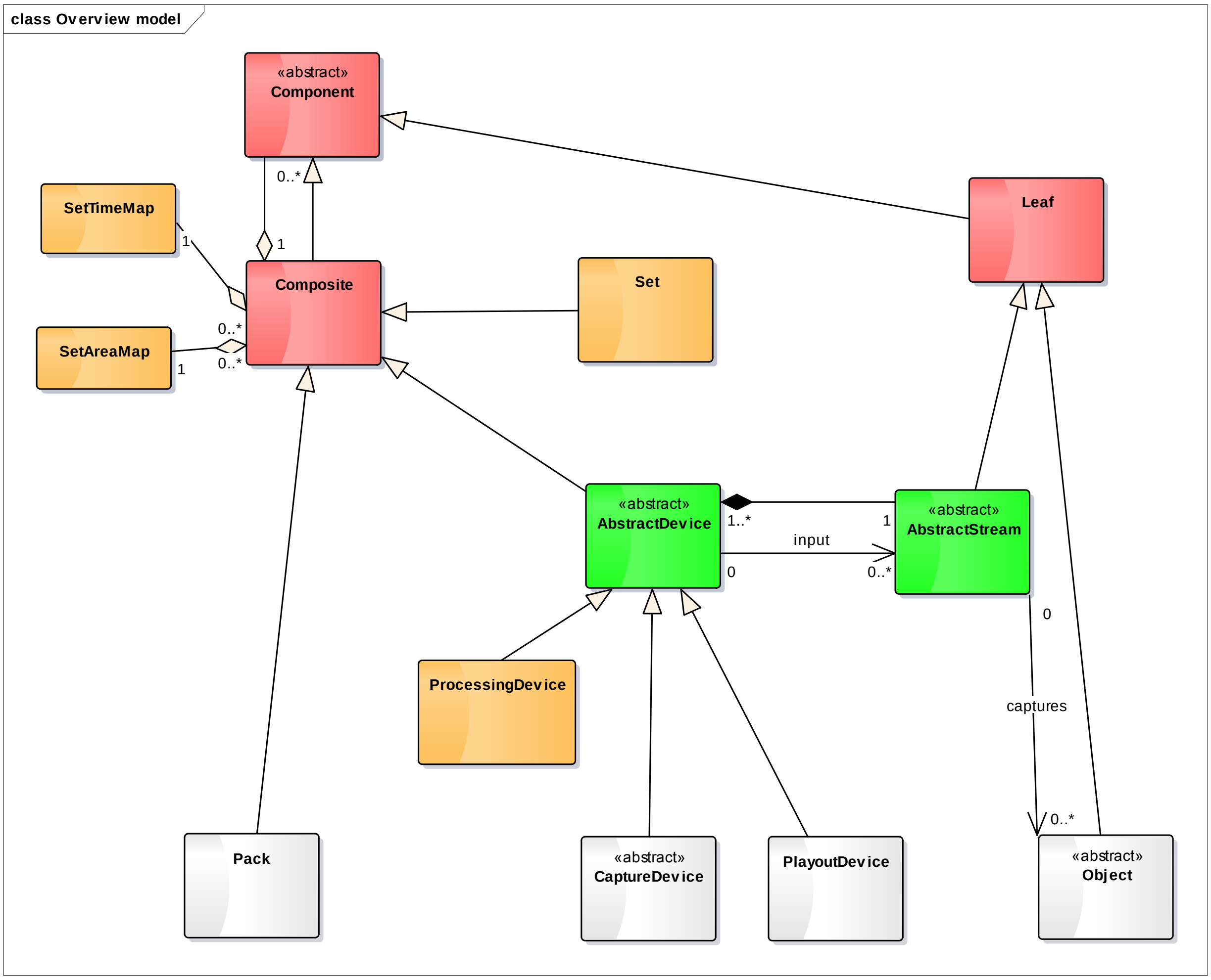

Figure 1: Overview of the model. Coloured entities are defined in the system architecture, the gray entities in the bottom row are specialisations added by the scene data model.

Figure 2 shows the specialised classes related to content capture. The properties of specialised capture devices are defined in D3.1.1. Due to the fact that AbstractDevice is a Composite, capture devices can be treated as sets of sensors (e.g., the set of machine vision cameras of an omnidirectional camera) or as a single sensor (e.g., an omnidirectional camera providing a stitched video stream. Devices can implement the Movable interface, which identifies them as movable devices and provides functions for parameter updates.

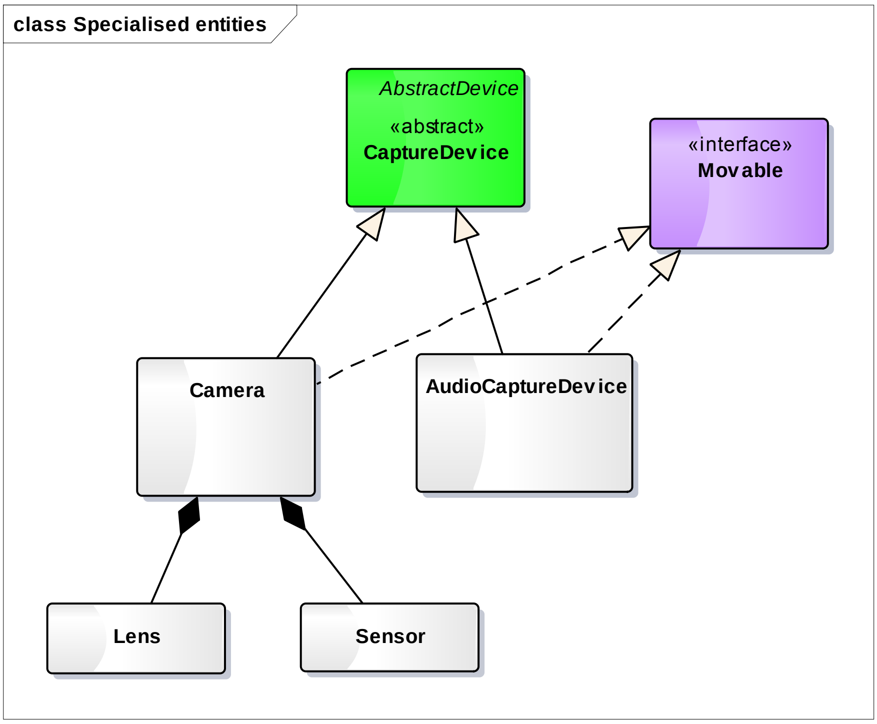

Figure 2: Specialised entities for capture devices

A shown in Figure 3, objects can be specialised in two dimensions: by modality (video, audio) or by being a person/group of persons (Agent) or a thing (Item). Agents may be compositions of other Agents (e.g., sports teams, music bands). As these specialisations are not mutually exclusive, Audio- and VisualObjects are modelled as interfaces of which one or both can be implemented.

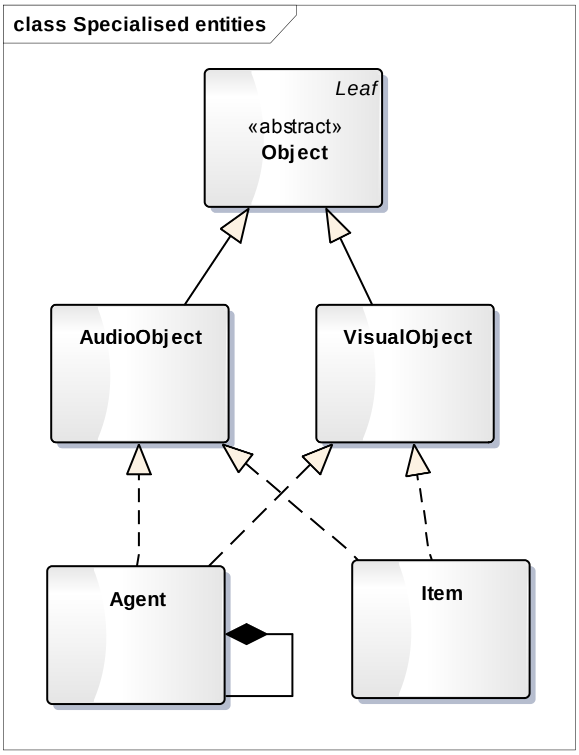

Figure 3: Specialised entities for objects.

Figure 4 shows specialisations of streams. Elementary streams can be further distinguished by the modality they carry. Metadata streams transport static or time-based metadata.

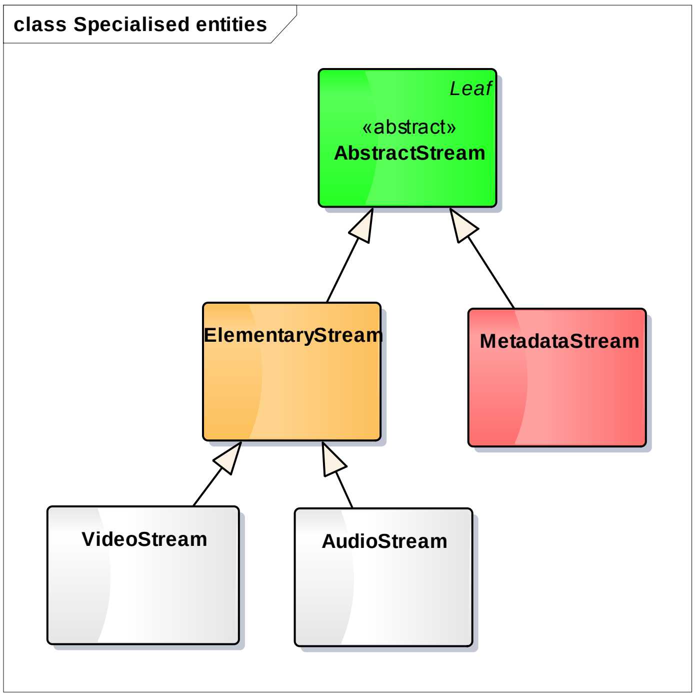

Figure 4: Specialised entities for streams.

Figure 5 shows some specialization of processing devices, however, this set of specialisation is not assumed to be complete, and further types will be defined when details of processing chains are developed.

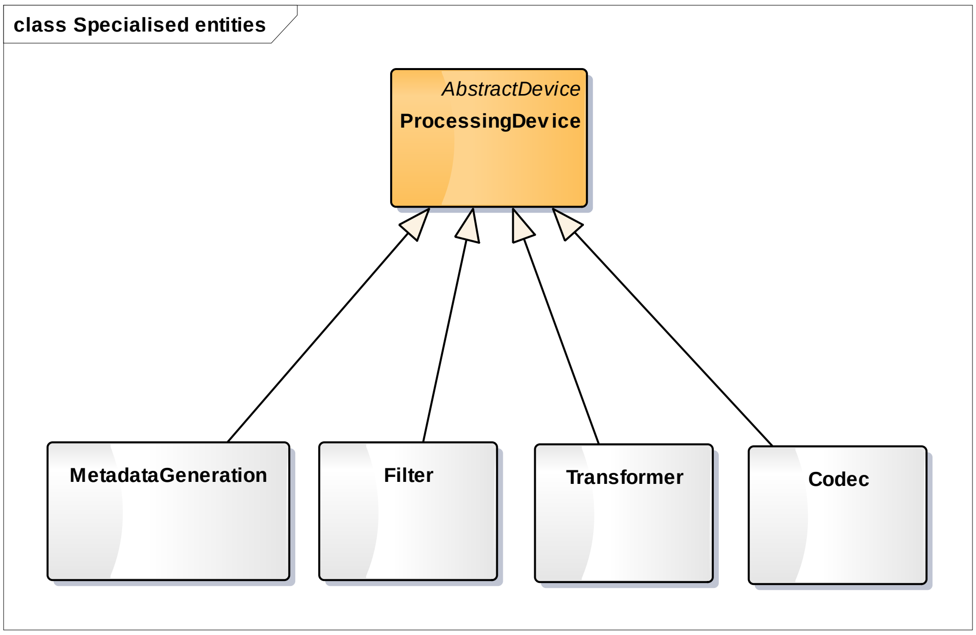

Figure 5: Specialised entities for processing devices.

Note that according to the system architecture (D2.3) ProcessingDevices use elementary streams as their inputs and outputs. The use of multiplexed streams should at all means be avoided, in order not to put the processing time and complexity burden on ProcessingDevices. CaptureDevices may implement a Demutiplexer interface if needed, and provide the extracted elementary streams. Similarly, PlayoutDevices may implement a Multiplexer interface, and perform the multiplexing if they need to provide a multiplexed output. If necessary, a set of demultiplexed streams from the same source can be represented as Pack.

Figure 6 shows the serialization of the semantic scene data model. A Set correlates with a Set from the system architecture. Therefore, it inherits an UUID, title, description and objects. A description can contain any string (e.g. XML data, JSON data,…), where the description type defines which kind of data can be found in this description. A set can have zero or more contributors, which are of the type Agent. In the performance, a link to the SetTimeMap and SetAreaMap (defined in the system architecture) is provided as well.

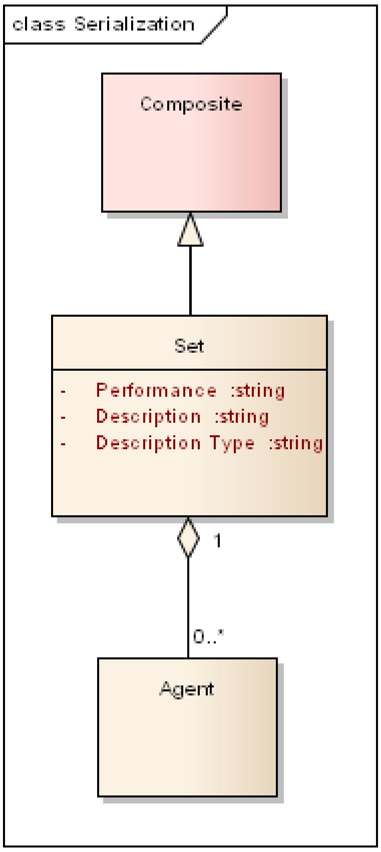

Figure 6: Calendar datamodel.

4.3 Serialization of semantic scene data model

To be able to use (parts of) the semantic model in (client) applications, the mechanism to manage Sets and Devices - as described in the system architecture definition document “Deliverable D2.3 – System Architecture and Interface Definitions” - will be used. This allows the creation, connection, and manipulation of Sets and Devices, as described in chapter 6.9 in D2.3. Due to the generic approach of the system architecture, no individual implementations, but general access methods to the Sets and Devices are defined. The powerful mechanism of schema-based XML validation is implemented in the system architecture and will be used to increase the system reliability as well as the implementation effort.

The system architecture management will handle the following attributes for sets:

- ObjectId (UUID)
- Parent
- SetName
- Location coordinates
- StartTime, EndTime
- Generic Parameters (to be added to the SA document)

And the following attributes for devices:

- HostName
- ObjectId (UUID)
- Parent

- DeviceName
- Streams
- StartTime, EndTime
- Generic Parameters

Besides common properties, all Devices and Sets have individual parameters - “generic parameters” in the above table - that are subject to processing in the application layer. Therefore, these parameters have to be managed by the SetComposer component to be transparently exchanged between the application and the Devices and Sets. Its generic handling is planned in the SetComposer, including XML validation of these parameters (to the date of the preparation of this document). The generic approach allows the arbitrary use of parameters. Therefore, the definition of these individual parameters is subject of this chapter and described below.

Class	Properties	Type
Set	System architecture inheritance	Composite
	Performance	string
	Contributors	Agent[]
	(Description, Description Type)	(String,String)[]
CaptureDevice	System architecture inheritance	AbstractDevice
	captures	uuid[]
Camera	System architecture inheritance	Device
	lens	uuid
	sensor	uuid
AudioCaptureDevice	System architecture inheritance	CaptureDevice
Object	System architecture inheritance	Leaf
	(Description, Description Type)	(String,String)[]
Agent	System architecture inheritance	Object
	cosistsOf	Agent[]
Item	System architecture inheritance	Object
Pack	System architecture inheritance	Composite

	(Description, Description Type)	(String,String)[]

Table 1: Definition of object-properties

5 Format specifications

5.1 Audio format specifications

5.1.1 Import format specification

Uncompressed

All digital audio should be linear PCM.

1 or 2 channel audio CoDec

48kHz mono HE-AAC 48..96 kBit/s

48kHz stereo HE-AAC 96..192 kBit/s

48kHz mono AAC-LC >48..96 kBit/s

48kHz stereo AAC-LC >96..192 kBit/s

Transport

ADTS (Audio Data Transport Stream)
LOAS (Low Overhead Audio Stream)
RTSP multiplex

5.1.2 Internal format specification

Uncompressed

All digital audio should be linear PCM sampled at 48 kHz with single precision floating point.

For measuring loudness levels it is recommended to use EBU R128 with -23 LUFS as the default loudness setting. This is likely to be more relevant for distribution and emission stages rather than capture and production.

5.1.3 Export format specification

To be defined by use-case scenario, except for the following format.

Special import/ export format specification

3D Audio

Spatial Coding for Higher Order Ambisonics MHAS (MPEG-H Audio Streaming) (Currently under development at MPEG-ISO/IEC)

5.2 Video format specifications

5.2.1 Import format specification:

Ultra-HD

2160p@50, MPEG-4 AVC High 4:2:2 / High 4:4:4 (Intra) @ L5+

HD

1080p@25, MPEG-2 422P@HL (XDCAM HD 422)

1080p@50, MPEG-4 AVC, High 4:2:2 / High 4:4:4 (Intra) @ L5+

UGC

1080p@~30, MPEG-4 AVC Baseline/High @ L4+

2160p@~30, MPEG-4 AVC High @ L4+

5.2.2 Internal format specification:

uncompressed RGB 16bit @25-50

2160p@50, MPEG-4 AVC High 4:2:2 / High 4:4:4 (Intra)

1080p@25-50, MPEG-4 AVC, Baseline

5.2.3 Export format specification:

Ultra HD

2160@50, MPEG-H

HD

1080p@50 MPEG-4 AVC, MP

720p@50 MPEG-4 AVC, MP

SD

576i@25, MPEG-2 MP

5.3 Metadata format specifications

5.3.1 Generic metadata streams

- Metadata that is potentially dynamic is treated as dynamic metadata (e.g., geo-location of a mobile device, which may be static for a long time)
- Spatial metadata (e.g., position of stages, performance lineup of live event) is represented using the same type of streams
- There is a specific stream type for every group of metadata properties defined in D3.1.1 (Section 5).
- Update rates for all metadata streams are dynamic, with a minimum to be defined per stream (in the order of seconds to minutes)

5.3.2 Audio control metadata streams

Audio streams can have corresponding meta-data streams describing rendering control data. These meta-data streams will take the form of an extensible JSON serialisation of the Audio Definition Model [1] [EBU3364]. This meta-data stream will provide the control data for rendering audio streams, including channel-based, object-based, and HOA content. The meta-data stream will specify how tracks of the corresponding audio stream should be interpreted, including how they should be rendered spatially, identifying relationships between components, and also may specify potential user interactions during rendering.

[1] The serialisation of the audio description model is work in progress and will be available in a later stage of the ICoSOLE project.

6 Conclusions

Because a lot of different capture devices, as well as different (client) applications, are involved in the ICoSOLE project, a well-described format for data and metadata is required. This document covers the semantic scene model (postproduction), the transformation from the capture scene model (preproduction) into the semantic scene model and the format specifications. Therefore, it will be the basis for several applications that use data or metadata from the ICoSOLE system.

7 References

[D2.3] ICoSOLE D2.3, System Architecture and Interface Definitions, DOI: 10.7800/304ICOSOLED23, http://icosole.eu/wp-content/uploads/2014/11/ICoSOLE-D2.3-DTO-System-Architecture-and-Interface-Definitions_v1.0.pdf

[D3.1.1] ICoSOLE D3.1.1, Initial Format Agnostic Scene Representation.

[EBU3364] Audio Definition Model, EBU Tech 3364, v1.0, Jan. 2014.

8 Glossary

Terms used within the ICoSOLE project, sorted alphabetically.

REST	Representational State Transfer
JSON	JavaScript Object Notation
UGC	User Generated Content

Partner Acronyms

BBC	British Broadcasting Corporation, UK
BIT	Bitmovin GmbH, AT
DTO	Technicolor, DE
iMinds	iMinds Vzw, BE
JRS	JOANNEUM RESEARCH Forschungsgesellschaft mbH, AT
TaW	Tools at Work Hard+Soft Vertriebsgmbh, AT
VRT	De Vlaamse Radio en Televisieomroeporganisatie NV, BE

Acknowlegdement: 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.

9 Appendix A – Example comparison of capture and semantic scene representations

This appendix presents an illustrative example of the differences between a scene representation of the capture process and a scene presentation of the produced content (the semantic scene), as described in Section 4.1. The example is based on recordings made at the “controlled environment test shoot” made by the ICoSOLE project in Salford on 25th March 2014.

The example specifically uses take 15 during session 4 of the shoot, where two small ensembles of musicians were playing different pieces simultaneously to simulate two stages of an event. An ensemble of flute, violin and cello played Mozart’s Divertimento No.2 on the left-hand side of the room and an ensemble of two violins, viola and cello played Bach’s Polomous on the right-hand side of the room, see Figure 7. There were professional audio and video devices capturing each ensemble, as well as some capturing the whole scene. There were also static and moving UGC devices capturing the ensembles with the moving devices transitioning between the two “stages”.

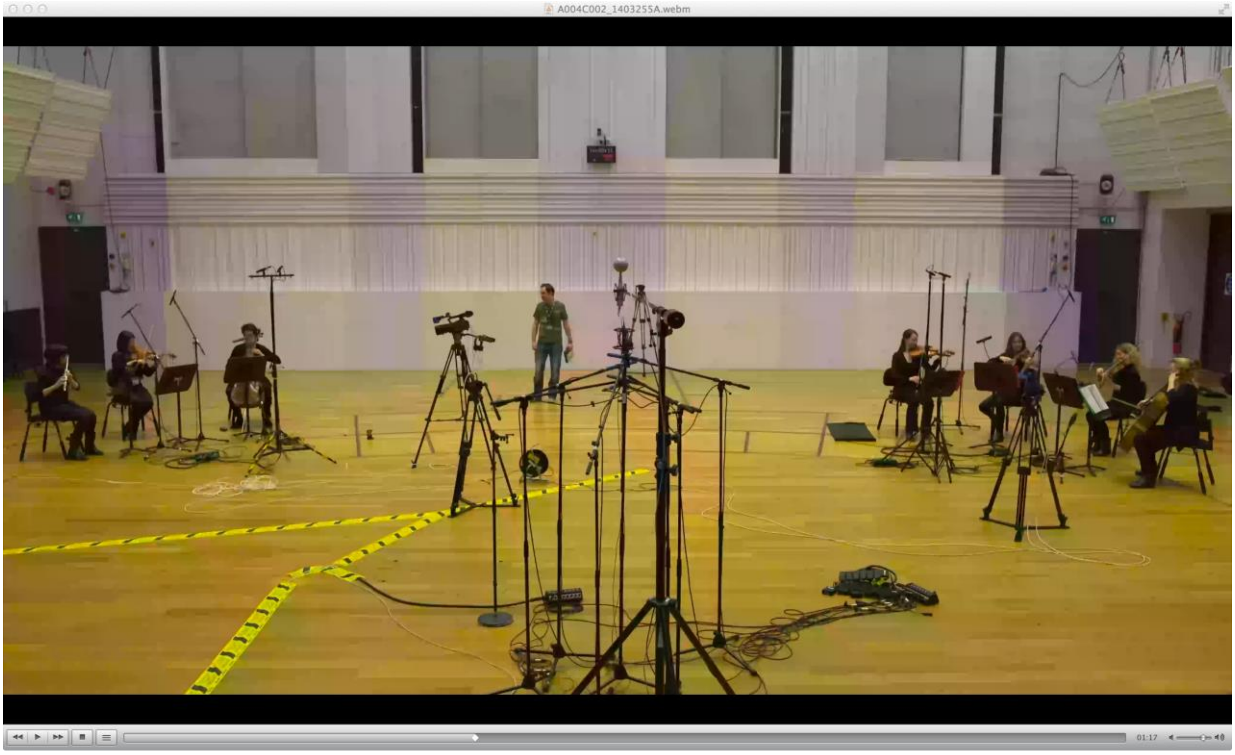

Figure 7: Scene used for example.

9.1 Capture Scene

The capture scene consists of the devices used to capture the event (except the moving UGC devices which are currently not included). Based on the definitions in D3.1.1 Initial Format Agnostic Scene Representation (Section 5) a JSON file has been created to store the capture scene data for this example. There are three sets (the global scene set and the two stages within that set). Each set contains video cameras and audio capture devices and in some cases Packs (see Table 1) which group streams together. Separate UGC Audio Entity and Spatial Audio Entity objects are also used. Figure 8 gives an insight into the data within this JSON document. It was created automatically using an online JSON visualisation tool, but the representation is incomplete due to class names being interpreted as object names.

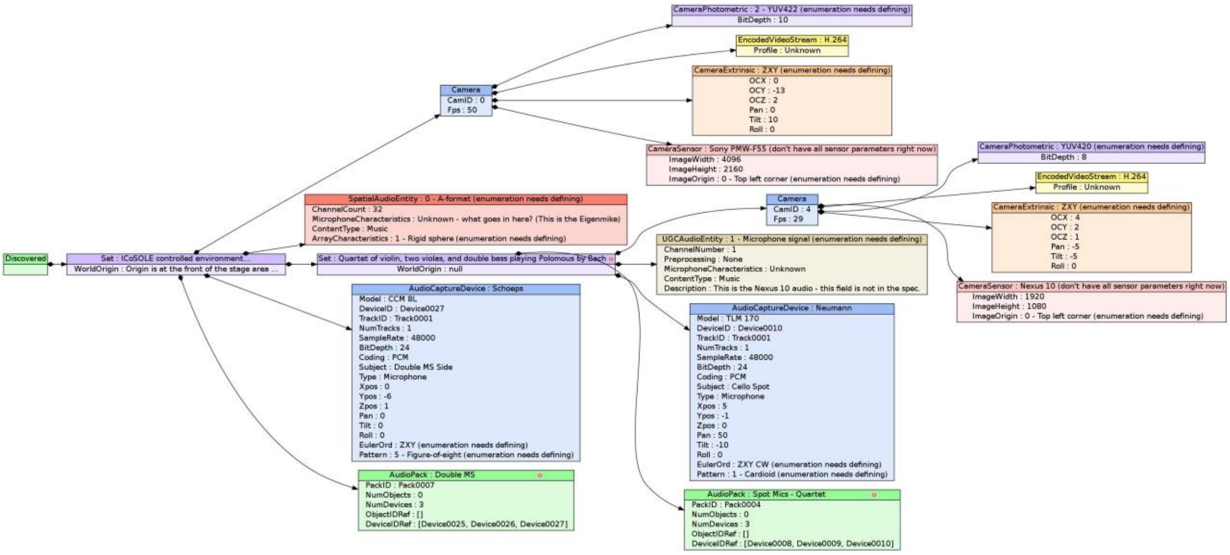
CameraPhotometric : 2 - YUV422 (enumeration needs defining)
BitDepth : 10
EncodedVideoStream : H.264
Profile : Unknown
CameraExtrinsic : ZXY (enumeration needs defining)
OCX : 0
OCY : -13
OCZ : 2
Pan : 0
Tilt : 10
Roll : 0
Camera
CamID : 0
Fps : 50
CameraSensor : Sony PMW-F55 (don't have all sensor parameters right now)
ImageWidth : 4096
ImageHeight : 2160
ImageOrigin : 0 - Top left corner (enumeration needs defining)
CameraPhotometric : YUV420 (enumeration needs defining)
BitDepth : 8
EncodedVideoStream : H.264
Profile : Unknown
SpatialAudioEntity : 0 - A-format (enumeration needs defining)
ChannelCount : 32
MicrophoneCharacteristics : Unknown - what goes in here? (This is the Eigenmike)
ContentType : Music
ArrayCharacteristics : 1 - Rigid sphere (enumeration needs defining)
Camera
CamID : 4
Fps : 29
CameraExtrinsic : ZXY (enumeration needs defining)
OCX : 4
OCY : 2
OCZ : 1
Pan : -5
Tilt : -5
Roll : 0
Discovered
Set : ICoSOLE controlled environment...
WorldOrigin : Origin is at the front of the stage area ...
Set : Quartet of violin, two violas, and double bass playing Polomous by Bach
WorldOrigin : null
UGCAudioEntity : 1 - Microphone signal (enumeration needs defining)
ChannelNumber : 1
Preprocessing : None
MicrophoneCharacteristics : Unknown
ContentType : Music
Description : This is the Nexus 10 audio - this field is not in the spec.
CameraSensor : Nexus 10 (don't have all sensor parameters right now)
ImageWidth : 1920
ImageHeight : 1080
ImageOrigin : 0 - Top left corner (enumeration needs defining)
AudioCaptureDevice : Schoeps
Model : CCM 8L
DeviceID : Device0027
TrackID : Track0001
NumTracks : 1
SampleRate : 48000
BitDepth : 24
Coding : PCM
Subject : Double MS Side
Type : Microphone
Xpos : 0
Ypos : -6
Zpos : 1
Pan : 0
Tilt : 0
Roll : 0
EulerOrd : ZXY (enumeration needs defining)
Pattern : 5 - Figure-of-eight (enumeration needs defining)
AudioCaptureDevice : Neumann
Model : TLM 170
DeviceID : Device0010
TrackID : Track0001
NumTracks : 1
SampleRate : 48000
BitDepth : 24
Coding : PCM
Subject : Cello Spot
Type : Microphone
Xpos : 5
Ypos : -1
Zpos : 0
Pan : 50
Tilt : -10
Roll : 0
EulerOrd : ZXY CW (enumeration needs defining)
Pattern : 1 - Cardioid (enumeration needs defining)
AudioPack : Double MS
PackID : Pack0007
NumObjects : 0
NumDevices : 3
ObjectIDRef : []
DeviceIDRef : [Device0025, Device0026, Device0027]
AudioPack : Spot Mics - Quartet
PackID : Pack0004
NumObjects : 0
NumDevices : 3
ObjectIDRef : []
DeviceIDRef : [Device0008, Device0009, Device0010]

Figure 8: Capture scene representation examples (created using json-discoverer[2]).

The aim of this process was to provide an example of use of the initial format agnostic scene representation, both to test it out and to compare it to an example semantic scene.

9.1.1 Devices Used

- Consumer devices
  - Nexus 10
  - iPhone 4S (moves between stages)
  - Nexus 7
  - Samsung Note (moves between stages)
  - Go Pro
- Professional video
  - Sony PMW-F55 4k
  - CanonXF305_1
  - CanonXF305_2
- Professional audio
  - Eigenmic
  - Hamasaki Square-Square
  - Super CMIT array
  - Double MS
  - Trio Stereo Pair
  - Trio - Flute (spot mic)
  - Trio - Violin (spot mic)
  - Trio - Cello (spot mic)
  - Quartet Stereo Pair
  - Quartet - Violin (spot mic)
  - Quartet - Violas (spot mic)
  - Quartet - Double Bass (spot mic)

[2] JSON Discoverer tool http://atlanmod.github.io/json-discoverer/

9.2 Semantic Scene

An associated (and synchronized) object-based audio stream would be delivered that can be adapted in the client application based on user interactions. The features presented here are not prescriptive or exhaustive, but will hopefully provide some ideas. At this stage the scene is not written in a JSON format but in note form.

Class names are represented in bold and parameters in italics. Explanatory text is also included.

- **Set**
    - *Title: ICoSOLE two-stage classical music festival*
    - This is the top-level set with other sets (stages) nested within it.
    - **StreamGroup**
        - *Title: Overview*
        - *Description: Venue Explorer*[3] *interactive navigation of the event*
        - This would be an interactive overview stream that allows pan and zoom around the scene.
        - **VideoStream**
            - The video in this case comes from the 4k camera but could equally be from different panoramic or omni-directional sources. Adaptive streaming with DASH could be used for delivering the appropriate view/tile of video to the user depending on their interactions.
        - **AudioStream**
            - An associated (and synchronised) object-based audio stream would be delivered that can be adapted in the client based on the user interactions. Three object groups would be delivered (main, focus, and diffuse) allowing adaptive mixing based on the video viewport to give an impression of corresponding changes to the audio perspective. The audio stream would contain meta-data describing the intended spatial location of the objects within the scene, as well as the range of allowed gain adjustments for each object.
            - **AudioObjects**:
                - Main (Double MS)
                - Focus (SuperCMIT shotgun array)
                - Diffuse (Hamasaki Sq-Sq)
        - **MetaDataStream**
            - A meta-data stream presenting schedule information describing the artists and music schedule with corresponding locations within the scene. This would allow graphical overlays within the scene. This could also include links to other content or services (such as nested sets in the scene or for example BBC Playlister to bookmark a music item).

    - **Set**
        - *Title: Stage A – BBC Philharmonic Trio*
        - **StreamGroup**
            - *Title: BBC Philharmonic Trio*
            - *Description: Trio of flute, violin, and cello playing Divertimento Number Two by Mozart*
            - **VideoStream**
                - This is the broadcaster's edited/vision-mixed HD stream of that set (uses XF305 and semi-pro GoPro footage).

[3] For description of Venue Explorer see D6.1 Initial Demonstrators

- **VideoStream**
  - Alternate UGC video (including iPhone, Samsung Note, and Nexus 7)
- **AudioStream**
  - Stereo Audio – with objects and meta-data to allow instrument levels to be adjusted to preference. With other source material this would more likely allow adjustment of music/crowd balance.
  - **AudioObject**s:
    - Individual instruments (with separate control of each)
    - Room sound

- **Set**
  - *Title: Stage B – BBC Philharmonic Quartet*
  - **StreamGroup**
    - *Title: BBC Philharmonic Trio*
    - *Description: Quartet of violin, two violas, and double bass playing Polomous by Bach*
    - **VideoStream**
      - This is the broadcaster's edited/vision-mixed HD stream of that set (uses XF305 and semi-pro GoPro footage).
    - **VideoStream**
      - Alternate UGC video (including iPhone, Samsung Note, and Nexus 7)
    - **AudioStream**
      - Stereo Audio – with objects and meta-data to allow instrument levels to be adjusted to preference. With other source material this would more likely allow adjustment of music/crowd balance.
      - **AudioObject**s:
        - Individual instruments (with separate control of each)
        - Room sound

The interactivity implied by this semantic scene representation includes pan/zoom around an audio-visual stream and manual adjustment of levels within a sound mix. These are just two examples of interactive/personalised experience that might be offered. Other examples could be to swap an object/stream for an alternative e.g. to select alternate commentary or language, to crop video according to screen size, or to adjust the spatial position of certain objects. It may be that relationships/dependencies between objects/streams within the scene also need representation.

The meta-data used to create these interactive experiences would be included with the scene representation and/or in the streams themselves (e.g. meta-data within an object-based audio stream). The client-side applications would require prior knowledge of the meta-data formats in order to enable these experiences.