**ISO/IEC 23090-28:202n(X)**

ISO/IEC JTC 1/SC 29/WG 7

Date: 2023-10-20

**Information Technology** 一 Coded representation of immersive media 一 Part 28: Interchangeable scene-based media representations

CD stage

**Warning for WDs and CDs**

This document is not an ISO International Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an International Standard.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

© ISO 20XX

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO’s member body in the country of the requester.

ISO copyright office

CP 401 • Ch. de Blandonnet 8

CH-1214 Vernier, Geneva

Phone: +41 22 749 01 11

Email: copyright@iso.org

Website: www.iso.org

Published in Switzerland

**Contents**

[1 Scope 1](#_Toc151014335)

[2 Normative references 1](#_Toc151014336)

[3 Terms and definitions 1](#_Toc151014338)

[1.1 General 1](#_Toc151014339)

[1.2 scene graph 2](#_Toc151014340)

[1.3 Immersive Technology Media Format 2](#_Toc151014341)

[1.4 node 2](#_Toc151014342)

[1.5 node graph 2](#_Toc151014343)

[1.6 node type 2](#_Toc151014344)

[1.7 attribute 2](#_Toc151014345)

[1.8 container 2](#_Toc151014346)

[1.9 serialisation 2](#_Toc151014347)

[1.10 renderer 2](#_Toc151014348)

[scene-based media 3](#_Toc151014349)

[1.11 RGB 3](#_Toc151014350)

[1.12 asset 3](#_Toc151014351)

[1.13 pin 3](#_Toc151014355)

[2 Abbreviated terms 3](#_Toc151014356)

[2.1 General 3](#_Toc151014357)

[2.2 2D 3](#_Toc151014358)

[2.3 3D 3](#_Toc151014359)

[2.4 AOV 3](#_Toc151014360)

[2.5 IDEA 3](#_Toc151014361)

[2.6 ITMF 3](#_Toc151014362)

[2.7 IMS 3](#_Toc151014363)

[2.8 OSL 3](#_Toc151014364)

[2.9 RGB 3](#_Toc151014365)

[2.10 USD 4](#_Toc151014366)

[2.11 glTF 4](#_Toc151014367)

[3 Architectures for scene-based media 4](#_Toc151014472)

[3.1 General 4](#_Toc151014473)

[3.2 Architecture for scene graph representations 4](#_Toc151014474)

[3.3 Scenes nested within scenes 5](#_Toc151014475)

[3.4 Architecture for annotated scene graph using IMS 5](#_Toc151014476)

[3.5 Architecture for translation between scene graph representations using IMS 6](#_Toc151014477)

[4 Independent Mapping Space 7](#_Toc151014478)

[4.1 General 7](#_Toc151014479)

[4.2 Structure of IMS 7](#_Toc151014480)

[4.3 IMS Systems 7](#_Toc151014481)

[4.3.1 Value nodes system 11](#_Toc151014482)

[4.3.2 Lighting nodes system 12](#_Toc151014483)

[4.3.3 Geometry nodes system 12](#_Toc151014484)

[4.3.4 Environment nodes system 13](#_Toc151014485)

[4.3.5 Camera nodes system 13](#_Toc151014486)

[4.3.6 Materials nodes system 14](#_Toc151014487)

[4.3.7 Object layer nodes system 15](#_Toc151014488)

[4.3.8 Surface (medium) nodes system 15](#_Toc151014489)

[4.3.9 Texture nodes system 15](#_Toc151014490)

[4.3.10 Transform nodes system 19](#_Toc151014491)

[4.3.11 Renderer instruction nodes system 19](#_Toc151014492)

[4.3.12 OpenColorIO nodes system 19](#_Toc151014493)

[4.3.13 Render Arbitrary Output Variables (AOVs) nodes system 20](#_Toc151014494)

[4.3.14 Arbitrary Output Variables (AOVs) compositor nodes system 22](#_Toc151014495)

[4.3.15 Kernel nodes system 23](#_Toc151014496)

[4.3.16 Node graph types system 23](#_Toc151014497)

[4.3.17 Buffer system 24](#_Toc151014498)

[4.3.18 Animation system 24](#_Toc151014499)

[4.3.19 Pins data system 25](#_Toc151014500)

[4.3.20 Attributes system 27](#_Toc151014501)

**Foreword**

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see [www.iso.org/directives](https://www.iso.org/directives-and-policies.html)).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see [www.iso.org/patents](https://www.iso.org/iso-standards-and-patents.html)).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see [www.iso.org/iso/foreword.html](https://www.iso.org/foreword-supplementary-information.html).

ISO/IEC 23090-28 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information Technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

A list of all parts in the ISO/IEC 23090 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at [www.iso.org/members.html](https://www.iso.org/members.html).

**Introduction**

This part of ISO/IEC 23090 applies to use cases where there is a need for interchange of scene-based media for presentation by systems with render-based technologies. One specific application of this part is to immersive display technologies that utilize a real-time renderer of 3D media to create their visual presentations, as opposed to a video decoder of 2D raster-based media. Another application of this part is for the distribution of media assets for use across a large variety of applications comprising the Metaverse.

Display technologies capable of creating fully formed holograms, e.g. 3D aerial images comprised of waveforms emitted from a display panel surface, require input source media that are sufficiently dense in information such that each of the objects in a holographic visual scene is described in terms of its complete and explicit geometry, and surface properties to characterize how the surface responds to the presence of light. Such visual information can furthermore facilitate display technologies to produce visual scenes of objects that cannot be distinguished from the same objects when viewed in the natural world; i.e., the visual information of the media is sufficiently dense such that the display is enabled to produce a photorealistic result.

Commonly used raster media formats such as those specified in ITU-R BT.601, ITU-R BT.709, ITU-R BT.2020, ITU-R BT.2100 were originally developed to efficiently provide information sufficient for 2D displays to produce 2D visual presentations, albeit with varying degrees of spatial resolutions, frame rates, and pixel depth. These raster formats, however, neither provide sufficient information in a practical nor efficient manner, to enable a holographic display to produce holograms. As an alternative to raster media, 3D graphics formats used in digital content creation tools and production workflows offer a solution to provide sufficiently dense information to enable photorealistic results where raster formats do not. The use of 3D graphics technologies hence serves as a fundamental technology in enabling immersive displays to create their optimal visual presentations.

To date, immersive displays provide a media input interface that is capable of ingesting 3D graphics formats for presentation by the display. Such interfaces may be enabled in the display by use of rendering technologies, currently Unreal Engine® by Epic Games, Inc. and Unity® by Unity Technologies, which may be shipped as part of the onboard processing features of the display. However, 3D scene media that are targeted for processing by the renderer must first be translated into a format that is consistent with the internal scene representations of the renderer. To facilitate the translation and distribution of 3D scene media into the renderer interface provided by the display, an *Independent Mapping Space* is specified as a part of ISO/IEC 23090 Coded Representation of Immersive Media.

Another application of this part is to guide the translation of media assets for use across a diverse set of applications comprising the Metaverse, e.g., gaming, social media, retail applications. In the absence of a common format that specifies media for use across the variety of Metaverse applications, the Independent Mapping Space may facilitate the translation from one asset representation to another.

The International Organization for Standardization (ISO) draws attention to the fact that it is claimed that compliance with this document may involve the use of a patent.

ISO takes no position concerning the evidence, validity and scope of this patent right.

The holder of this patent right has assured ISO that he/she is willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with ISO. Information may be obtained from the patent database available at [www.iso.org/patents](http://www.iso.org/patents).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those in the patent database. ISO shall not be held responsible for identifying any or all such patent rights.

**Information Technology** 一 Coded representation of immersive media 一 Part 28: Efficient 3D graphics media representation for render-based systems and applications

# Scope

This part of ISO/IEC 23090 specifies a model for interchange of scene-based media for use in systems and applications that employ renderers for the presentation of immersive media, including audio, visual, tactile and other representations of media types. Clients of such systems and applications are referred to as *render-based* clients. At the core of the model is a vocabulary of terms and definitions that can be used to annotate a scene graph or scene asset with metadata, i.e., to facilitate the translation of the annotated media from one format to another. Such translation from one format to another format is a commonly encountered challenge in the distribution of scene-based media, as a number of formats have emerged to gain popularity in various application areas, e.g., gaming, social media, e-commerce, and content production. The corresponding vocabulary of terms and definitions describes nodes, media formats, media attributes, and render processing instructions of commonly-used scene graph formats and media representations. This collection of terms and definitions is referred to as the *Independent Mapping Space* (IMS). Elements from the Independent Mapping Space may be used to annotate a scene graph or media format so that its translation to another format is more straightforward and consistently performed. Thus, the IMS facilitates the distribution and interoperability of a variety of scene-based media into render applications that are limited in the types of formats that they are designed to support. Such render applications are anticipated to serve an essential role in the emerging Metaverse.

# Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

*ISO/IEC DIS 12113:2021, Information technology — Runtime 3D asset delivery format — Khronos glTF 2.0[[1]](#footnote-1)*

*ITMF Scene Graph Specification 2.0 —* [*https://immersivealliance.org*](https://immersivealliance.org)

*ITMF Data Encoding Specification —* [*https://immersivealliance.org*](https://immersivealliance.org)

*IETF RFC 8259 (December 2017), The JavaScript Object Notation (JSON) Data Interchange Format*

*XML 1.1 W3C Recommendation Second Edition* [*https://www.w3.org/TR/2006/REC-xml11-20060816/*](https://www.w3.org/TR/2006/REC-xml11-20060816/)

# Terms and definitions

## General

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

* ISO Online browsing platform: available at <https://www.iso.org/obp>
* IEC Electropedia: available at <https://www.electropedia.org/>

## scene graph

general data structure commonly used by vector-based graphics editing applications and modern computer games, that arranges the logical and often (but not necessarily) spatial representation of a graphical scene; a collection of nodes and vertices in a graph structure

## Immersive Technology Media Format

a suite of specifications for the Immersive Technology Media Format developed by the Immersive Digital Experiences Alliance

## node

fundamental element of the scene graph comprised of information related to the logical or spatial or temporal representation of visual or audio information

## node graph

A collection of nodes that can be positioned or modified as a single unit and linked to the remainder of the scene graph by linker nodes.

## node type

A general category of node that performs a certain function, for example, a camera node type can be a panoramic lens camera node, an OSL node, or other types of cameras.

## attribute

metadata associated with a node used to describe a particular characteristic or feature of a node either in a canonical or more complex form (e.g. in terms of another node)

## container

serialised format to store and exchange information to represent all natural, all synthetic, or a mixture of synthetic and natural scenes including a scene graph and all of the media resources that are required for rendering of the scene

## serialisation

process of translating data structures or object state into a format that can be stored (for example, in a file or memory buffer) or transmitted (for example, across a network connection link) and reconstructed later (possibly in a different computer environment)

Note 1 to entry: When the resulting series of bits is reread according to the serialisation format, it can be used to create a semantically identical clone of the original object.

## renderer

software-based) application or process, based on a selective mixture of disciplines related to: acoustic physics, light physics, visual perception, audio perception, mathematics, and software development, that, given an input scene graph and asset container, emits a visual and/or audio signal suitable for presentation on a targeted device or conforming to the desired properties as specified by attributes of a render target node in the scene graph

Note 1 to entry: For visual-based media assets, a renderer may emit a visual signal suitable for a targeted display, or for storage as an intermediate asset (e.g. repackaged into another container i.e. used in a series of rendering processes in a graphics pipeline); for audio-based media assets, a renderer may emit an audio signal for presentation in a multi-channel loudspeaker and/or binauralized headphones, or for repackaging into another (output) container.

## scene-based media

audio, visual, haptic, and other primary types of media and media-related information organized logically and spatially by a use of a scene graph

## RGB

referring to an additive colour model in which red, green, and blue are the primary colours; the abbreviation RGB is derived from the first letter of each of the primary colours

## asset

one of potentially multiple objects comprising the media for a scene in scene-based media; each object fully formed and packaged such that it can be rendered separately from other objects in the scene

Note 1 to entry: individual assets may be shared, e.g., reused, across multiple scenes.

## pin

an input parameter to a node, or an output from a node

# Abbreviated terms

## General

For the purposes of this document, the following abbreviations apply.

## 2D

two dimensional

## 3D

three dimensional

## AOV

Arbitrary output variables

## IDEA

Immersive Digital Experiences Alliance

## ITMF

Immersive Technology Media Format

## IMS

Independent mapping space

## OSL

Open Shading Language

## RGB

Red, Green, Blue

## USD

Universal Scene Description

## glTF

Graphics Language Transmission Format

# Architectures for scene-based media

## General

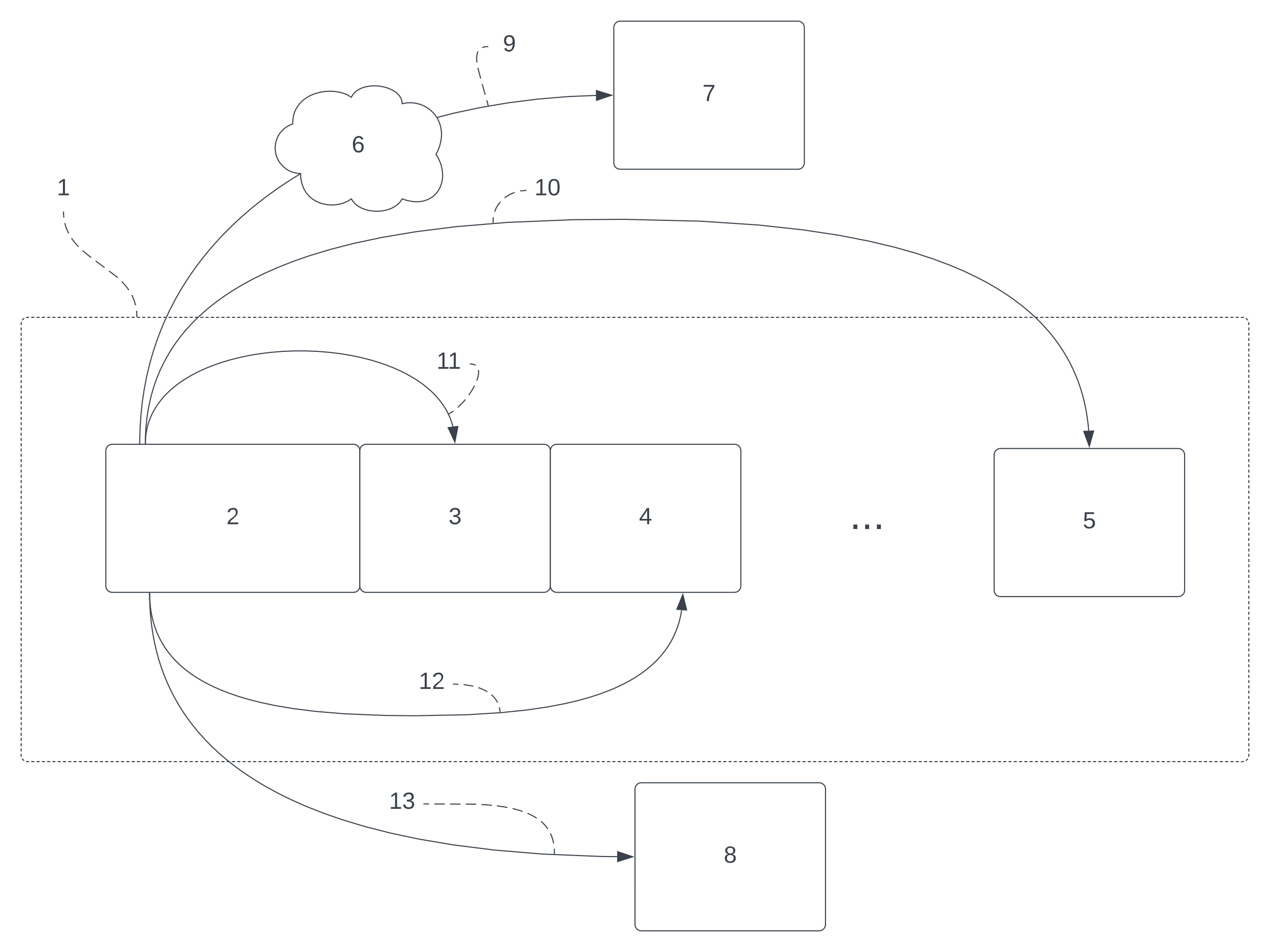
This clause illustrates architectures related to the organization of scene-based media and how such media are annotated by a process that stores IMS metadata. Furthermore, this clause provides a hypothetical architecture for a process that translates one scene-based media format to another.

A complete description of these architectures is provided in ISO/IEC 23090-27.

Note: This specification references only some of the architectures in ISO/IEC 23090-27 for the purposes of providing additional context.

## Architecture for scene graph representations

Figure 1 illustrates the architecture of scene graphs for the representation of scene-based immersive media.



1. **Architecture for scene graph representations of scene-based immersive media**

**Figure 1 key**

|  |  |  |
| --- | --- | --- |
| 1 | binary container for immersive media |  |
| 2 | human-readable or binary scene description |  |
| 3 | media asset #1 |  |
| 4 | media asset #2 |  |
| 5 | media asset #N |  |
| 6 | network |  |
| 7 | media asset #K |  |
| 8 | media asset #M |  |
| 9 | network-based reference from scene description to asset #K |  |
| 10 | local reference from scene description to asset #N stored within binary container |  |
| 11 | local reference from scene description to asset #1 stored within binary container |  |
| 12 | local reference from scene description to asset #2 stored within binary container |  |
| 13 | external storage reference from scene description to asset #M |  |

In this illustration of a scene graph architecture, the description of the scene (1), including spatial, temporal, or logical relationships between objects, is stored in a human-readable format (e.g., XML or JSON). This human-readable portion of the scene graph comprises the information that may be annotated with metadata from the IMS. The scene is further comprised of binary assets that are stored adjacent to the human-readable description in a “container” (13), such as files that are combined, stored, and compressed into a single archive file format (similar to files with a zip archive extension), or assets that are stored externally from the container, e.g., either in a local or remote access network.

## Scenes nested within scenes

An important aspect of scene graph architectures is that there is no explicit restriction within this Specification that prohibits an individual asset from comprising an entirely self-described scene as depicted in Figure 1. That is, referring to the architecture depicted in Figure 1, assets A, B, K, M, and N may themselves be scenes stored in individual containers. Furthermore, these assets may likewise be annotated with IMS metadata.

## Architecture for annotated scene graph using IMS

Figure 2 illustrates an example of a scene graph that is annotated with metadata from the IMS. In the figure, items 1A, 1B, 1C, 1D, and 1E refer to individual parts of the human readable description of the scene. Each part is separately annotated with IMS metadata that corresponds to the description of the scene for that particular part. The figure also illustrates that the scene contains four binary assets labelled: 2A, 2B, 2C, and 2D. The assets themselves may or may not be separately annotated with IMS metadata. Furthermore, the assets themselves may be individual scenes, or may be individual assets that are likewise annotated with IMS metadata.



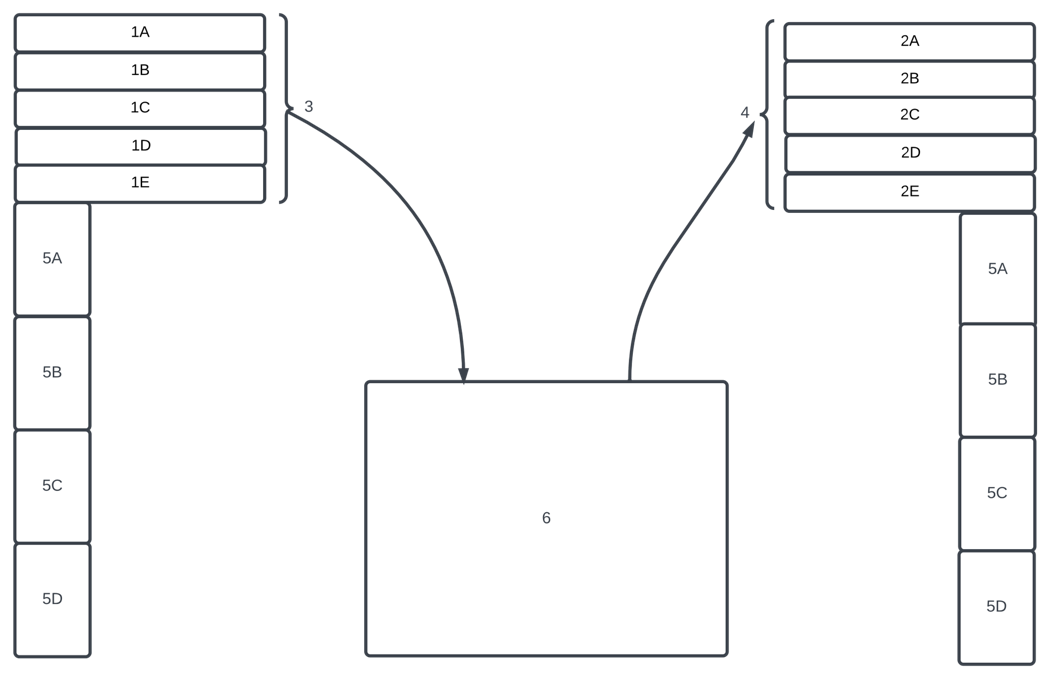
1. **Architecture for scene graph annotated with IMS metadata**

**Figure 2 key**

|  |  |  |
| --- | --- | --- |
| 1A | Portion of human-readable scene description, e.g. XML or JSON, annotated with IMS metadata |  |
| 1B | Portion of human-readable scene description, e.g. XML or JSON, annotated with IMS metadata |  |
| 1C | Portion of human-readable scene description, e.g. XML or JSON, annotated with IMS metadata |  |
| 1D | Portion of human-readable scene description, e.g. XML or JSON, annotated with IMS metadata |  |
| 1E | Portion of human-readable scene description, e.g. XML or JSON, annotated with IMS metadata |  |
| 2A | Binary asset A optionally annotated with IMS metadata |  |
| 2B | Binary asset B optionally annotated with IMS metadata |  |
| 2C | Binary asset C optionally annotated with IMS metadata |  |
| 2D | Binary asset D optionally annotated with IMS metadata |  |

## Architecture for translation between scene graph representations using IMS

A hypothetical architecture for translating one scene graph format to another scene graph format is illustrated in Figure 3.



1. **Architecture for scene graph translation**

**Figure 3 key**

|  |  |  |
| --- | --- | --- |
| 1A | Portion of human-readable Scene Description 1 annotated with IMS metadata |  |
| 1B | Portion of human-readable Scene Description 1 annotated with IMS metadata |  |
| 1C | Portion of human-readable Scene Description 1 annotated with IMS metadata |  |
| 1D | Portion of human-readable Scene Description 1 annotated with IMS metadata |  |
| 1E | Portion of human-readable Scene Description 1 annotated with IMS metadata |  |
| 2A | Portion of human-readable Scene Description 2 translated with IMS metadata from 1A |  |
| 2B | Portion of human-readable Scene Description 2 translated with IMS metadata from 1B |  |
| 2C | Portion of human-readable Scene Description 2 translated with IMS metadata from 1C |  |
| 2D | Portion of human-readable Scene Description 2 translated with IMS metadata from 1D |  |
| 2E | Portion of human-readable Scene Description 1 translated with IMS metadata from 1E |  |
| 3 | Scene Description 1 annotated with IMS metadata |  |
| 4 | Scene Description 2 translated from Scene Description 1 and IMS metadata |  |
| 5A | Binary asset A associated with both Scene Description 1 and Scene Description 2 |  |
| 5B | Binary asset B associated with both Scene Description 1 and Scene Description 2 |  |
| 5C | Binary asset C associated with both Scene Description 1 and Scene Description 2 |  |
| 5D | Binary asset D associated with both Scene Description 1 and Scene Description 2 |  |
| 6 | Translation process |  |

In this figure, Scene Description 2 is derived via a translation process (6) from Scene Description 1 and its corresponding IMS metadata. Scene Description 1 is comprised of five parts labelled 1A, 1B, 1C, 1D, and 1E. Each part is annotated with corresponding IMS metadata. Associated with Scene Description 1 are four binary assets, each individually labelled as: 5A, 5B, 5C, and 5D. Assets 5A, 5B, 5C, and 5D may be optionally annotated. The combined IMS metadata and parts comprising Scene Description 1 are labelled as 3. Component 3 serves as input to a translation process (6) that is guided by the IMS metadata to produce output 4 which is Scene Description 2. Scene Description 2 is also comprised of five parts, although there does not need to be a one-to-one correspondence between the number of input scene description parts to the number of output scene description parts.

# Independent Mapping Space

## General

The Independent Mapping Space (IMS) defines a vocabulary and corresponding set of labels that can be used to describe scene graphs to aid in the translation of a single scene graph format into another single scene graph format, or to aid in the distribution of a scene graph to a particular rendering interface. While descriptions and terms that comprise the IMS are sufficiently dense and robust to describe a scene graph capable of representing photorealistic scenes, the IMS may also be used to describe scene representations that are not designed to represent photorealistic visualizations.

## Structure of IMS

The IMS is comprised of individual systems that collectively describe the organisation of a graph comprised of 3D assets for a scene, and its related processing instructions.

NOTE 1 – A single scene graph may contain multiple sets of rendering instructions, each set relevant to the interface for a particular renderer.

NOTE 2 – In general, immersive media may be comprised of media that is created to stimulate any of the human senses, including the visual and auditory senses of humans. Scenes are likewise organized according to the types of media that stimulate human senses, i.e., visual and auditory scenes. While there are different types of scenes that are used to create immersive media, the current edition of this Specification, defines the components of a scene graph that are designed to describe media for visual scenes.

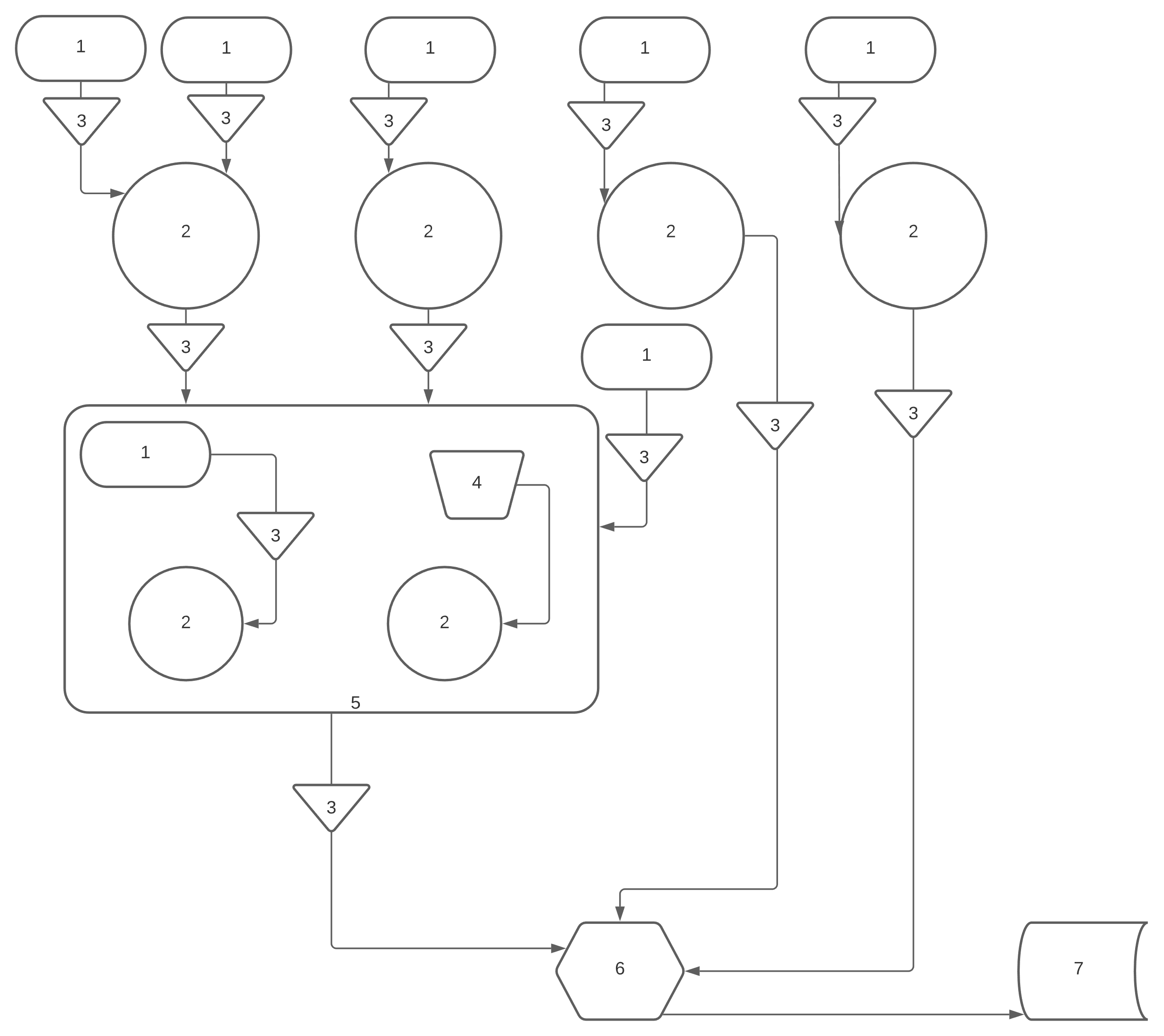
Each of the IMS systems is associated with a description of semantics, i.e., *a processing model*, for how a renderer may interpret the nodes that are associated with each system.

## IMS Systems

The individual systems that comprise the IMS are summarized in Table 1. Associated with each system is a type, description, and hypothetical semantics for how a renderer might interpret a node belonging to each system.

NOTE –These systems are designed to mirror the structure, representation, and encoding of scene-based media as specified by the ITMF Scene Graph Specification and ITMF Data Encoding Specification.

Associated with each system is one or more nodes that belong to the system. Figure 4 provides a canonical representation of how the IMS systems relate to each other.



1. **Canonical relationship of IMS systems**

**Figure 4 key**

|  |  |  |
| --- | --- | --- |
| 1 | Canonical IMS values |  |
| 2 | IMS lighting, cameras, materials, textures, geometry, environments, object layers, surfaces, arbitrary output variables, arbitrary output variable compositors, or transformations |  |
| 3 | input or output pin |  |
| 4 | attribute |  |
| 5 | node graph |  |
| 6 | renderer instructions and parameters |  |
| 7 | rendered output (not a system in the IMS) |  |

An IMS value (item 1) shall represent an input value that describes any one of the other IMS systems (item 2) including: system values, lighting, cameras, materials, textures, geometry, environments, object layers, surfaces, arbitrary output variables, arbitrary output variable compositors, and transformation.

Input or output relationships between any of the system objects shall be established with IMS data pins (item 3). Attributes shall represent other IMS system items that are considered to be immutable (not changeable by rendering interfaces). A node graph (item 5) shall represent a collection of other IMS system items that may also include attributes or render-specific metadata (not in scope of this Specification). Renderer instructions and parameters (item 6) shall denote inputs to rendering interfaces that may each be unique to specific renderers.

**Table 1 Summary of IMS systems**

| **System node type** | **Label** | **Description** | **Hypothetical renderer semantics** |
| --- | --- | --- | --- |
| System value node | sysVal | An input value (e.g., integers, single-precision floats, spatial coordinates, vectors) to an associated node | Values provided by system value nodes are used by renderers for processing of other nodes to which the values are associated. |
| Lighting node | lighting | The lighting (e.g., ambient, point source) for a scene or environment | Lighting determines which objects in a scene are potentially visible to the camera used by the renderer for a scene. |
| Camera node | camera | The camera(s) for the scene | Cameras define the field of view, lens properties, and size of the viewport that is output by a renderer of a scene. |
| Material node | material | Surface properties of individual geometric objects or groups of geometric objects in the scene | Materials provide characteristics (e.g., porousness, bumps, metallic properties) for the surfaces of individual objects that are rendered in the scene. |
| Open Colour IO | ocio | For management of colour to facilitate consistent color transforms and image display across multiple application pipelines. | OpenColorIO nodes specify transforms to be applied by a renderer. |
| Kernel node | kernel | The particular rendering algorithm used during the rendering process. | Kernel nodes define the specific algorithm and methods for use during the rendering process. |
| Texture node | texture | Colour and brightness for a surface corresponding to a geometric objects or group of geometric objects in the scene | Textures define individual colours and brightness for surfaces of individual objects that are rendered in the scene. |
| Geometry node | geometry | Geometric shape of an object or group of objects in a scene or geometry archive | Geometry defines the 3D shape or volume of individual objects that are rendered in the scene. |
| Environment node | environment | The environment (e.g. background and lighting) of the scene | Environments describe properties, of lighting and surrounding backgrounds for objects in the scene. |
| Object layer node | layer | Attributes of an object with respect to its visibility to the camera | Visibility of an object in the scene is controlled by object layer nodes. |
| Surface (medium) node | surface | A description of how light is reflected, refracted, or absorbed by an object | The behaviour of individual rays in ray-tracing algorithms is characterized by formulae provided in surface nodes. |
| Transformation node | transform | A description of how geometric objects are scaled, translated, or rotated. | The parameters are used to transform individual geometric objects within a scene. |
| Render arbitrary output variables (AOV) | aov | A collection of data values saved during a rendering process for a particular object that can subsequently be used for compositing a different version of the same object. | Individual AOVs are saved from the intermediate data values created by a renderer process for a particular target output. |
| Arbitrary output variables compositor | aovCompositor | A collection of nodes that facilitate the creation of composite outputs using arbitrary output variables as inputs. | Compositor nodes create composites from AOVs provided as input. |
| Render instruction node | render | Instructions for how a renderer should serialise and create a representation of the scene | The parameters and instructions guide the creation of rendered output. |
| Graph type | graph | The type of graph | The graph type guides the processing of nodes by the renderer. |
| Data pins | pins.input (input pin)  pins.output (output in) | Parameters used as input to a node, or a single output from a node, i.e., input data pin(s) or output data pin | One or more data pins are inputs to a node or a single data pin is an output from a node. |
| Data attributes | attribute | Non-mutable characteristics, features, or values of or for an object | The renderer does not allow other nodes to alter attribute values. |
| Buffer | buffer | Organization of the binary data comprising the media asset within a buffer. | The organization of the buffer guides the access of the media by the renderer. |
| Animation | animation | Description of how an asset is to be animated. | Animation of an asset by the renderer is guided by the asset’s animation parameters. |
| Scene object | sceneObject | Formats of other scenes used as individual objects for a composite scene. | Denotes use of Alembic, Universal Scene Description, glTF, and Filmbox as individual geometry assets of a larger scene. |

### Value nodes system

This subclause specifies the types of values and labels that comprise the Value Nodes System within the Independent Mapping Space. Further information for how each value node is specified may be obtained from the ITMF Data Encoding Specification.

**Table 2. Nodes and Labels for Value Node System**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| float value | 6 | float | sysVal.float |
| int value | 9 | integer | sysVal.interger |
| bool value | 11 | boolean | sysVal.boolean |
| image resolution | 12 | imageResolution | sysVal.imageResolution |
| sun direction | 30 | sunDirection | sysVal.sunDirection |
| enumeration | 57 | enumeration | sysVal.enumeration |
| annotation | 68 | annotaiton | sysVal.annotation |
| string | 84 | string | sysVal.string |
| file | 88 | file | sysVal.file |
| directory | 101 | directoryName | sysVal.directoryName |
| bit mask | 132 | bitMask | sysVal.bitMask |

### Lighting nodes system

This subclause specifies the node names and corresponding labels that comprise the Lighting Nodes System within the Independent Mapping Space. Further information for how each lighting node is specified may be obtained from the ITMF Data Encoding Specification.

**Table 3. Nodes for Lighting Nodes System**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| blackbody emission | 53 | blackBodyEmission | lighting.blackBodyEmission |
| texture emission | 54 | textureEmission | lighting.textureEmission |
| toon point light | 123 | toonPointLight | lighting.toonPointLight |
| toon directional light | 124 | toonDirectionalLight | lighting.toonDirectionalLight |
| quad light | 148 | quadLight | lighting.quadlLght |
| sphere light | 149 | sphereLight | lighting.sphereLight |
| volumetric spotlight | 152 | volumetricSpotlight | lighting.volumetricSpotLight |

### Geometry nodes system

This subclause specifies the node names and corresponding labels that comprise the Geometry Nodes System within the Independent Mapping Space. Further information for how each geometry node is specified may be obtained from the ITMF Data Encoding Specification.

**Table 4. Nodes and labels for Geometry Nodes System**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| mesh | 1 | mesh | node.mesh |
| geometry group | 3 | geometryGroup | node.geometryGroup |
| placement | 4 | placement | node.placement |
| scatter | 5 | scatter | node.scatter |
| volume | 91 | volume | node.volume |
| joint | 102 | joint | node.joint |
| plane | 110 | plane | node.plane |
| vectron | 133 | vectron | node.vectron |
| geometric primitive | 153 | geometricPrimitive | node.geometricPrimitive |
| union | 154 | union | node.union |
| subtract | 155 | subtract | node.subtract |
| scatter on surface | 164 | scatterOnSurface | node.scatterOnSurface |
| scatter in volume | 165 | scatterInVolume | node.scatterInVolume |

### Environment nodes system

This subclause specifies the node names and corresponding labels that comprise the Environment Nodes System within the Independent Mapping Space. Further information for how each environment node is specified may be obtained from the ITMF Data Encoding Specification.

**Table 5. Nodes and labels for Environment Nodes System**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| daylight environment | 14 | daylight | environment.daylight |
| texture environment | 37 | texture | environment.texture |
| planetary environment | 129 | planetary | environment.planetary |

### Camera nodes system

This subclause specifies the node names and corresponding labels that comprise the Camera Nodes System within the Independent Mapping Space. Further information for how each camera node is specified may be obtained from the ITMF Data Encoding Specification.

**Table 6. Nodes and labels for Camera Nodes System**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| thin lens camera | 13 | thinlens | camera.thinlens |
| panoramic camera | 62 | panoramic | camera.panoramic |
| baking camera | 94 | baking | camera.baking |
| OSL camera | 126 | osl | camera.osl |
| OSL baking camera | 128 | oslBaking | camera.oslBaking |
| universal camera | 157 | universal | camera.universal |

### Materials nodes system

This subclause specifies the node names and corresponding labels that comprise the Material Nodes System within the Independent Mapping Space. Further information for how each materials node is specified may be obtained from the ITMF Data Encoding Specification.

**Table 7. Nodes and labels for Material Nodes System.**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| material map | 2 | map | material.map |
| glossy material | 16 | glossy | material.glossy |
| diffuse material | 17 | diffuse | material.diffuse |
| specular material | 18 | specular | material.specular |
| mix material | 19 | mix | material.mix |
| portal material | 20 | portal | material.portal |
| metallic material | 120 | metallic | material.metallic |
| toon material | 121 | toon | material.toon |
| toon ramp material | 122 | toonRamp | material.toonRamp |
| universal material | 130 | universal | material.universal |
| composite material | 138 | composite | material.composite |
| specular material layer | 139 | specularMaterialLayer | material.specularMaterialLayer |
| diffuse material layer | 140 | diffuseMaterialLayer | material.diffuseMaterialLayer |
| metallic material layer | 141 | metallicMaterialLayer | material.metallicMaterialLayer |
| sheen material layer | 142 | sheenMaterialLayer | material.sheenMaterialLayer |
| layered material | 143 | layered | material.layered |
| material layer group | 144 | materialLayerGroup | material.materialLayerGroup |
| shadow catcher | 145 | shadowCatcher | material.shadowCatcher |
| hair material | 147 | hair | material.hair |
| null material | 159 | null | material.null |
| clipping material | 178 | clipping | material.clipping |

### Object layer nodes system

This subclause specifies the node names and corresponding labels that comprise the Object Layer Nodes System within the Independent Mapping Space. Further information for how each object layer node is specified may be obtained from the ITMF Data Encoding Specification.

**Table 8. Nodes and labels for Object Layer Nodes System.**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| object layer map | 64 | objectLayerMap | layer.object |
| object layer | 65 | objectLayer | layer.objectLayer |

### Surface (medium) nodes system

This subclause specifies the node names and corresponding labels that comprise the Surface Nodes System within the Independent Mapping Space. Further information for how each surface node is specified may be obtained from the ITMF Data Encoding Specification.

**Table 9. Nodes and labels for Surface Nodes System.**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| absorption medium | 58 | absorption | surface.absorption |
| scattering medium | 59 | scattering | surface.scattering |
| Schlick phase function | 60 | schlickPhaseFunction | surface.schlickPhaseFunction |
| volume gradient | 95 | volumeGradient | surface.volumeGradient |
| volume medium | 98 | volume | surface.volume |
| random walk medium | 146 | randomWalk | surface.randomWalk |

### Texture nodes system

This subclause specifies the node names and corresponding labels that comprise the Texture Nodes System within the Independent Mapping Space. The image node is described within this subclause. Further information for the remaining texture nodes may be obtained from the ITMF Data Encoding Specification.

**Table 10. Nodes and labels for Texture Nodes System**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| turbulence texture | 22 | turbulence | texture.turbulence |
| grayscale colour | 31 | grayscaleColour | texture.grayscaleColour |
| gaussian spectrum | 32 | gaussianSpectrum | texture.gaussianSpectrum |
| RGB colour | 33 | rgbColour | texture.rgbColour |
| RGB image | 34 | rgbImage | texture.rgbImage |
| alpha image | 35 | alphaImage | texture.alphaImage |
| grayscale image | 36 | grayscaleImage | texture.grayscaleImage |
| mix texture | 38 | mix | texture.mix |
| multiply texture | 39 | multiply | texture.multiply |
| cosine mix texture | 40 | cosineMix | texture.cosineMix |
| clamp texture | 41 | clamp | texture.clamp |
| saw wave texture | 42 | sawWave | texture.sawWave |
| triangle wave texture | 43 | triangleWave | texture.triangleWave |
| sine wave texture | 44 | sineWave | texture.sineWave |
| checks texture | 45 | checks | texture.checks |
| invert texture | 46 | invert | texture.invert |
| marble texture | 47 | marble | texture.marble |
| rigid fractal texture | 48 | rigidFractal | texture.rigidFractal |
| gradient map texture | 49 | gradientMap | texture.gradientMap |
| falloff map | 50 | falloffMap | texture.falloffMap |
| colour correction | 51 | colourCorrection | texture.colourCorrection |
| dirt texture | 63 | dirt | texture.dirt |
| cylindrical | 74 | cylindrical | texture.cylindrical |
| linear XYZ to UVW | 75 | linearXYZtoUVW | texture.linearXYZtoUVW |
| perspective | 76 | perspective | texture.perspective |
| spherical | 77 | spherical | texture.shperical |
| mesh UV | 78 | meshUV | texture.meshUV |
| box | 79 | box | texture.box |
| displacement | 80 | displacement | texture.displacement |
| random colour texture | 81 | randomColour | texture.randomColour |
| noise texture | 87 | noise | texture.noise |
| polygon side | 89 | polygonSide | texture.polygonSide |
| vertex displacement | 97 | vertexDisplacement | texture.vertexDisplacement |
| W-coordinate | 104 | wCoordinate | texture.wCoordinate |
| add texture | 106 | add | texture.add |
| comparison | 107 | comparison | texture.comparison |
| subtract texture | 108 | subtract | texture.subtact |
| triplanar map | 109 | triplanarMap | texture.triplanarMap |
| triplanar | 111 | triplanar | texture.triplanar |
| instance colour | 113 | instanceColour | texture.instanceColour |
| instance range | 114 | instanceRange | texture.instanceRange |
| baking texture | 115 | baking | texture.baking |
| OSL texture | 117 | oslTexture | texture.oslTexture |
| UVW transform | 118 | uvwTransform | texture.uvwTransform |
| OSL | 125 | osl | texture.osl |
| OSL delayed UV | 127 | oslDelayedUV | texture.oslDelayedUV |
| image tiles | 131 | imageTiles | texture.imageTiles |
| colour vertex attribute texture | 135 | colourVertexAttribute | texture.colourVertexAttribute |
| grayscale colour vertex attribute texture | 136 | grayscaleColourVertexAttribute | texture.grayscaleColourVertexAttribute |
| vertex displacement mixer | 151 | vertexDisplacementMixer | texture.vertexDisplacementMixer |
| spotlight distribution | 158 | spotlightDistribution | texture.spotlightDistribution |
| Cinema4D noise texture | 162 | cinema4dNoise | texture.cinema4dNoise |
| chaos texture | 170 | chaos | texture.chaos |
| channel picker | 171 | channelPicker | texture.channelPicker |
| channel merger | 172 | channelMerger | texture.channelMerger |
| ray switch | 173 | raySwitch | texture.raySwitch |
| channel inverter | 174 | channelInverter | texture.channelInverter |
| channel mapper | 175 | channelMapper | texture.channelMapper |
| composite texture | 176 | composite | texture.composite |
| mesh UV | 177 | meshUV | texture.meshUV |
| iridescent texture | 187 | iridescent | texture.iridescent |
| volume to texture | 256 | volumeToTexture | texture.volumeToTexture |
| colour to UVW | 258 | colourToUVW | texture.colourToUVW |
| smooth Voronoi contours texture | 260 | smoothVoronoiContours | texture.smoothVoronoiContours |
| tile patterns texture | 261 | tilePatterns | texture.tilePatterns |
| procedural effects texture | 262 | proceduralEffects | texture.procedurealEffects |
| chainmail texture | 263 | chainmail | texture.chainmail |
| Moire mosaic texture | 264 | moireMosaic | texture.moireMosaic |
| colour squares texture | 265 | colourSquares | texture.colourSquares |
| stripes texture | 266 | stripes | texture.stripes |
| flakes texture | 267 | flakes | texture.flakes |
| fractal texture | 268 | fractal | texture.fractal |
| hagelslag texture | 269 | hagelslag | texture.hagelslag |
| glowing circle texture | 270 | glowingCircle | texture.glowingCircle |
| curvature texture | 271 | curvature | texture.curvature |
| cell noise texture | 275 | cellNoise | texture.cellNoise |
| sample position to UV | 317 | samplePositionToUV | texture.samplePositionToUV |
| composite texture layer | 318 | compositeTextureLayer | texture.compositeTextureLayer |
| floats to colour | 320 | floatsToColour | texture.floatsToColour |
| float3 to colour | 321 | float3ToColour | texture.float3ToColour |
| distorted mesh UV | 322 | distortedMeshUV | texture. distortedMeshUV |
| capture to custom AOV | 323 | captureToCustomAOV | texture.captureToCustomAOV |
| float to greyscale | 324 | floatToGreyscale | texture.floatToGreyscale |
| ray direction | 326 | rayDirection | texture.rayDirection |
| normal texture | 327 | normal | texture.normal |
| position texture | 328 | position | texture.position |
| relative distance | 329 | relativeDistance | texture.relativeDistance |
| UV coordinate | 330 | uvCoordinate | texture.uvCoordinate |
| Z depth | 331 | zDepth | texture.zDepth |
| gradient generator texture | 332 | gradientGenerator | texture.gradientGenerator |
| random map | 333 | randomMap | texture.randomMap |
| range texture | 334 | range | texture.range |
| surface tangent uPdu | 335 | surfaceTangentUPdu | texture.surfaceTangentUPdu |
| surface tangent uPdv | 336 | surfaceTangentUPdv | texture.surfaceTangentUPdv |
| sample position | 337 | samplePosition | texture.samplePosition |
| matcap | 338 | matcap | texture.matcp |
| binary math operation | 339 | binaryMathOperation | texture.binaryMathOperation |
| unary math operation | 340 | unaryMathOperation | texture.unaryMathOperation |
| image | none | image | texture.image |

**image** describes a single image file, or the URI to an image file, that is part of the scene-based media.

### Transform nodes system

This subclause specifies the node names and corresponding labels that comprise the Transform Nodes System within the Independent Mapping Space. Further information for how each transform node is specified may be obtained from the ITMF Data Encoding Specification.

**Table 11. Nodes and Labels for Transform Node System**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| 3D transformation | 27 | 3dTransformation | transform.3dTransformation |
| 3D scale | 28 | 3dScale | transform.3dScale |
| 3D rotation | 29 | 3dRotation | transform.3dRotation |
| 2D transform | 66 | 2dTransform | transform.2dTransform |
| transform value | 67 | transformValue | transform.transformValue |

### Renderer instruction nodes system

This subclause specifies the node names and corresponding labels that comprise the Renderer Instruction Nodes System within the Independent Mapping Space. Further information for how each renderer instruction node is specified may be obtained from the ITMF Data Encoding Specification.

**Table 12. Nodes and Labels for Renderer Instruction Nodes System**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| camera imager | 15 | cameraImager | render.cameraImager |
| render target | 56 | renderTarget | render.renderTarget |
| post processing | 61 | postProcessing | render.postProcessing |
| render passes | 86 | renderPasses | render.renderPasses |
| render layer | 90 | renderLayer | render.renderLayer |
| animation settings | 99 | animationSettings | render.animationSettings |
| film settings | 100 | filmSettings | render.filmSettings |

### OpenColorIO nodes system

This subclause specifies the node names and corresponding labels that comprise the OpenColorIO Nodes System within the Independent Mapping Space. Further information for how each OpenColorIO node is specified may be obtained from the ITMF Data Encoding Specification.

**Table 13. Nodes and Labels for OpenColorIO Node System**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| OCIO colour view | 160 | colourView | ocio.colourView |
| OCIO colour look | 161 | colourLook | ocio.colourLook |
| OCIO colour space | 163 | colourSpace | ocio.colourSpace |

### Render Arbitrary Output Variables (AOVs) nodes system

This subclause specifies the node names and corresponding labels that comprise the Render Arbitrary Output Variables (AOVs) Node System within the Independent Mapping Space. Further information for how each Render AOV node is specified may be obtained from the ITMF Data Encoding Specification.

**Table 14 Nodes and Labels for Render Arbitrary Output Variables Node System**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| render AOV group | 179 | renderAovGroup | aov.renderAovGroup |
| ambient occlusion AOV | 183 | ambientOcclusion | aov.ambientOcclusion |
| baking group ID AOV | 184 | bakingGroupId | aov.bakingGroupId |
| cyrptomatte AOV | 185 | cyrptomatte | aov.cyrptomatte |
| custom AOV | 186 | custom | aov.custom |
| diffuse AOV | 188 | diffuse | aov.diffuse |
| diffuse direct AOV | 189 | diffuseDirect | aov.diffuseDirect |
| denoised diffused direct AOV | 190 | denoisedDiffusedDirect | aov.denoisedDiffusedDirect |
| diffuse filter (beauty) AOV | 191 | diffuseFilterBeauty | aov.diffuseFilterBeautty |
| diffuse filter (info) AOV | 192 | diffuseFilterInfo | aov.diffuseFilterInfo |
| diffuse indirect AOV | 193 | diffuseIndirect | aov.diffuseIndirect |
| denoised diffused indirect AOV | 194 | denoisedDiffusedIndirect | aov.denoisedDiffusedIndirect |
| emitters | 196 | emitters | aov.emitters |
| environment AOV | 197 | environment | aov.environment |
| normal (geometric) AOV | 198 | normalGeometric | aov.normalGeometric |
| global texture AOV | 199 | globalTexture | aov.globalTexture |
| index of refraction AOV | 200 | indexOfRefraction | aov.indexOfRefraction |
| irradiance AOV | 201 | irradiance | aov.irradiance |
| black layer shadows AOV | 202 | blackLayerShadows | aov.blackLayerShadows |
| layer reflections AOV | 203 | layerReflections | aov.layerRefelctions |
| layer shadows AOV | 204 | layerShadows | aov.layerShadows |
| light AOV | 205 | light | aov.light |
| light direct AOV | 206 | lightDirect | aov.lightDirect |
| light direction AOV | 207 | lightDirection | aov.lightDirection |
| light indirect AOV | 208 | lightIndirect | aov.lightIndirect |
| light pass ID AOV | 209 | lightPassId | aov.lightPassId |
| material ID AOV | 210 | materialId | aov.materialId |
| motion vector AOV | 211 | motionVector | aov.motionVector |
| noise AOV | 212 | noise | aov.noise |
| object ID AOV | 217 | objectId | aov.objectId |
| object layer colour AOV | 218 | objectLayerColour | aov.objectLayerColour |
| opacity AOV | 219 | opacity | aov.opacity |
| position AOV | 220 | position | aov.position |
| post processing AOV | 221 | postProcessing | aov.postProcessing |
| reflection AOV | 222 | reflection | aov.reflection |
| reflection direct AOV | 223 | reflectionDirect | aov.reflectionDirect |
| denoised reflection direct AOV | 224 | denoisedReflectionDirect | aov.denoisedReflectionDirect |
| reflection filter (beauty) AOV | 225 | reflectionFilterBeauty | aov.reflectionFilterBeauty |
| reflection filter (info) AOV | 226 | reflectionFilterInfo | aov.refelctionFilterInfo |
| reflection indirect AOV | 227 | reflectionIndirect | aov.reflectionIndirect |
| denoised reflection indirect AOV | 228 | denoisedReflectionIndirect | aov.denoisedReflectionIndirect |
| refraction AOV | 229 | refraction | aov.refraction |
| refraction filter (beauty) AOV | 230 | refractionFilterBeauty | aov.refractionFilterBeauty |
| denoised remainder AOV | 232 | denoisedRemainder | aov.denoisedRemainder |
| render layer ID AOV | 233 | renderLayerId | aov.renderLayerId |
| render layer mask AOV | 234 | renderLayerMask | aov.renderLayerMask |
| roughness AOV | 235 | roughness | aov.roughness |
| normal (shading) AOV | 236 | normalShading | aov.normalShading |
| shadow AOV | 237 | shadow | aov.shadow |
| subsurface scattering AOV | 238 | subsurfaceScattering | aov.subsurfactScattering |
| normal (tangent) AOV | 239 | normalTangent | aov.normalTangent |
| texture tangent AOV | 240 | textureTangent | aov.textureTangent |
| transmission filter (beauty) AOV | 242 | transmissionFilterBeauty | aov.transmissionFilterBeauty |
| transmission filter (info) AOV | 243 | transmissionFilterInfo | aov.transmissionFilterInfo |
| transmission AOV | 244 | transmission | aov.transmission |
| UV coordinates AOV | 245 | uvCoordinates | aov.uvCoordinates |
| volume AOV | 246 | volume | aov.volume |
| denoised volume AOV | 247 | denoisedVolume | aov.denoisedVolume |
| volume emission AOV | 248 | volumeEmission | aov.volumeEmission |
| denoised volume emission AOV | 249 | denoisedVolumeEmission | aov.denoisedVolumeEmission |
| volume mask AOV | 250 | volumeMask | aov.volumeMask |
| volume Z-depth front AOV | 251 | volumeZDepthFront | aov.volumeZDepthFront |
| volume Z-depth back AOV | 252 | volumeZDepthBack | aov.volumeZDepthBack |
| normal (smooth) AOV | 253 | normalSmooth | aov.normalSmooth |
| wireframe AOV | 254 | wireframe | aov.wireframe |
| Z-depth AOV | 255 | zDepth | aov.zDepth |

### Arbitrary Output Variables (AOVs) compositor nodes system

This subclause specifies the node names and corresponding labels that comprise the Arbitrary Output Variables (AOVs) Nodes System within the Independent Mapping Space. Further information for how each graph type node is specified may be obtained from the ITMF Data Encoding Specification.

**Table 15. Nodes and Labels for Arbitrary Output Variables Compositor Nodes System**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| composite AOV output | 166 | composite | aovCompositor.composite |
| AOV output group | 167 | group | aovCompositor.group |
| image AOV output | 168 | image | aovCompositor.image |
| render AOV output | 169 | render | aovCompositor.render |
| colour AOV output | 177 | colour | aovCompositor.colour |
| composite AOV output layer | 319 | compositeLayer | aovCompositor.compositeLayer |
| colour correction AOV output | 373 | colourCorrection | aovCompositor.colourCorrection |
| clamp AOV output | 374 | clamp | aovCompositor.clamp |
| map range AOV output | 375 | mapRange | aovCompositor.mapRange |
| light mixer AOV output | 376 | lightMixer | aovCompositor.lightMixer |

### Kernel nodes system

This subclause specifies the node names and corresponding labels that comprise the Kernel Nodes System within the Independent Mapping Space. Further information for how each kernel node is specified may be obtained from the ITMF Data Encoding Specification.

**Table 16. Nodes and Labels for Kernel Nodes System**

| **Node** | **Code Point** | **Label** | **Usage** |
| --- | --- | --- | --- |
| PMC kernel | 23 | pmc | kernel.pmc |
| direct lighting kernel | 24 | directLighting | kernel.directLighting |
| path tracing kernel | 25 | pathTracing | kernel.pathTracing |
| info channels kernel | 26 | infoChannels | kernel.infoChannels |

### Node graph types system

This subclause specifies the node names, their definitions, and corresponding labels that comprise the Graph Types Nodes System within the Independent Mapping Space. There are no code points for the Graph Types Nodes System.

**Table 17. Nodes and Labels for Graph Types Node System**

| **Node** | **Label** | **Usage** |
| --- | --- | --- |
| scene graph | sceneGraph | graph.sceneGraph |
| geometry archive | geometryArchive | graph.geometryArchive |
| project settings | projectSettings | graph.projectSettings |
| scripted graph | scriptedGraph | graph.scriptedGraph |

**sceneGraph** describes an entire scene, consistent with the definition of a scene graph.

**geometryArchive** describes an individual geometric object.

**projectSettings** describes a collection of nodes and values to serve as default settings for the graph, e.g., the camera to be used, lighting, and the path to objects that might exist externally to the graph.

**scriptedGraph** describes a script that can be used to create input to the scene graph, e.g., a compressed video stream that is first reconstructed and then linked to a particular node or object within a scene.

### Buffer system

This subclause specifies the node names, their definitions, and corresponding labels that comprise the Buffer Nodes System within the Independent Mapping Space. There are no code points for the Buffer Nodes System.

**Table 18. Nodes and labels for buffer nodes system**

| **Node** | **Label** | **Usage** |
| --- | --- | --- |
| binary blob | binaryBlob | buffer.binaryBlob |
| buffer specification | bufferSpecification | buffer.bufferSpecification |
| GLB buffer | GLBBuffer | buffer.GLBBuffer |
| Open SubDiv buffer | openSubDivBuffer | buffer.openSubDivBuffer |
| shading buffer | shadingBuffer | buffer.shadingBuffer |
| asset buffer | assetBuffer | buffer.assetBuffer |
| accessor | accessor | buffer.accessor |
| accessor sparse | accessorSparse | buffer.accessorSparse |
| accessor sparse indices | accessorSparseIndices | buffer.accessorSparseIndices |
| accessor sparse values | accessorSparseValues | buffer.accessorSparseValues |

**binaryBlob** describes …

**bufferSpecification** describes …

**GLBBuffer** describes …

**openSubDiv buffer** describes …

**shading Buffer** describes …

**asset Buffer** describes …

**accessor** describes …

**accessorSparse** describes …

**accessorSparseIndices** describes …

**accessorSparseValues** describes …

### Animation system

This subclause specifies the node names, their definitions, and corresponding labels that comprise the animation nodes system within the Independent Mapping Space. There are no code points for the animation nodes system.

**Table 19. Nodes and labels for animation nodes system**

| **Node** | **Label** | **Usage** |
| --- | --- | --- |
| data type | dataType | animation.dataType |
| period | period | animation.period |
| pattern | pattern | animation.pattern |
| animation type | animationType | animation.animationType |
| end time | endTime | animation.endTime |
| node target | nodeTarget | animation.nodeTarget |
| input accessor | inputAccessor | animation.inputAccessor |
| output accessor | outputAccessor | animation.outputAccessor |
| interpolation | interpolation | animation.interpolation |
| channel | channel | animation.channel |
| animation settings | animationSettings | animation.animationSettings |

**dataType** describes …

**period** describes …

**pattern** describes …

**animationType** describes …

**endTime** describes …

**nodeTarget** describes …

**inputAccessor** describes …

**outputAccessor** describes …

**interpolation** describes …

**channel** describes …

**animationSettings** describes …

### Pins data system

Pin data (item 3 in Figure 4) are represented by labels that signal the type of data that are used as inputs or outputs between other node systems as shown in Figure 4. This subclause specifies the pin data types and corresponding labels that comprise the Pins Data System within the Independent Mapping Space. The complete list of individual pins is specified in the ITMF Data Encoding Specification. At the date of this specification, there are over 700 different pins that comprise the complete set of pins available to further define each of the pin data types. These pins and their corresponding labels are defined in the ITMF Data Encoding Specification.

**Table 20. Pins data system**

| **Pin data type** | **Label** |
| --- | --- |
| bool | 1 |
| float | 2 |
| int | 3 |
| transform | 4 |
| texture | 5 |
| emission | 6 |
| material | 7 |
| camera | 8 |
| environment | 9 |
| imager | 10 |
| kernel | 11 |
| geometry | 12 |
| medium | 13 |
| phase function | 14 |
| film settings | 15 |
| enum | 16 |
| object layer | 17 |
| postproc | 18 |
| render target | 19 |
| work plane | 20 |
| projection | 21 |
| displacement | 22 |
| string | 23 |
| render passes | 24 |
| render layer | 25 |
| volume ramp | 26 |
| animation settings | 27 |
| lut | 28 |
| render job | 29 |
| toon ramp | 30 |
| bit mask | 31 |
| round edges | 32 |
| material layer | 33 |
| ocio view | 34 |
| ocio look | 35 |
| ocio colour space | 36 |
| output aov group | 37 |
| output aov | 38 |
| tex composite layer | 39 |
| composite aov layer | 40 |

### Attributes system

Attribute data (item 4 in Figure 4) are represented by labels that signal the type of data that are used to further describe other node systems as shown in Figure 4. In terms of semantics for a hypothetical renderer, the renderer should not allow the values encompassed by attributes to change during serialisation.

This subclause specifies the attribute data types and corresponding labels that comprise the Attributes Data System within the Independent Mapping Space. The complete list of individual attributes is specified in the ITMF Data Encoding Specification. At the date of this specification, there are over 400 different attributes that comprise the complete set of attribute data available to further define each of the attribute data types. These attributes and their corresponding labels are defined in the ITMF Data Encoding Specification.

**Table 21. Attributes data system**

| **Attribute data type** | **Label** |
| --- | --- |
| bool | 1 |
| int | 2 |
| int2 | 3 |
| int3 | 4 |
| int4 | 5 |
| float | 6 |
| float2 | 7 |
| float3 | 8 |
| float4 | 9 |
| string | 10 |
| filename | 11 |
| byte | 12 |
| matrix | 13 |
| long | 14 |
| long2 | 15 |

1. (normative) **Annotation using IMS and ITMF**
   1. **IMS and Immersive Technology Media Format (ITMF)**

This annex specifies the location of metadata information within the ITMF specifications and how to combine such ITMF metadata with IMS metadata to extend IMS metadata in this specification for the purposes of annotating scene-based media or translating scene-based media from one representation to another.

* + 1. **Introduction to Immersive Technology Media Format (ITMF)**

The ITMF is currently comprised of three specifications.

*Immersive Technology Media Format Container Specification, Version 2.0 (17 February 2023)*

*Immersive Technology Media Format Data Encoding Specification, Version 2.0 (17 February 2023)*

*Immersive Technology Media Format Scene Graph Specification, Version 2.0 (01 December 2022)*

The ITMF Data Encoding Specification provides metadata that augment, but do not replace, metadata in the IMS. The ITMF Scene Graph Specification describes metadata and relationships between ITMF metadata in the context of an ITMF scene. The ITMF Container Specification describes how to package and encrypt scene assets and an ITMF scene graph into a single file, i.e., “container.”

In cases where metadata from ITMF Data Encoding duplicates metadata from the IMS, the IMS metadata is used.

* + 1. **On the use of ITMF and IMS to annotate scene media**
       1. **IMS metadata**

The metadata in this specification guides the use of the ITMF metadata that can be used to supplement the IMS. In some cases, there is no ITMF metadata to further describe IMS metadata. In such cases, the metadata that is used to annotate scene-based media is entirely specified in this part.

The IMS is a superset of ITMF metadata. The difference between IMS and ITMF metadata is that the ITMF does not define metadata for certain features of scene-based media, for example, to describe the organization of an asset’s binary data within a buffer of memory. In this case, the IMS provides metadata that can be used to describe the organization of a buffer without any reference to the ITMF specifications.

For all cases where ITMF metadata can be used to supplement IMS metadata, the description of the IMS metadata includes an ITMF code point, i.e., a numeric value that can be used to locate the corresponding ITMF description. Associated with the description of that code point in the ITMF Data Encoding Specification, there may be a list of pins and attributes that can be used to extend the IMS metadata.

* + - 1. **Example**

Both the IMS and ITMF can be used to create metadata for a specular material node, which has a code point value of 18 as identified in the materials node subsystem in the IMS.

An annotation application may wish to identify a particular node within scene-based media as a specular material node using the label “material.specular” from the IMS. However, the ITMF Data Encoding Specification lists several pins that may be used to further describe the specular material node.

Using the code point 18 to identify the correct list of pins and attributes that may extend the IMS metadata frpm the ITMF Date Encoding Specification for the specular material node, **Table 22** identifies the list of pins that serve as properties of a specular material node, i.e., to enable a renderer to create the specular material.

**Table 22 ITMF pins for specular material node (18)**

|  |  |
| --- | --- |
| reflection | smooth |
| transmission | smoothShadowTerminator |
| brdf | roundEdges |
| roughness | medium |
| anistropy | fake\_shadows |
| rotation | refractionAlpha |
| spread | thinWall |
| index | filmwidth |
| dispersion\_coefficient\_B | filmindex |
| bump | priority |
| normal | customAov |
| displacement | customAovChannel |
| opacity | layer |

Referring to ITMF Data Encoding specification and its complete list of pin names in alphabetical order, **Table 23** provides the labels that are used as the metadata for the pins in **Table 22**.

Note: For the purposes of this example, the pin labels are taken from Table 294 in Version 2.0 of the ITMF Data Encoding Specification.

**Table 23. Pins and pin labels for specular material node (18)**

| **Pin** | **Pin label** | **Usage** |
| --- | --- | --- |
| reflection | Reflection | material.specular.reflection |
| transmission | Transmission | material.specular.transmission |
| brdf | Brdf | material.specular.brdf |
| roughness | Roughness | material.specular.roughness |
| anistropy | Anistropy | material.specular.anistropy |
| rotation | Rotation | material.specular.rotation |
| spread | Spread | material.specular.spread |
| index | Index | material.specular.index |
| dispersion\_coefficient\_B | DispersionCoefficientB | material.specular.dispersionCoefficientB |
| bump | Bump | material.specular.bump |
| normal | Normal | material.specular.normal |
| displacement | Displacement | material.specular.displacement |
| opacity | Opacity | material.specular.opacity |
| smooth | Smooth | material.specular.smooth |
| smoothShadowTerminator | SmoothShadowTerminator | material.specular.smoothShadowTerminator |
| roundEdges | RoundEdges | material.specular.roundEdges |
| medium | Medium | material.specular.medium |
| fake\_shadows | FakeShadows | material.specular.fakeShadows |
| refractionAlpha | RefractionAlpha | material.specular.refractionAlpha |
| thinWall | ThinWall | material.specular.thinWall |
| filmwidth | FilmWidth | material.specular.filmWidth |
| filmindex | FilmIndex | material.specular.filmIndex |
| priority | Priority | material.specular.priority |
| customAov | CustomAov | material.specular.customAoV |
| customAovChannel | CustomAovChannel | material.specular.customAovChannel |
| layer | Layer | material.specular.layer |

Although the labels shown in **Table 23** illustrate different use of case formats, an annotation or translation process shall not be sensitive to case when processing

* 1. Initial annotation process for scene media with IMS metadata

An IMS-annotated scene or media asset is derived by the following steps.

1. If the specification that defines the format of the asset, e.g., the glTF 2.0 specification for glTF assets, provides a mechanism to store annotation metadata, that mechanism shall be used to record IMS metadata into the asset or scene. Otherwise, the IMS metadata shall be stored into the scene or media asset via the commenting mechanism supported by the format.
2. IMS labels are stored into the media file either in the form of comments or according to the mechanism provided by the media format.

Note: the proximity of the label to the media content that it describes within the file should be such that upon visual inspection of the annotation, it is obvious that the label describes the portion of the file to which it is closely located.

1. Starting with the value of 0, each label that is recorded into the media file is assigned a unique integer value in ascending order.
2. An IMS annotation header that conforms to the data model specified in Clause C.1 shall be created and stored at the beginning of the file for the scene or media asset.

Note: the value portion of the MD5 keyword and value pair is empty when initially recording the header into the media file.

1. The next integer ID value that should be used in the consecutive numbering process of storing annotation lables into the media file is stored as the value portion of the NEXT\_ID keyword and value pair of the annotation header.
2. An MD5 value string is created for the media file containing the annotated scene or media asset.
3. The MD5 value string is stored in the value portion of the MD5 keyword and value pair within the annotation header.
   1. Fixity checking for scene media that is already annotated with IMS metadata

Remove and save the MD5 value from the annotation header of the scene media file. The scene media file should be restored to its exact contents prior to step 7 in the process specified in Clause A.2

Regenerate the MD5 using the annotated scene file with the MD5 value now removed from its header.

The MD5 regenerated in this process should match the MD5 value that was previously stored in the header.

* 1. Subsequent modification of scene media that is already annotated with IMS metadata

Update the media i.e., insert, remove, or change the media in the media file.

If any updates to the media result in the removal of existing IMS metadata, then the process to remove all existing IMS metadata should be followed.

Regenerate the IMS metadata following the Initial annotation process specified in clause A.2.

If the update to the scene media is to insert additional media into the media file, then the following process is executed.

Starting with the integer value at the NEXT ID keyword pair, annotate the newly inserted media using the next integer value.

Update the value for the NEXT ID keyword and value pair in the annotation header.

Regenerate the MD5.

Store the newly regenerated MD5 in the value portion of the MD5 keyword and value pair in the annotation header.

1. (normative) **Annotation of glTF**
   1. **Mapping of glTF 2.0 properties to IMS**

This clause provides a mapping of glTF 2.0 properties to the IMS. Not all properties are mapped to IMS values as they may carry application specific metadata or authorship information.

The KHR\_xmp\_jsonld is used to annotate glTF media.

glTF properties are mapped to the IMS as follows:

| **glTF 2.0 property** | **IMS identifiers** |
| --- | --- |
| Accessor | buffer.accessor |
| Accessor Sparse | buffer.accessorSparese |
| Accessor Sparse Indices | buffer.accessorSparseIndices |
| Accessor Sparse Values | buffer.accessorSparseValues |
| Animation | animation |
| Animation Channel | animation.channel |
| Animation Chanel Target | animation.target |
| Animation Sampler | animation.sampler |
| Asset | graph.geometryArchive |
| Buffer | buffer |
| Buffer View | buffer.bufferSpecification |
| Camera | camera |
| Camera Orthographic | camera.universal.orthographic |
| Camera Perspective | camera.thinlens |
| glTF | graph.sceneGraph |
| Image | texture.image |
| Material | Material |
| Material Normal Texture Info | Material |
| Material Occlusion Texture Info | Material |
| Material PBR Metallic Roughness | Material |
| Mesh | Geometry |
| Mesh Primitive | Geometry |
| Node | Transformation |
| Sampler | Render Instructions (Animations?) |
| Scene | Graph Type |
| Skin | Texture |
| Texture | Texture |
| Texture Info | Texture |

* + 1. **Subclause title**

Type text.

* + 1. **Subclause title**
       1. **Subclause title**

Type text.

* + - 1. **Subclause title**

Type text.

* + - 1. **Subclause title**

Type text.

1. (Normative) **IMS Annotation data model** 
   1. **Annotation header**

Specifies a header structure to include:

Systems used

Next ID

MD5

* + 1. **Systems used keyword and value pair**
    2. **Next ID keyword and value pair**
    3. **MD5 keyword and value pair**
  1. **Annotation labels**

Each label is associated with a unique label ID. Labels increment by one starting with the value zero.

1. (informative) **Translation process between scene formats**
   1. **Clause title**

This annex provides an example architecture for a translation process that uses lookup tables.

* + 1. **Subclause title**

Type text.

* + 1. **Subclause title**
       1. **Subclause title**

Type text.

* + - 1. **Subclause title**

Type text.

* + - 1. **Subclause title**

Type text.

**Bibliography**

[1] ISO/IEC 23090-27:202x, *Information technology — Coded representation of immersive media — Part 27 Media and architectures for render-based systems and applications*

1. Under preparation. Stage at time of publication: ISO/IEC DIS 12113:2021. [↑](#footnote-ref-1)