INTERNATIONAL STANDARD

ISO/IEC 14496-15

First edition
2004-04-15

AMENDMENT 2
2008-07-15

Information technology — Coding of audio-visual objects —

Part 15: Advanced Video Coding (AVC) file format

AMENDMENT 2: File format support for Scalable Video Coding (SVC)

Technologies de l'information — Codage des objets audiovisuels —

Partie 15: Format de fichier de codage vidéo avancé (AVC)

AMENDEMENT 2: Support de format de fichier pour codage vidéo extensible (SVC)

Reference number
ISO/IEC 14496-15:2004/Amd.2:2008(E)

© ISO/IEC 2008

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2008

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS

© ISO/IEC 2008 – All rights reserved

Not for Resale

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

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

Amendment 2 to ISO/IEC 14496-15:2004 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information Technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*

--`,,```,,,,````-`-`,,`,,`,`,,`---
© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

--`,,```,,,,````-`-`,,`,,`,`,,`---

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale

Information technology — Coding of audio-visual objects —

Part 15:
Advanced Video Coding (AVC) file format

AMENDMENT 2: File format support for Scalable Video Coding (SVC)

Add to the Introduction:

This International Standard defines the storage for both plain AVC and SVC video streams, where 'plain AVC' refers to the main part of ISO/IEC 14496-10, not including Annex G (Scalable Video Coding), and SVC refers to ISO/IEC 14496-10 when the techniques in Annex G (Scalable Video Coding) are in use. Specific techniques are introduced for the handling of scalable streams, enabling their use, and assisting the extraction of subsets of scalable streams.

Add the following to the end of Clause 1 (Scope):

The file format for storage of SVC content, as defined in Annexes A-E, uses the existing capabilities of the ISO base media file format and the AVC file format. In addition, the following new extensions to support SVC-specific features are specified:

- **Scalable Grouping**: A structuring and grouping mechanism to indicate the association of NAL units with different types and hierarchy levels of scalability.
- **Aggregator**: A structure to enable efficient scalable grouping of NAL units by changing irregular patterns of NAL units into regular patterns of aggregated data units.
- **Extractor**: A structure to enable efficient extraction of NAL units from other tracks than the one containing the media data.
- **Temporal metadata statements**: Structures for storing time-aligned information of media samples.
- **AVC Compatibility**: A provision for storing an SVC bitstream in an AVC compatible manner, such that the AVC compatible base layer can be used by any existing AVC file format compliant reader.

Add the following terms (or replace when the item exists) to 3.2 maintaining alphabetical order:

AVC	Advanced Video Coding. Where contrasted with SVC in this International Standard, this term refers to the main part of ISO/IEC 14496-10, not including Annex G (Scalable Video Coding)
FF	File Format
ROI	Region-Of-Interest
SVC	Scalable Video Coding. Refers to ISO/IEC 14496-10 when the techniques in Annex G (Scalable Video Coding) are in use
VCL	Video Coding Layer

Replace the text of 4.1 with the following:

The technologies originally documented in Clause 4 are now defined in ISO/IEC 14496-12:2008 (technically identical to ISO/IEC 15444-12:2008).

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Replace the text of 4.2 (file identification) with the following:

See 6.3 in ISO/IEC 14496-12:2008.

Replace the text of 4.3 (independent and disposable samples) with the following:

See 8.6.4 in ISO/IEC 14496-12:2008 for the definition of this box.

Replace the text of 4.4 (sample groups) with the following:

See 8.9 in ISO/IEC 14496-12:2008.

Replace the text of 4.5 (random access recovery points) with the following:

See 10.1 in ISO/IEC 14496-12:2008.

Replace the text of 4.6 (representation of new structures) with the following:

See 8.9.4 in ISO/IEC 14496-12:2008.

Add to the end of 5.3.3 (Track Structure):

A video stream is represented by one or more video tracks in a file.

If there is more than one track representing scalable aspects of a single stream, then they form alternatives to each other, and the field '`alternate_group`' should be used, or the composition system used should select one of them, as appropriate. See 10.3 "Grouping and labelling of tracks" of ISO/IEC 14496-12:2008 for informative labelling of why tracks are members of alternate groups.

Add this new subclause as 5.3.16 Definition of a sub-sample for AVC:

For the use of the sub-sample information box (7.7 of ISO/IEC 14496-12:2008) in an AVC stream, a sub-sample is defined as one or more contiguous NAL units within a sample and having the same value of the following fields; `RefPicFlag`, `RedPicFlag` and `VclNalUnitFlag`. Each sub-sample includes both NAL unit(s) and their preceding NAL unit length field(s). The presence of this box is optional; however, if present in a track containing AVC data, it shall have the semantics defined here.

The subsample_priority field shall be set to a value in accordance with the specification of this field in ISO/IEC 14496-12.

The discardable field shall be set to 1 only if this sample can still be decoded if this sub-sample is discarded (e.g. the sub-sample consists of an SEI NAL unit, or a redundant coded picture).

The reserved field is defined for AVC as follows:

```
unsigned int(1) RefPicFlag;
unsigned int(1) RedPicFlag;
unsigned int(1) VclNalUnitFlag;
unsigned int(29) reserved = 0;
```

`RefPicFlag` equal to 0 indicates that all the NAL units in the sub-sample have `nal_ref_idc` equal to 0. `RefPicFlag` equal to 1 indicates that all the NAL units in the sub-sample have `nal_ref_idc` greater than 0.

`RedPicFlag` equal to 0 indicates that all the NAL units in the sub-sample have `redundant_pic_cnt` equal to 0. `RedPicFlag` equal to 1 indicates that all the NAL units in the sub-sample have `redundant_pic_cnt` greater than 0.

`VclNalUnitFlag` equal to 0 indicates that all NAL units in the sub-sample are non-VCL NAL units. Value 1 indicates that all NAL units in the sub-sample are VCL NAL units.

--`,,```,,,,````-`-`,,`,,`,`,,`---
© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Insert the following Annexes A to E:

Annex A
(normative)

SVC elementary stream and sample definitions

A.1 Introduction

This annex specifies the basic storage format of SVC data. It extends the definitions of the storage format of AVC in Clause 5.

A.2 Terms and definitions

For the purpose of Annex A to Annex E, inclusive, the following terms and definitions apply. The definitions in ISO/IEC 14496-10 (including Annex G) also apply.

A.2.1
Aggregator
Aggregators are in-stream structures using a NAL unit header including a NAL unit header SVC extension, with a NAL unit type equal to 30. Aggregators are used to group NAL units belonging to the same sample.

A.2.2
AVC base layer
The AVC base layer is the maximum subset of a scalable bitstream that is AVC compatible – a bitstream not using any of the functionality of ISO/IEC 14496-10 Annex G. The AVC base layer is represented by AVC VCL NAL units and associated non-VCL NAL units.

NOTE The AVC base layer itself may be a temporal scalable bitstream.

A.2.3
AVC NAL units
AVC VCL NALs collectively refer to AVC VCL NAL units and their associated non-VCL NAL units in a bitstream.

A.2.4
AVC VCL NAL unit
AVC VCL NAL units are NAL units with type in the range of 1 to 5 (inclusive).

A.2.5
Extraction path
An extraction path is a set of operations on the original bitstream, each yielding a subset bitstream, ordered such that the complete bitstream is first in the set, and the base layer is last, and all the bitstreams are in decreasing complexity (along one of the scalability axes, such as resolution), and where every bitstream is a valid operating point.

NOTE An extraction path may be represented by the values of priority_id in the NAL unit headers. Alternatively an extraction path can be represented by the run of tiers or by a set of hierarchically dependent tracks.

A.2.6
Extractor
Extractors are in-stream structures using a NAL unit header including a NAL unit header SVC extension, with a NAL unit type equal to 31. Extractors contain instructions on how to extract data from other tracks. Logically an Extractor can be seen as a 'link'. While accessing a track containing Extractors, the Extractor is replaced by the data it is referencing.

--`,,```,,,,````-`-`,,`,,`,`,,`---
© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

A.2.7
In-stream
In-stream structures reside within sample data.

A.2.8
Operating point
A subset of a scalable bitstream, representing a particular spatial resolution, temporal resolution, and quality. Each operating point consists of all the data needed to decode this particular bitstream subset.

NOTE In an SVC stream an operating point can be represented either by (i) specific values of DTQ (dependency_id, temporal_id and quality_id) or (ii) specific values of P (priority_id) or (iii) combinations of them (e.g. PDTQ). Note that the usage of priority_id is defined by the application. In an SVC file a track represents one or more operating points. Within a track tiers may be used to define multiple operating points.

A.2.9
Prefix NAL unit
Prefix NAL units are NAL units with type 14. Prefix NAL units provide scalability information about AVC VCL NAL units and filler data NAL units. Prefix NAL units do not affect the decoding process of a legacy AVC decoder. The behaviour of a legacy AVC file reader as a response to prefix NAL units is undefined.

A.2.10
Scalable layer
A scalable layer consists of a set of VCL NAL units with the same values of dependency_id, quality_id, and temporal_id and the associated non-VCL NAL units. A scalable layer with any of dependency_id, quality_id, and temporal_id not equal to 0 enhances the video by one or more scalability levels in at least one direction (temporal, quality or spatial resolution). A scalable layer may also be simply referred to a layer.

NOTE SVC uses a “layered” encoder design which results in a bitstream representing “coding layers”. In some publications the ‘base layer’ is the first quality layer of a specific coding layer. In some publications the base layer is the scalable layer with the lowest priority. The SVC file format uses “scalable layer” or “layer” in a general way for describing nested bitstreams (using terms like AVC base layer or SVC enhancement layer).

A.2.11
Scalable layer representation
A scalable layer representation refers to the bitstream subset that is required for decoding the scalable layer, and consists of the scalable layer itself and all the scalable layers on which the scalable layer depends. A scalable layer representation is also referred to as the representation of the scalable layer.

A.2.12
Sub-picture
A sub-picture consists of a proper subset of coded slices of a layer representation.

A.2.13
Sub-picture tier
A sub-picture tier is a tier that consists of sub-pictures. Any coded slice that is not included in the tier representation of a sub-picture tier shall not be referred to in inter prediction or inter-layer prediction for decoding of the sub-picture tier.

A.2.14
SVC enhancement layer
An SVC enhancement layer specifies a part of a scalable bitstream that enhances the video. An SVC enhancement layer is represented by SVC VCL NAL units and the associated non-VCL NAL units and SEI messages.

NOTE Usually an SVC enhancement layer represents a spatial or coarse-grain scalability (CGS) coding layer (identified by a specific value of dependency_id).

A.2.15
SVC NAL units
SVC NAL units collectively refer to SVC VCL NAL units and their associated non-VCL NAL units in an SVC stream.

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

A.2.16
SVC stream
Let the greatest value of dependency_id of all the operating points represented by DTQ (dependency_id, temporal_id and quality_id) combinations be equal to mDid, and the set of all the operating points with dependency_id equal to mDid be mOpSet. SVC stream refers to the bitstream represented by the operating point for which dependency_id is equal to mDid, temporal_id is the greatest temporal_id value among mOpSet, and quality_id is the greatest quality_id value among mOpSet. The term "SVC stream" is referenced by 'decoding/accessing the entire stream' in this document. There may be NAL units which are not required for decoding this operating point.

A.2.17
SVC VCL NAL unit
SVC VCL NAL units are NAL units with type 20, and NAL units with type 14 when the immediately following NAL units are AVC VCL NAL units. SVC VCL NAL units do not affect the decoding process of a legacy AVC decoder. The behaviour of a legacy AVC file reader as a response to SVC VCL NAL units is undefined.

A.2.18
Tier
Tiers define a set of operating points within a track, providing information about the operating points and instructions on how to access the corresponding bitstream portions (using maps and groups). A tier represents one or more scalable layers of an SVC bitstream.

NOTE The term "tier" is used to avoid confusion with the frequently used term layer. A tier represents a subset of a track and represents an operating point of an SVC bitstream. Tiers in a track subset the entire track, no matter whether the track references another track by extractors.

A.2.19
Tier representation
A tier representation refers to the bitstream subset that is required for decoding the tier, and consists of the tier itself and all the tiers on which the tier depends. A tier representation is also referred to as the representation of the tier.

A.3 Elementary stream structure

SVC streams are stored in accordance with subclause 5.1, with the following definition of an SVC video elementary stream:

- **SVC Video Elementary Streams** shall contain all video coding related NAL units (i.e. those NAL units containing video data or signalling video structure) and may contain non-video coding related NAL units such as SEI messages and access unit delimiter NAL units. Also Aggregators (see B.2) or Extractors (see B.3) may be present. Aggregators and Extractors shall be processed as defined in this International Standard (e.g. shall not directly be placed in the output buffer while accessing the file). Other NAL units that are not expressly prohibited may be present, and if they are unrecognized they should be ignored (e.g. not placed in the output buffer while accessing the file).

SVC streams may also be stored using associated parameter set streams, if needed.

For SVC streams, Table 1 is updated as follows; only entries where the definition for SVC differs from AVC, are shown.

--`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Table A.1 — NAL Unit Types in SVC and AVC Streams

Value of nal_unit_type	Description	AVC video elementary stream	SVC video elementary stream	Parameter set elementary stream
14	Prefix NAL unit in scalable extension prefix_nal_unit_rbsp( )	Not specified	Yes	No
15	Subset sequence parameter set subset_seq_parameter_set_rbsp( )	Not specified	No. If parameter set elementary stream is not used, Subset SPS shall be stored in the Decoder Specific Information.	Yes
20	Coded slice in scalable extension slice_layer_in_scalable_extension_rbsp( )	Not specified	Yes	No
24 – 29	Not specified	Not specified	Not specified	Not specified
30	Aggregator	Not specified	Yes	No
31	Extractor	Not specified	Yes	No

There may be AVC VCL NAL units, SVC VCL NAL units and other NAL units, i.e. non-VCL NAL units, present in an SVC video elementary stream. Additionally, there may be Aggregator NAL units and Extractor NAL units present in an SVC video elementary stream.

An AVC VCL NAL unit in an SVC video elementary stream conforming to one or more profiles specified in Annex G of ISO/IEC 14496-10 shall be immediately preceded by a prefix NAL unit containing the scalability information for the AVC VCL NAL unit. In this file format an AVC VCL NAL unit and the immediately preceding prefix NAL unit are logically seen as one NAL unit: the prefix NAL unit provides the scalability information and the AVC VCL NAL unit provides the NAL unit type and payload.

A.4 Use of the AVC file format

The SVC file format is an extension of the AVC file format defined in this International Standard.

Subclause 5.3.12 is defined for use with plain AVC streams. Its use with SVC streams is deprecated.

A.5 Sample and configuration definition

A.5.1 Introduction

SVC Sample: An SVC sample is also an access unit as defined in subclause 7.4.1.2 of ISO/IEC 14496-10.

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

A.5.2 Canonical order and restrictions

A.5.2.1 Restrictions

The following restrictions apply to SVC data in addition to the requirements in subclause 5.2.2:

- **SVC coded slice NAL units** (Coded slices in scalable extension): All SVC coded slice NAL units for a single instant in time shall be contained in the sample whose composition time is that of the picture represented by the access unit. An SVC sample shall contain at least one AVC or SVC VCL NAL unit.

- **Prefix NAL units** (Prefix NAL unit in scalable extension): Each prefix NAL unit is placed immediately before the corresponding AVC VCL NAL unit, providing scalability information about the AVC VCL NAL unit.

  NOTE Prefix NAL units may also be associated with filler data NAL units, which are not present in elementary streams.

- **Aggregators/Extractors**: The order of all NAL units included in an Aggregator or referenced by an Extractor is exactly the decoding order as if these NAL units were present in a 'plain' sample. After processing the Aggregator or the Extractor, all NAL units must be in valid decoding order as specified in ISO/IEC 14496-10.

A.5.2.2 Decoder configuration record

When the decoder configuration record defined in subclause 5.2.4.1 is used for a stream which can be interpreted as either an SVC or AVC stream, the AVC decoder configuration record shall reflect the properties of the AVC compatible base layer, e.g. it shall contain only parameter sets needed for decoding the AVC base layer.

A parameter set stream may be used with SVC streams, as with AVC streams. In that case, parameter sets shall not be included in the decoder configuration record.

Sequence parameter sets are numbered in order of storage from 1 to `numOfSequenceParameterSets` or `numOfPictureParameterSets` respectively. Sequence and Picture parameter sets stored in this record in a file may be referenced using this 1-based index by the `InitialParameterSetBox`.

The `SVCDecoderConfigurationRecord` is structurally identical to an `AVCDecoderConfigurationRecord`. However, the reserved bits preceding and succeeding the `lengthSizeMinusOne` field are re-defined. The syntax is as follows:

```
aligned(8) class SVCDecoderConfigurationRecord {
   unsigned int(8) configurationVersion = 1;
   unsigned int(8) AVCProfileIndication;
   unsigned int(8) profile_compatibility;
   unsigned int(8) AVCLevelIndication;
   bit(1) complete_represenation;
   bit(5) reserved = '11111'b;
   unsigned int(2) lengthSizeMinusOne;
   bit(1) reserved = '0'b;
   unsigned int(7) numOfSequenceParameterSets;
   for (i=0; i< numOfSequenceParameterSets; i++) {
      unsigned int(16) sequenceParameterSetLength ;
      bit(8*sequenceParameterSetLength) sequenceParameterSetNALUnit;
   }
   unsigned int(8) numOfPictureParameterSets;
   for (i=0; i< numOfPictureParameterSets; i++) {
      unsigned int(16) pictureParameterSetLength;
      bit(8*pictureParameterSetLength) pictureParameterSetNALUnit;
   }
}
```

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

The semantics of the fields `AVCProfileIndication`, `profile_compatibility`, and `AVCLevelIndication` differ from the `AVCDecoderConfigurationRecord` as follows:

The fields `AVCProfileIndication`, `AVCLevelIndication` carry the profile and level indications, respectively, indicating the profile and level of the entire scalable stream in this track. They, and the `profile_compatibility` field, must have values such that a conforming SVC decoder is able to decode bitstreams conforming to the profile, level and profile compatibility flags indicated in any of the sequence parameter sets or subset sequence parameter sets contained in this record.

The semantics of other fields are as follows, or are as defined for an `AVCDecoderConfigurationRecord`:

`complete_representation` is set on a minimal set of tracks that contain a portion of the original encoded scalable stream, as defined in A.6.1. Other tracks may be removed from the file without loss of any portion of the original encoded bitstream, and, once the set of tracks has been reduced to only those in the complete subset, any further removal of a track removes a portion of the encoded information.

`numOfSequenceParameterSets` indicates the number of SPSs and subset SPSs that are used for decoding the SVC elementary stream. The value of `numOfSequenceParameterSets` shall be in the range of 0 to 64, inclusive.

`SequenceParameterSetLength` indicates the length in bytes of the SPS or subset SPS NAL unit.

`SequenceParameterSetNALUnit` contains a SPS or subset SPS NAL unit. SPSs shall occur in order of ascending parameter set identifier with gaps being allowed. Subset SPSs shall occur in order of ascending parameter set identifier with gaps being allowed. Any SPS shall occur before all the subset SPSs, if any.

A.6 Derivation from the ISO base media file format

A.6.1 SVC track structure

A scalable video stream is represented by one or more video tracks in a file. Each track represents one or more operating points of the scalable stream. A scalable stream may, of course, be further thinned, if desired.

There is a minimal set of one or more tracks that, when taken together, contain the complete set of encoded information. All these tracks shall have the flag "`complete_representation`" set in all their sample entries. This group of tracks that form the complete encoded information are called the "complete subset".

Let the lowest operating point be the one of all the operating points represented by DTQ (dependency_id, temporal_id and quality_id) combinations that has the least values of dependency_id, temporal_id and quality_id, respectively. The track that has the flag "`complete_representation`" set and contains the lowest operating point shall be nominated as the 'scalable base track'. All the other tracks that are part of the same scalable encoded information shall be linked to this base track by means of a track reference of type '`sbas`' (scalable base). The complete encoded information can be retained when the tracks included in the "complete subset" are retained; all other tracks shall be extractions, subsets, copies or re-orderings of the complete subset.

NOTE 1 An alternate group may also include completely independent bitstreams, as well as alternative operating points of the same bitstream. The SVC tracks in the alternate group must be examined to see how many scalable base tracks are identified.

NOTE 2 "A scalable bitstream" may require more than one track to represent it (consider a stream with a low-resolution, low-frame-rate base layer, and a high resolution enhancement layer, and a high frame-rate enhancement layer, but missing the data for high resolution high frame-rate). However, such a scalable bitstream is typically a non-conforming bitstream.

All the tracks sharing the same scalable base track must share the same timescale.

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

A.6.2 Data sharing and extraction

Different tracks may logically share data. This sharing can take one of the following two forms:

a) The sample data is copied from one track into another track (and possibly compacted or re-interleaved with other data, such as audio). This creates larger overall files, but the low bit rate data may be compacted and/or interleaved with other material, for ease of extraction.
b) There may be instructions on how to perform this copy at the time that the file is read.

For the second case, Extractors (defined in B.3) are used.

A.6.3 SVC video stream definition

A.6.3.1 Sample description name and format

A.6.3.1.1 Definition

Types: 'avc2', 'avcC', 'svc1','svcC','seib'
Container: Sample Table Box ('stbl')
Mandatory: Either the avc1, or avc2 or svc1 box is mandatory.
Quantity: One or more sample entries may be present

If an SVC elementary stream contains a usable AVC compatible base layer, then an AVC visual sample entry ('avc1' or 'avc2') shall be used. Here, the entry shall contain initially an AVC Configuration Box, possibly followed by an SVC Configuration Box as defined below. The AVC Configuration Box documents the Profile, Level and Parameter Set information pertaining to the AVC compatible base layer as defined by the `AVCDecoderConfigurationRecord`. The SVC Configuration Box documents the Profile, Level and Parameter Set information pertaining to the entire stream containing the SVC compatible enhancement layers as defined by the `SVCDecoderConfigurationRecord`, stored in the SVCConfigurationBox.

For all sample entries except for 'svc1', i.e. 'avc1' and 'avc2', the width and height fields in the sample entry document the AVC base layer. For an 'svc1' sample entry, the width and height document the resolution achieved by decoding the entire stream.

If the SVC elementary stream does not contain a usable AVC base layer, then an SVC visual sample entry ('svc1') shall be used. The SVC visual sample entry shall contain an SVC Configuration Box, as defined below. This includes an `SVCDecoderConfigurationRecord`, as defined in this International Standard.

The `lengthSizeMinusOne` field in the SVC and AVC configurations in any given sample entry shall have the same value.

A priority assignment URI provides the name (in the URI space) of a method used to assign priority_id values. When it occurs in an AVC or SVC sample entry, exactly one URI shall be present, that documents the priority_id assignments in the stream. The URI is treated here as a name only; it should be de-referenceable, though this is not required. File readers may be able to recognize some methods and thereby know what stream extraction operations based on priority_id would do.

The sample entry name 'avc1' may only be used when the entire stream is a compliant and usable AVC stream as viewed by an AVC decoder operating under the configuration (including profile and level) given in the AVCConfigurationBox. The file format specific structures that resemble NAL units may be present but must not be used to access the AVC base data; that is, the AVC data must not be contained in Aggregators (though they may be included within the bytes referenced by the additional_bytes field) nor referenced by Extractors. The sample entry name 'avc2' indicates that, in order to form the intended AVC stream, Extractors must be replaced with the data they are referencing, and Aggregators must be examined for contained NAL Units. Extractors or aggregators may be used for SVC VCL NAL units in 'avc1', 'avc2' or 'svc1' tracks.

> NOTE When AVC compatibility is indicated, it may be necessary to indicate an unrealistic level for the AVC base layer, to accommodate the bit rate of the entire stream, because all the NAL units are considered as included in the AVC base layer and hence may be fed to the decoder, which is expected to discard those NAL unit it does not recognize. This case happens when the 'avc1' sample entry is used and both AVC and SVC configurations are present.

--`,,```,,,,````-`-`,,`,,`,`,,`---
© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Either or both of a ScalabilityInformationSEIBox or SVCConfigurationBox may be present in an 'avc1' sample entry. In this case the AVCSVCSampleEntry definition below applies.

`Compressorname` in the base class `VisualSampleEntry` indicates the name of the compressor used, with the value `"\012SVC Coding"` being recommended (\012 is 10, the length of the string "SVC coding" in bytes).

The following table shows for a video track all the possible uses of sample entries, configurations and the SVC tools (excluding timed metadata, which is always used in another track):

sample entry name	with configuration records	Meaning
'avc1'	AVC Configuration Only	A plain AVC track without SVC NAL units; Extractors, aggregators, and tier grouping shall not be present.
'avc1'	AVC and SVC Configurations	An SVC track with both AVC and SVC NAL units; Extractors and aggregators may be present; Extractors shall not reference AVC NAL units; Aggregators shall not contain but may reference AVC NAL units; Tier grouping may be present.
'avc2'	AVC Configuration Only	A plain AVC track without SVC NAL units; Extractors may be present and used to reference AVC NAL units; Aggregators may be present to contain and reference AVC NAL units; Tier grouping may be present.
'avc2'	AVC and SVC Configurations	An SVC track with both AVC NAL units and SVC NAL units; Extractors may be present and used to reference both AVC and SVC NAL units; Aggregators may be present to contain and reference both AVC and SVC NAL units; Tier grouping may be present.
'svc1'	SVC Configuration	An SVC track without AVC NAL units; Extractors may be present and used to reference both AVC and SVC NAL units; Aggregators may be present to contain and reference both AVC and SVC NAL units; Tier grouping may be present.

A.6.3.1.2 Syntax

```
class SVCConfigurationBox extends Box('svcC') {
   SVCDecoderConfigurationRecord() SVCConfig;
}

class ScalabilityInformationSEIBox extends Box('seib', size)
{
   unsigned int(8*size-64) scalinfosei;
}

class SVCPriorityAssignmentBox extends Box('svcP')
{
   unsigned int(8)   method_count;
   string PriorityAssignmentURI[method_count];
}


class AVCSVCSampleEntry() extends AVCSampleEntry (){
   SVCConfigurationBox  svcconfig;      // optional
   ScalabilityInformationSEIBox  scalability;  // optional
   SVCPriorityAssignmentBox   method;          // optional
}
```


© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

```
class AVC2SampleEntry() extends VisualSampleEntry ('avc2'){
   AVCConfigurationBox  avcconfig;
   SVCConfigurationBox  svcconfig;      // optional
   MPEG4BitRateBox bitrate;             // optional
   MPEG4ExtensionDescriptorsBox descr;  // optional
   ScalabilityInformationSEIBox  scalability;  // optional
   SVCPriorityAssignmentBox   method;          // optional
}

// Use this if the track is NOT AVC compatible
class SVCSampleEntry() extends VisualSampleEntry ('svc1'){
   SVCConfigurationBox     svcconfig;
   MPEG4BitRateBox bitrate;             // optional
   MPEG4ExtensionDescriptorsBox descr;  // optional
   ScalabilityInformationSEIBox  scalability;  // optional
   SVCPriorityAssignmentBox   method;          // optional
}
```

A.6.3.1.3 Semantics

`scalinfosei` contains an SEI NAL unit containing only a scalability information SEI message as specified in ISO/IEC 14496-10 Annex G. The 'size' field of the container box ScalabilityInformationSEIBox shall not be equal to 0 or 1.

`method_count` provides a count of the number of following URIs. This field must take the value 1 in an 'avc1', 'avc2' or 'svc1' sample entry.

`PriorityAssignmentURI` provides a unique name of the method used to assign prioritiy_id values. In the case of absence of this box, the priority assignment method is unknown.

A.6.4 SVC visual width and height

The visual width and height documented in a VisualSampleEntry of a stream containing SVC VCL NAL unit is the visual width and height of the AVC base layer, if the stream is described by a sample entry of type 'avc1' or 'avc2'; otherwise it is the visual width and height of decoded pictures by decoding the entire stream.

A.6.5 Sync sample (IDR)

For video data described by a sample entry of type 'avc1' or 'avc2', the sync sample table identifies IDR access units for both an AVC decoder, and an SVC decoder (if any) operating on the entire bitstream.

For video data described by a sample entry of type 'svc1', the sync sample table identifies IDR access units in the entire SVC bitstream.

> NOTE The sync sample table, if present, documents only access units that are IDR access units for both the AVC compatible base layer and the layer corresponding to decoding the entire bitstream contained in the track. In case documenting of layer-specific IDR access units is desired, the stream should be stored in separate tracks, e.g. two tracks, one containing the AVC base layer with a sample entry of type 'avc1', and the other containing the SVC enhancement layers with a sample entry of type 'svc1'. However, extractors must then be used for tracks that are not the scalable base track.

A.6.6 Shadow sync

A shadow sync box shall not be used for video data described by an 'svc1' sample entry. Its use for AVC is deprecated.

A.6.7 Independent and disposable samples box

If it is used in a track which is both AVC and SVC compatible, then care should be taken that the statements are true no matter what valid subset of the SVC data (possibly only the AVC data) is used. The 'unknown' values (value 0 of the fields sample-depends-on, sample-is-depended-on, and sample-has-redundancy) may be needed if the information varies.

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

A.6.8 Random access recovery points

For video data described by a sample entry of type ‘avc1’ or ‘avc2’, the random access recovery sample group identifies random access recovery points for both an AVC decoder, and an SVC decoder (if any) operating on the entire bitstream.

> NOTE If the random access recovery points for the AVC decoder and the SVC decoder operating on the entire bitstream are not all aligned, the random access recovery points table will not document all of them. In this case, the stream can be stored in multiple tracks, e.g. two tracks, one containing the AVC base layer with a sample entry of type ‘avc1’, and the other containing the SVC enhancement layers with a sample entry of type ‘svc1’.

For video data described by a sample entry of type ‘svc1’, the random access recovery sample group identifies random access recovery in the entire SVC bitstream.

A.6.9 Hinting

Care should be taken when the SVC structures (aggregators or extractors) are in use and the track is hinted. These structures are defined only for use in the file format and should not be transmitted. In particular, a hint track that points at an extractor in a video track would cause the extractor itself to be transmitted (which is probably both incorrect and not the desired behaviour), not the data the extractor references. Hint tracks should normally directly reference NAL units specified in ISO/IEC 14496-10.

A.6.10 Definition of a sub-sample for SVC

This subclause extends the definition of a sub-sample for AVC in subclause 5.3.16.

For the use of the sub-sample information box (subclause 8.42.1 of ISO/IEC 14496-12) in an SVC stream, a sub-sample is defined as one or more contiguous whole NAL units having the same values of the following fields: RefPicFlag, RedPicFlag, VclNalUnitFlag, IdrFlag, PriorityId, DependencyId, QualityId, TemporalId, UseRefBasePicFlag, DiscardableFlag and StoreBaseRepFlag, specified subsequently. Each sub-sample includes both NAL unit(s) and their preceding NAL unit length field(s). The presence of this box is optional; however, if present in a track containing SVC data, it shall have the semantics defined here.

As required in subclause 5.3.16, the subsample_priority field shall be set to a value in accordance with the specification of this field in ISO/IEC 14496-12.

The reserved field is defined for SVC as follows:

```
unsigned int(1) RefPicFlag;
unsigned int(1) RedPicFlag;
unsigned int(1) VclNalUnitFlag;
unsigned int(5) reserved = 0;
unsigned int(1) reserved = 0;
unsigned int(1) IdrFlag;
unsigned int(6) PriorityId;
unsigned int(1) reserved = 0; // corresponding to no_inter_layer_pred_flag
unsigned int(3) DependencyId;
unsigned int(4) QualityId;
unsigned int(3) TemporalId;
unsigned int(1) UseRefBasePicFlag;
unsigned int(1) DiscardableFlag;
unsigned int(1) reserved = 0; // corresponding to output_flag
unsigned int(1) StoreBaseRepFlag;
unsigned int(1) reserved = 0;
```

For an AVC VCL NAL unit in an SVC context, the prefix NAL unit shall be grouped with the AVC VCL NAL unit in the same sub-sample, and its fields values apply to the AVC VCL NAL unit.

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

`RefPicFlag` equal to 0 indicates that all the NAL units in the sub-sample have nal_ref_idc equal to 0. `RefPicFlag` equal to 1 indicates that all the NAL units in the sub-sample have nal_ref_idc greater than 0.

`RedPicFlag` equal to 0 indicates that all the NAL units in the sub-sample have redundant_pic_cnt equal to 0. `RedPicFlag` equal to 1 indicates that all the NAL units in the sub-sample have redundant_pic_cnt greater than 0.

`VclNalUnitFlag` equal to 0 indicates that all NAL units in the sub-sample are non-VCL NAL units. Value 1 indicates that all NAL units in the sub-sample are VCL NAL units.

`IdrFlag` indicates the idr_flag value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same value of idr_flag.

`PriorityId` indicates the priority_id value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same value of priority_id.

`NoInterLayerPredFlag` indicates the no_inter_layer_pred_flag of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same value of no_inter_layer_pred_flag.

`DependencyId` indicates the dependency_id value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same dependency_id value.

`QualityId` indicates the quality_id value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same quality_id value.

`TemporalId` indicates the temporal_id value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same temporal_id value.

`UseRefBasePicFlag` indicates the use_ref_base_pic_flag value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same value of use_ref_base_pic_flag.

`DiscardableFlag` indicates the discardable_flag value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same discardable_flag value.

NOTE that this is not the same definition as the discardable field in the sub-sample information box.

`StoreBaseRepFlag` indicates the store_base_rep_flag value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same value of store_base_rep_flag.

--`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Annex B
(normative)

SVC-file-format-specific in-stream structures

B.1 Introduction

Aggregators and Extractors are file format internal structures enabling efficient grouping of NAL units or extraction of NAL units from other tracks.

Aggregators and Extractors use the NAL unit structure. These structures are seen as NAL units in the context of the sample structure. While accessing a sample, Aggregators must be removed (leaving their contained or referenced NAL Units) and Extractors must be replaced by the data they reference. Aggregators and Extractors must not be present in a stream outside the file format.

These structures use NAL unit types reserved for the application/transport layer by ISO/IEC 14496-10.

NOTE The following is from ISO/IEC 14496-10:

"NOTE – NAL unit types 0 and 24..31 may be used as determined by the application. No decoding process for these values of nal_unit_type is specified in this Recommendation | International Standard."

B.2 Aggregators

B.2.1 Definition

This subclause describes Aggregators, which enable NALU-map-group entries to be consistent and repetitive. (See Annex C).

Aggregators are used to group NAL units belonging to the same sample. Aggregators use the same NAL unit header as SVC VCL NAL units of type 20 but with a different value of NAL unit type.

Aggregators can both aggregate, by *inclusion,* NAL units within them (within the size indicated by their length) and also aggregate, by *reference,* NAL units that follow them (within the area indicated by the additional_bytes field within them), When the stream is scanned by an AVC file reader, only the included NAL units are seen as "within" the aggregator; this permits, for example, an AVC file reader to skip a whole set of un-needed SVC VCL NAL units. Similarly if AVC NAL units are aggregated by reference, the AVC reader will not skip them and they remain in-stream for that reader.

When the aggregator is referenced by either an extractor with data_length equal to zero, or by a Map sample group, the aggregator is treated as aggregating both the included and referenced bytes.

An Aggregator may include or reference Extractors. An Extractor may extract from Aggregators. An aggregator must not include or reference another aggregator directly; however, an aggregator may include or reference an extractor which references an aggregator.

When scanning the stream:

a) if the aggregator is unrecognized (e.g. by an AVC reader or decoder) it is easily discarded with its included content;
b) if the aggregator is not needed (i.e. it belongs to an undesired layer) it and its contents both by inclusion and reference are easily discarded (using its length and additional_bytes fields);
c) if the aggregator is needed, its header is easily discarded and its contents retained.

An aggregator is stored within a sample like any other NAL unit.

All NAL units remain in decoding order within an aggregator.

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

B.2.2 Syntax

```
class aligned(8) Aggregator (AggregatorSize) {
   NALUnitHeader();
   unsigned int i = sizeof(NALUnitHeader());
   unsigned int((lengthSizeMinusOne+1)*8)
      additional_bytes;
   i += lengthSizeMinusOne+1;
   while (i<AggregatorSize) {
      unsigned int((lengthSizeMinusOne+1)*8)
               NALUnitLength;
      unsigned int(NALUnitLength*8) NALUnit;
      i += NALUnitLength+lengthSizeMinusOne+1;
   };
}
```

B.2.3 Semantics

The value of the variable AggregatorSize is equal to the size of the aggregator NAL unit, and the function sizeof(X) returns the size of the field X in bytes.

`NALUnitHeader()`: The NAL unit structure as specified in ISO/IEC 14496-10 Annex G for NAL units of type 20:

- `nal_unit_type` shall be set to the aggregator NAL unit type (type 30).
- forbidden_zero_bit, reserved_one_bit, and reserved_three_2bits shall be set as specified in ISO/IEC 14496-10 Annex G.
- Other fields (`nal_ref_idc`, `idr_flag`, `priority_id`, `no_inter_layer_pred_flag`, `dependency_id`, `quality_id`, `temporal_id`, `use_ref_base_pic_flag`, `discardable_flag`, and `output_flag`) shall be set as specified in B.4.

`additional_bytes`: The number of bytes following this aggregator NAL unit that should be considered as aggregated when this aggregator is referenced by an extractor with data_length equal to zero or Map sample group.

`NALUnitLength`: Specifies the size, in bytes, of the NAL unit following. The size of this field is specified with the `lengthSizeMinusOne` field.

`NALUnit`: a NAL unit as specified in ISO/IEC 14496-10, including the NAL unit header. The size of the NAL unit is specified by `NALUnitLength`.

B.3 Extractors

B.3.1 Definition

This subclause describes Extractors, which enable compact formation of tracks that extract, by reference, SVC data from other tracks.

An Aggregator may include or reference Extractors. An Extractor may reference Aggregators. When an extractor is processed by a file reader that requires it, the extractor is logically replaced by the bytes it references. Those bytes must not contain extractors; an extractor must not reference, directly or indirectly, another extractor.

NOTE The track that is referenced may contain extractors even though the data that is referenced by the extractor must not.

An extractor contains an instruction to extract data from another track, which is linked to the track in which the extractor resides, by means of a track reference of type '`scal`'.

--`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

The bytes copied shall be one of the following:

a) One entire NAL unit; note that when an Aggregator is referenced, both the included and referenced bytes are copied
b) More than one entire NAL unit

In both cases the bytes extracted start with a valid length field and a NAL unit header.

The bytes are copied only from the single identified sample in the track referenced through the indicated '`scal`' track reference. The alignment is on decoding time, i.e. using the time-to-sample table only, followed by a counted offset in sample number. Extractors are a media-level concept and hence apply to the destination track before any edit list is considered. (However, one would normally expect that the edit lists in the two tracks would be identical).

B.3.2 Syntax

```
class aligned(8) Extractor () {
   NALUnitHeader();
   unsigned int(8) track_ref_index;
   signed   int(8) sample_offset;
   unsigned int((lengthSizeMinusOne+1)*8)
      data_offset;
   unsigned int((lengthSizeMinusOne+1)*8)
      data_length;
}
```

B.3.3 Semantics

NALUnitHeader: The NAL unit structure as specified in ISO/IEC 14496-10 Annex G for NAL units of type 20:

nal_unit_type shall be set to the extractor NAL unit type (type 31).

forbidden_zero_bit, reserved_one_bit, and reserved_three_2bits shall be set as specified in ISO/IEC 14496-10 Annex G.

Other fields (nal_ref_idc, idr_flag, priority_id, no_inter_layer_pred_flag, dependency_id, quality_id, temporal_id, use_ref_base_pic_flag, discardable_flag, and output_flag) shall be set as specified in B.4.

track_ref_index specifies the index of the track reference of type 'scal' to use to find the track from which to extract data. The sample in that track from which data is extracted is temporally aligned or nearest preceding in the media decoding timeline, i.e. using the time-to-sample table only, adjusted by an offset specified by sample_offset with the sample containing the Extractor. The first track reference has the index value 1; the value 0 is reserved.

sample_offset gives the relative index of the sample in the linked track that shall be used as the source of information. Sample 0 (zero) is the sample with the same, or the closest preceding, decoding time compared to the decoding time of the sample containing the extractor; sample 1 (one) is the next sample, sample -1 (minus 1) is the previous sample, and so on.

data_offset: The offset of the first byte within the reference sample to copy. If the extraction starts with the first byte of data in that sample, the offset takes the value 0. The offset shall reference the beginning of a NAL unit length field.

data_length: The number of bytes to copy. If this field takes the value 0, then the entire single referenced NAL unit is copied (i.e. the length to copy is taken from the length field referenced by the data offset, augmented by the additional_bytes field in the case of Aggregators).

NOTE If the two tracks use different `lengthSizeMinusOne` values, then the extracted data will need re-formatting to conform to the destination track's length field size.

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

B.4 NAL unit header values

Both extractors and aggregators use NAL unit headers with the NAL unit header SVC extension. The NAL units extracted by an extractor or aggregated by an aggregator are all those NAL units that are referenced or included by recursively inspecting the contents of aggregator or extractor NAL units. The fields nal_ref_idc, idr_flag, priority_id, no_inter_layer_pred_flag, dependency_id, quality_id, temporal_id, use_ref_base_pic_flag, discardable_flag, and output_flag shall take the following values:

Aggregators can be used to group AVC base layer NAL units. If these Aggregators are used in an 'avc1' track then an aggregator shall not use inclusion but reference of AVC base layer NAL units (the length of the Aggregator includes only its header and the NAL units referenced by the Aggregator are specified by `additional_bytes`).

`nal_ref_idc` shall be set to the highest values of the fields, respectively, in all the extracted or aggregated NAL units.

`idr_flag` shall be set to the highest values of the fields, respectively, in all the extracted or aggregated NAL units.

`priority_id`, `temporal_id`, `dependency_id`, and `quality_id` shall be set to the lowest values of the fields, respectively, in all the extracted or aggregated NAL units.

`discardable_flag` shall be set to 1 if and only if all the extracted or aggregated NAL units have the `discardable_flag` set to 1, and set to 0 otherwise.

`output_flag` should be set to 1 if at least one of the aggregated or extracted NAL units has this flag set to 1, and otherwise set to 0.

`use_ref_base_pic_flag` shall be set to 1 if and only if at least one of the extracted or aggregated VCL NAL units have the `use_ref_base_pic_flag` set to 1, and set to 0 otherwise.

`no_inter_layer_pred_flag` shall be set to 1 if and only if all the extracted or aggregated VCL NAL units have the `no_inter_layer_pred_flag` set to 1, and set to 0 otherwise.
If the set of extracted or aggregated NAL units is empty, then each of these fields takes a value conformant with the mapped tier description.

NOTE 1 Aggregators could group NAL units with different scalability information.

NOTE 2 Aggregators could be used to group NAL units belonging to a level of scalability which may not be signalled by the NAL unit header SVC extension (e.g. NAL units belonging to a region of interest). The description of such Aggregators may be done with the tier description and the NAL unit map groups. In this case more than one Aggregator with the same scalability information may occur in one sample.

NOTE 3 If multiple scalable tracks reference the same media data, then an aggregator should group NAL units with identical scalability information only. This ensures that the resulting pattern can be accessed by each of the tracks.

NOTE 4 If no NAL unit of a particular layer exist in an access unit then an empty Aggregator (in which the length of the Aggregator includes only the header, and additional_bytes is zero) may exist.

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Annex C
(normative)

SVC sample group definitions

C.1 Introduction

The following sample groups may be used in an SVC track to document the structure of the SVC stream and to ease obtaining information of subsets of the stream and extraction of any of the subsets.

There are a number of boxes, defined below, which may occur in the sample group description, namely the Scalable Group Entry..

Each Scalable Group Entry documents a subset of the SVC stream. Each of the subsets is associated with a tier and may contain one or more operating points. These entries are defined using a grouping type of '`scif`'.

> NOTE For each tier, there may be more than one ScalableGroupEntry in the SampleGroupDescriptionBox of grouping type '`scif`'. Only one of those entries is the primary definition of the tier.

Though the Scalable Group Entries are contained in the SampleGroupDescription box, the grouping is not a true *sample* grouping as each sample may be associated with more than one scalable group, as these groups are used to describe *sections* of the samples – the NAL units. As a result, it is possible that there may not be a SampleToGroup box of the grouping type '`scif`', unless it happens that a group does, in fact, describe a whole sample. Even if a SampleToGroup box of the grouping type '`scif`' is present, the information is not needed for extraction of NAL units of tiers; instead, the map groups must always document the 'pattern' of NAL units within the samples and provide the NAL-unit-to-tier mapping information that may be needed for extraction of NAL units.

C.2 Definition

C.2.1 Tier information box

C.2.1.1 Definition

Box Types: 'tiri'
Container: ScalableGroupEntry
Mandatory: Yes
Quantity: Zero or One // depends on primary_definition

The tier information box provides information about the profile, level, frame size, discardability, and frame-rate of a tier.

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

C.2.1.2 Syntax

```
class TierInfoBox extends Box('tiri'){ //Mandatory Box
   unsigned int(16) tierID;
   unsigned int(8) profileIndication;
   unsigned int(8) profile_compatibility;
   unsigned int(8) levelIndication;
   unsigned int(8) reserved = 0;

   unsigned int(16) visualWidth;
   unsigned int(16) visualHeight;

   unsigned int(2) discardable;
   unsigned int(2) constantFrameRate;
   unsigned int(4) reserved = 0;
   unsigned int(16) frameRate;
}
```

C.2.1.3 Semantics

`tierID` gives the identifier of the tier.

`profileIndication` contains the profile_idc as defined in ISO/IEC 14496-10, when the parameter applies to the bitstream subset consisting of the tier and all the dependent tiers.

`profile_compatibility` is a byte defined exactly the same as the byte which occurs between the profile_idc and level_idc in a sequence parameter set or a subset sequence parameter set, as defined in ISO/IEC 14496-10 Annex G, when the parameters apply to the bitstream subset consisting of the tier and all the dependent tiers.

`levelIndication` contains the level_idc as defined in ISO/IEC 14496-10, when the parameter applies to the bitstream subset consisting of the tier and all the dependent tiers.

The profile, profile compatibility flags and level indicated by the fields `profileIndication`, `profile_compatibility`, and `levelIndication` specifies an interoperability point with which the bitstream obtained from the particular tier and all the dependent tiers is compatible.

`visualWidth` gives the value of the width of the coded picture or coded sub-picture in luma pixels of the representation of this tier in the SVC stream. A coded sub-picture consists of a proper subset of coded slices of a coded picture. A tier may consist of only sub-pictures. In this case, the tier is referred to as a sub-picture tier. A sub-picture tier may represent a region-of-interest part of the region represented by the entire stream.

NOTE The tier representation of a sub-picture tier might not be a valid SVC stream. One example is as follows. An AVC bitstream is encoded using two slice groups. The first slice group includes the macroblocks representing a region-of-interest and is coded without referring to slices in the other slice group for inter prediction over all the access units. The slices of the first slice group in each access unit then form a sub-picture and a sub-picture tier can be specified to include all the sub-pictures over all the access units.

`visualHeight` gives the value of the height of the coded picture or coded sub-pictures in luma pixels of the representation this tier in the SVC stream.

`discardable` takes one of the following values; the value 02 is reserved.

00 this tier does not contain NAL units with discardable_flag equal to 1.

01 this tier contains both NAL units with discardable_flag equal to 1 and discardable_flag equal to 0.

03 all NAL units in this tier are with discardable_flag equal to 1.

`constantFrameRate` specifies if the frame rate of this tier is constant. A value of 0 denotes a non-constant frame rate, a value of 1 denotes a constant frame rate and a value of 2 denotes that it is not clear whether the frame rate is constant. A value of 3 is reserved.

--`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

`frameRate` gives the frame rate when the bitstream corresponding to this tier and all the lower tiers that this tier depends on is decoded. If `constantFrameRate` has a value of 0 or 2 then `frameRate` gives the average frame rate. If `constantFrameRate` has a value of 1 then `frameRate` gives the constant frame rate. `frameRate` equal to 0 indicates an unspecified frame rate.

C.2.2 Tier bit rate box

C.2.2.1 Definition

Box Type: 'tibr'
Container: ScalableGroupEntry
Mandatory: No
Quantity: Zero or One

The tier bit rate box provides information about the bit rate values of a tier. Two sets of information are provided: for the tier representation, including all the tiers on which the current tier depends, and for the tier alone. Similarly, for each set of information, three values are supplied:

a) the lowest long-term average bit rate that this tier could deliver. Let maxDid be the greatest dependency_id for all NAL units of the tier, and minQid be the least quality_id for all the NAL units of the tier and having dependency_id equal to maxDid. The following NAL units of this tier are not considered in calculating this bit rate value: those having dependency_id equal to maxDid and quality_id greater than minQid.

b) the long-term average bit rate of the tier; all NAL units of the tier are considered.

c) the maximum, or peak, bit rate of the tier; all NAL units of the tier are considered.

C.2.2.2 Syntax

```
class TierBitRateBox extends Box('tibr'){
   unsigned int(32) baseBitRate;
   unsigned int(32) maxBitRate;
   unsigned int(32) avgBitRate;

   unsigned int(32) tierBaseBitRate;
   unsigned int(32) tierMaxBitRate;
   unsigned int(32) tierAvgBitRate;
}
```

C.2.2.3 Semantics

`baseBitRate` gives the lowest long-term average bit rate in bits/second of the stream made from this tier and the dependent lower tiers over the entire stream. Let maxDid be the greatest dependency_id for all NAL units of the tier, and minQid be the least quality_id for all NAL units of the tier and having dependency_id equal to maxDid. The NAL units that are taken into account when calculating this bit rate value are as follows: 1) all NAL units of the tier except for those having dependency_id equal to maxDid and quality_id greater than minQid; 2) all NAL units of the lower tiers the current tier depends on.

`maxBitRate` gives the maximum bit rate in bits/second of the stream containing all NAL unit mapped to this tier and the dependent lower tiers, over any window of one second. All NAL units in this tier and the lower tiers this tier depends on are taken into account.

`avgBitRate` gives the long-term average bit rate in bits/second of the stream containing all NAL unit mapped to this tier and the dependent lower tiers, averaged over the entire stream. All NAL units in this tier and the lower tiers this tier depends on are taken into account.

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

`tierBaseBitRate` gives the lowest long-term average bit rate in bits/second of the stream made from only this tier over the entire stream. The set of NAL units that are taken into account when calculating this bit rate value is the same as for baseBitRate but excluding all NAL units of the dependent lower tiers.

`tierMaxBitRate` gives the maximum bit rate in bits/second that is provided by only this tier over any window of one second. All NAL units mapped to this tier are taken into account. All NAL units of the dependent lower tiers are not considered.

`tierAvgBitRate` - gives the long-term average bit rate in bits/second that is provided by only this tier, averaged over the entire stream. All NAL units mapped to this tier are taken into account. All NAL units of the dependent lower tiers are not considered.

C.2.3 SVC priority range

C.2.3.1 Definition

Box Types: 'svpr'
Container: ScalableGroupEntry
Mandatory: Yes
Quantity: Exactly One

This box reports the minimum and maximum priority_id of the NAL units mapped to this tier.

C.2.3.2 Syntax

```
class SVCPriorityRangeBox extends Box('svpr') {
   unsigned int(2) reserved1 = 0;
   unsigned int(6) min_priorityId;
   unsigned int(2) reserved2 = 0;
   unsigned int(6) max_priorityId;
}
```

C.2.3.3 Semantics

`min_priority_id, min_priority_id` take the minimum or maximum value of the priority_id syntax element that is present in the scalable extension NAL unit defined in the SVC video specification of the NAL units mapped to the tier. For AVC streams this takes the value that is, or would be, in the prefix NAL unit.

C.2.4 SVC dependency range

C.2.4.1 Definition

Box Types: 'svdr'
Container: ScalableGroupEntry
Mandatory: Yes
Quantity: Exactly One

This box reports the minimum and maximum dependency_id of the NAL units mapped to this tier.

The field min_temporal_id reports the minimum value of temporal_id of the NAL units in the tier having dependency_id equal to min_dependency_id, Similarly the field min_quality_id reports the minimum quality_id of those NAL units. The fields max_temporal_id and max_quality_id similarly report on the maximum values of the respective fields in those NAL units having dependency_id equal to max_dependency_id.

--`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

C.2.4.2 Syntax

```
class SVCDependencyRangeBox extends Box('svop') {
   unsigned int(3) min_dependency_id;
   unsigned int(3) min_temporal_id;
   unsigned int(6) reserved = 0;
   unsigned int(4) min_quality_id;
   unsigned int(3) max_dependency_id;
   unsigned int(3) max_temporal_id;
   unsigned int(6) reserved = 0;
   unsigned int(4) max_quality_id;
}
```

C.2.4.3 Semantics

`min_dependency_id, max_dependency_id` take the minimum or maximum value of the `dependency_id` syntax element that is present in the scalable extension NAL unit header defined in the SVC video specification of the NAL units mapped to the tier. For AVC streams this takes the value that is, or would be, in the prefix NAL unit (note: this value is zero).

`min_temporal_id, max_temporal_id` take the minimum or value of the temporal_id syntax element that is present in the scalable extension NAL unit header defined in the SVC video specification of the NAL units mapped to the tier having dependency_id equal to `min_dependency_id` and `max_dependency_id` respectively . For AVC streams this takes the value that is, or would be, in the prefix NAL unit.

`min_quality_id, max_quality_id` take the minimum or value of the quality_id syntax element that is present in the scalable extension NAL unit header defined in the SVC video specification of the NAL units mapped to the tier having dependency_id equal to `min_dependency_id` and `max_dependency_id` respectively . For AVC streams this takes the value that is, or would be, in the prefix NAL unit.

C.2.5 SVC initial parameter sets box

C.2.5.1 Definition

Box Type: 'svip'
Container: ScalableGroupEntry
Mandatory: No
Quantity: Zero or One

The SVC initial parameter sets box documents which parameter sets are needed for decoding this tier and all the lower tiers it depends on.

C.2.5.2 Syntax

```
class InitialParameterSetBox extends Box ('svip') {
   unsigned int(8) sps_id_count;
   for (i=0; i< sps_id_count; i++)
      unsigned int(8) SPS_index;
   unsigned int(8) pps_id_count;
   for (i=0; i< pps_id_count; i++)
      unsigned int(8) PPS_index;
}
```


© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

C.2.5.3 Semantics

`sps_id_count, pps_id_count` gives the number of entries in the following tables.

`SPS_index` specifies that the SPS or subset SPS with this index is needed for decoding this tier and all the lower tiers it depends on. These are 1-based indices into the arrays in the SVCDecoderConfigurationRecord.

`PPS_index` specifies that the PPS with this index is needed for decoding this tier and all the lower tiers it depends on. These are 1-based indices into the arrays in the SVCDecoderConfigurationRecord.

C.2.6 SVC rect region box

C.2.6.1 Definition

Box Type: 'rrgn'
Container: ScalableGroupEntry
Mandatory: No
Quantity: Zero or One

The SVC rect region box documents the geometry information of the region represented by the current tier relative to the region represented by another tier. When extended spatial scalability was used to encode in the current tier a cropped region of another tier, then the geometry information of the cropped region can be signaled by this box. This box can also be used to signal the geometry information of a region-of-interest (ROI) when the current tier is a sub-picture tier. This area can either be static for all samples or vary at sample-by-sample basis. Note that it is possible that independent sub-pictures do not depend on all the tiers with lower tierID. In this case dependencies can be given with the tier dependency box.

C.2.6.2 Syntax

```
class RectRegionBox extends Box('rrgn'){
   unsigned int(16) base_region_tierID;
   unsigned int(1) dynamic_rect;
   unsigned int(7) reserved = 0;
   if(dynamic_rect == 0) {
      unsigned int(16) horizontal_offset;
      unsigned int(16) vertical_offset;
      unsigned int(16) region_width;
      unsigned int(16) region_height;
   }
}
```

C.2.6.3 Semantics

`base_region_tierID` gives the tierId value of the tier wherein the represented region is used as the base region for derivation of the region represented by the current tier.

`dynamic_rect` equal to 1 indicates that the region represented by the current tier is a dynamically changing rectangular part of the base region. Otherwise the region represented by the current tier is a fixed rectangular part of the base region.

`horizontal_offset` and `vertical_offset` give respectively the horizontal and vertical offsets of the top-left pixel of the rectangular region represented by the tier, in relative to the top-left pixel of the base region, in luma samples of the base region.

`region_width` and `region_height` give respectively the width and height of the rectangular region represented by the tier, in luma samples of the base region.

--`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

C.2.7 SVC buffering information box

C.2.7.1 Definition

Box Type: 'buff'
Container: ScalableGroupEntry
Mandatory: No
Quantity: Zero or One

The BufferingBox contains the buffer information of the tier.

C.2.7.2 Syntax

```
class BufferingBox extends Box('buff'){
   unsigned int(16)     operating_point_count
   for (i = 0; i < operating_point_count; i++){
      unsigned int (32)    byte_rate
      unsigned int (32)    cpb_size
      unsigned int (32)    dpb_size
      unsigned int (32)    init_cpb_delay
      unsigned int (32)    init_dpb_delay
   }
}
```

C.2.7.3 Semantics

`operating_point_count` specifies the number of HRD operating points for the bitstream required for decoding the current tier. Values of the HRD parameters are specified separately for each operating point. The value of operating_point_count shall be greater than 0.

`byte_rate` specifies the input byte rate (in bytes per second) to the coded picture buffer (CPB) of the HRD. The bitstream is constrained by the value of BitRate equal to byte_rate * 8 for NAL HRD parameters as specified in SVC. For VCL HRD parameters, the value of BitRate is equal to byte_rate * 40 / 6. The value of byte_rate shall be greater than 0.

`cpb_size` specifies the required size of the coded picture buffer in bytes. The bitstream is constrained by the value of CpbSize equal to cpb_size * 8 for NAL HRD parameters as specified in SVC. For VCL HRD parameters, the value of CpbSize is equal to cpb_size * 40 / 6.

At least one pair of values of byte_rate and cpb_size of the same operating point shall conform to the maximum bit rate and CPB size allowed by profile and level of the bitstream required for decoding the current tier.

`dpb_size` specifies the required size of the decoded picture buffer (DPB), in unit of bytes. The bitstream is constrained by the value of max_dec_frame_buffering equal to Min(16, Floor(dpb_size) / (PicWidthMbs * FrameHeightInMbs * 256 * ChromaFormatFactor))) as specified in SVC.

`init_cpb_delay` specifies the required delay between the time of arrival in the CPB of the first bit of the first access unit and the time of removal from the CPB of the first access unit. It is in units of a 90 kHz clock. The bitstream is constrained by the value of the nominal removal time of the first access unit from the CPB, $t_{r,n}(0)$, equal to init_cpb_delay as specified in SVC.

`init_dpb_delay` specifies the required delay between the time of arrival in the DPB of the first decoded picture and the time of output from the DPB of the first decoded picture. It is in units of a 90 kHz clock. The bitstream is constrained by the value of dpb_output_delay for the first decoded picture in output order equal to init_dpb_delay as specified in SVC assuming that the clock tick variable, t_c, is equal to 1 / 90 000.

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

C.2.8 SVC tier dependency box

C.2.8.1 Definition

Box Type: 'ldep'
Container: ScalableGroupEntry
Mandatory: No
Quantity: Zero or One

The TierDependencyBox identifies the tiers that the current tier is dependent on.

C.2.8.2 Syntax

```
class TierDependencyBox extends Box('ldep'){
   unsigned int(16) entry_count;
   for (i=0; i < entry_count; i++)
      unsigned int(16) dependencyTierId;
}
```

C.2.8.3 Semantics

`dependencyTierId` gives the tierId of a tier on which the current tier is directly or indirectly dependent. Tier A is directly dependent on tier B if there is at least one NAL unit in tier A using inter prediction or inter-layer prediction from tier B. Tier A is indirectly dependent on tier B if tier A is not directly dependent on tier B while decoding of tier A requires the presence of tier B. The value of dependencyTierId shall be smaller than the tierId of the current tier. The decoding of the current tier requires the presence of the tier indicated by dependencyTierId. All dependencies up to the tier with the lowest tierId shall be given with the TierDependencyBox.

C.2.9 SVC region of interest box

C.2.9.1 Definition

Box Type: 'iroi'
Container: ScalableGroupEntry
Mandatory: No
Quantity: Zero or One

This box provides the geometry information of region-of-interest (ROI) divisions of the current tier, when the current tier is encoded to multiple (typically a large number of) independent rectangular ROIs.

NOTE This box is typically used for interactive ROI use cases, where the server can interactively transmit only the NAL units belonging to the ROIs requested by a client.

The assignment of NAL units to a ROI is done in a time parallel metadata track as specified in Annex D.

A ROI ID, denoted as roi_id, is specified for each ROI in a tier. If iroi_type is equal to 0, roi_id is equal to the index of a ROI in a ROI raster scan (see ISO/IEC 14496-10 for the definition of "raster scan" and the use of "macroblock raster scan") of the region represented by the tier starting with zero for the top-left ROI in the region. If iroi_type is equal to 1, roi_id is equal to the entry index i in the syntax of IroiInfoBox(). If iroi_type is equal to 2, roi_id is set to a number identifying the region of interest. In this case, the temporal metadata must contain statements mapping NAL units to roi_ids.

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

C.2.9.2 Syntax

```
class IroiInfoBox extends Box('iroi'){
   unsigned int(2) iroi_type;
   unsigned int(6) reserved = 0;
   if(iroi_type == 0) {
      unsigned int(8) grid_roi_mb_width;
      unsigned int(8) grid_roi_mb_height;
   }
   else if(iroi_type == 1){
      unsigned int(24) num_roi;
      for(int i=1; i<= num_roi; i++) {
         unsigned int(32) top_left_mb;
         unsigned int(8) roi_mb_width;
         unsigned int(8) roi_mb_height;
      }
   }
}
```

C.2.9.3 Semantics

`iroi_type` indicates the types of region division for all the ROIs. The value 0 indicates that all the ROIs (except possibly the right-most ones and the bottom-most ones) are of identical width and height. The value 1 indicates that the geometry information for each ROI is separately signalled. The value 2 indicates that the geometry can not be given. Values greater than 2 are reserved.

`grid_roi_mb_width` and `grid_roi_mb_height` indicate the width and height, respectively, in units of macroblocks, of the ROIs. All the ROIs have identical width and height, with the following exceptions.

When (`PicWidthInMbs % grid_roi_mb_width`) is not equal to 0, the right-most ROIs have a width equal to `( PicWidthInMbs % grid_roi_mb_width )` macroblocks.

When (`PicHeightInMbs % grid_roi_mb_height`) is not equal to 0, the bottom-most ROIs have a height equal to (`PicHeightInMbs % grid_roi_mb_height`) macroblocks.

Where `PicWidthInMbs` and `PicHeightInMbs` are the visual width and height of a decoded picture of the tier representation in units of macroblocks, respectively, as specified in ISO/IEC 14496-10, (x % y) returns the remainder of x divided by y.

`num_roi` indicates the number of ROIs in a coded picture of the tier representation.

`top_left_mb` specifies the macroblock address of the first macroblock in raster scan order in the ROI of the current entry. The value of `top_left_mb` shall be equal to the syntax element first_mb_in_slice of the coded slices that belong to the current tier and that cover the top-left macroblock of the ROI of the current entry.

`roi_mb_width` and `roi_mb_height` indicate the width and height, respectively, in unit of macroblocks, of the ROI of the current entry.

C.2.10 SVC lightweight transcoding Box

C.2.10.1 Definition

The presence of the box indicates that the bitstream represented by this tier (and tiers it depends upon) can be transcoded from an SVC stream to an AVC stream as indicated, and that the transcoded bitstream can be given the indicated profile and level indicators, with the indicated bit rates. The information on the resulting profile, level, and bit rate may be given for either of the entropy coding systems, or both.

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

C.2.10.2 Syntax

```
class TranscodingInfoBox extends Box('tran'){
   unsigned int(4) reserved = 0;
   unsigned int(2) conversion_idc;
   unsigned int(1) cavlc_info_present_flag;
   unsigned int(1) cabac_info_present_flag;
   if(cavlc_info_present_flag){
      unsigned int(24) cavlc_profile_level_idc;
      unsigned int(32) cavlc_max_bitrate;
      unsigned int(32) cavlc_avg_bitrate;
   }
   if(cabac_info_present_flag){
      unsigned int(24) cabac_profile_level_idc;
      unsigned int(32) cabac_max_bitrate;
      unsigned int(32) cabac_avg_bitrate;
   }
}
```

C.2.10.3 Semantics

`conversion_idc` equal to 0, 1, or 2 indicates that the representation of the current tier can be translated to an AVC bit-stream as specified in the semantics of the scalability information SEI message in ISO/IEC 14496-10 Annex G. `conversion_idc` equal to 3 is reserved.

`cavlc_info_present_flag` specifies whether the transcoding information of the translated bitstream using the Context-based Adaptive Variable Length Coding (CAVLC) entropy coder (i.e. when the syntax element entropy_coding_mode_flag in the translated bitstream is equal to 0) as specified in ISO/IEC 14496-10 Annex G is present.

`cabac_info_present_flag` specifies whether the transcoding information of the translated bitstream using the Context-based Adaptive Binary Arithmetic Coding (CABAC) entropy coder (i.e. when entropy_coding_mode_flag in the translated bitstream is equal to 1) as specified in ISO/IEC 14496-10 Annex G is present.

`cavlc_profile_level_idc` is the exact copy of the three bytes comprised of profile_idc, constraint_set0_flag, constraint_set1_flag, constraint_set2_flag, constraint_set3_flag and level_idc, if these syntax elements were used to specify the profile and level compliancy of the transcoded bitstream using the CAVLC entropy coder.

`cavlc_max_bitrate` specifies the maximum bit rate in bits/second (units of 1000 bits/sec), over any window of one second, that is provided by the transcoded bitstream using the CAVLC entropy coder.

`cavlc_avg_bitrate` specifies the average bit rate in bits/second (units of 1000 bits/sec) that is provided by the transcoded bitstream using the CAVLC entropy coder.

`cabac_profile_level_idc` is the exact copy of the three bytes comprised of profile_idc, constraint_set0_flag, constraint_set1_flag, constraint_set2_flag, constraint_set3_flag and level_idc, if these syntax elements were used to specify the profile and level compliancy of the transcoded bitstream using the CABAC entropy coder.

`cabac_max_bitrate` specifies the maximum bit rate in bits/second (units of 1000 bits/sec), over any window of one second, that is provided by the transcoded bitstream using the CABAC entropy coder.

`cabac_avg_bitrate` specifies the average bit rate in bits/second (units of 1000 bits/sec) that is provided by the transcoded bitstream using the CABAC entropy coder.

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

C.2.11 SVC scalable group entry

C.2.11.1 Definition

Group Type: 'scif'
Container: Sample Group Description Box ('sgpd')
Mandatory: No
Quantity: Zero or More

Each scalable group entry is associated with a groupID and a tierID. The tierID entries are ordered in terms of their dependency signalled by the value of tierID. A larger value of tierID indicates a higher tier. A value 0 indicates the lowest tier. Decoding of a tier is independent of any higher tier but may be dependent on lower tier. Therefore, the lowest tier can be decoded independently, decoding of tier 1 may be dependent on tier 0, decoding of tier 2 may be dependent on tiers 0 and 1, and so on. A tier can include data from one or more layers in the video stream.

If two tiers are mutually independent, then it is required that the tier that has the greater importance, in the view of the content creator, shall be the lower tier (i.e. have the smaller tierID).

> NOTE For example, two tiers are mutually independent (though there may be some lower tiers that they both depend on). The first tier, if presented, has higher frame rate but lower individual picture quality, while the second tier, if presented, has lower frame rate but higher individual picture quality. If the file composer can identify that the first tier offers a better user experience for this content than the second tier, then the first tier is assigned a lower tierID value than the second tier.

There shall be exactly one primary definition for each tier. For each ScalableGroupEntry, when the field primary_groupID is equal to the field groupID, the group is the primary definition of this tier, and the following applies.

- A TierInfoBox, SVCDependencyRangeBox and SVCPriorityRangeBox shall be present.
- For a certain tier, if any of the optional boxes is not present, then that information is not defined for that tier (there is no inheritance of tier information). If for a certain tier no TierDependencyBox is present then this tier may depend on all tiers with lower tierID.
- If the `InitialParameterSetBox` is present then the parameter sets needed for decoding this tier and all the lower tiers it depends on are indicated with this box. If this box is not present then it is not signalled whether all the parameter sets given by the `SVCDecoderConfigurationRecord` are needed. If parameter set streams are used, then the InitialParameterSetBox shall not be present.
- The values of tierIDs are not required to be contiguous.

For each specified tierID, there shall be at least one NAL unit that is associated with it. In other words, it is disallowed to specify tiers that are not used in the track.

Each NAL unit in the elementary stream is associated with a tierID value as follows. First, each sample is associated with a map of groupID values through the sample grouping of type "`scnm`" as specified subsequently. The "`scnm`" sample grouping therefore indicates the association between NAL units and groupID values within each sample. Values of groupID can then be associated with values of tierID using the sample group description box of type "`scif`". NAL units associated with a particular tierID value may require all or some of the NAL units associated with all the smaller tierID values for proper decoding operation, but will never require any NAL unit associated with a greater tierID value. (i.e., dependency will only exist in the direction of lower tiers).

A Server or Player can choose a subset of tierID values that will be needed for proper decoding operation based on the values of the description fields present within the entries (e.g., frame rate, etc) of the sample group description box of type "`scif`".

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Since the ScalableGroupEntry is of variable length and has no internal length field, the SampleGroupDescription Box which contains it must carry length information for its entries according to version 1 of the SampleGroupDescription box definition.

The scalable video coding data in a particular tier may be protected; this is indicated by the presence of a ProtectionSchemeInfoBox in the tier definition. If any tier is so protected then:

- If the base layer (AVC) is protected, then the sample entry *must* also be transformed by changing its four-character code, and adding a ProtectionSchemeInfoBox, in the standard way.
- If *any* layer is protected in a track, a ProtectionSchemeInfoBox of some kind must be added to the sample entry (this is the 'warning' that some protection is in effect). The original format box in the ProtectionSchemeInfoBox is required but may not be needed as the four-character code in the SampleEntry might not have changed if, for example, the base layer is un-protected.
- Extractors may point to data in protected streams; the byte references are to data 'on disc' (i.e. possibly protected). When protecting, if extractors are permitted by the scheme in use, and the protection changes data sizes, then extractors may need re-writing.

C.2.11.2 Syntax

```
class ScalableGroupEntry() extends VisualSampleGroupEntry ('scif') {
   unsigned int(8) groupID;
   unsigned int(8) primary_groupID;
   unsigned int(1) is_tier_IDR;
   unsigned int(1) noInterLayerPredFlag;
   unsigned int(1) useRefBasePicFlag;
   unsigned int(1) storeBaseRepFlag;
   unsigned int(1) is_tl_switching_point;
   unsigned int(3) reserved = 0;
   unsigned int(8) tl_switching_distance;

   if (groupID == primary_groupID)  // primary definition of tier
   {
      TierInfoBox();              // Mandatory
      SVCDependencyRangeBox();    // Mandatory
      SVCPriorityRangeBox();      // Mandatory

      //Optional Boxes or fields may follow when defined  later
      TierBitRateBox();              // optional
      RectRegionBox();               // optional
      BufferingBox();                // optional
      TierDependencyBox();           // optional
      InitialParameterSetBox();      // optional
      IroiInfoBox();                 // optional
      ProtectionSchemeInfoBox();     // optional
      TranscodingInfoBox();          // optional
   }
}
```

C.2.11.3 Semantics

`groupID` gives the identifier of the group entry. groupIDs are arbitrary values but shall be unique.

`primary_groupID` specifies the group containing the primary definition of this tier. If this value is equal to the value of groupID then this group is the primary definition of this tier.

`is_tier_IDR` when set to 1, indicates that, for the members of this group, the coded pictures of the representation of the highest layer (i.e. the layer with the highest value of dependency_id as specified in ISO/IEC 14496-10 Annex G) are IDR pictures. A value of 0 indicates that, for the members of this group, the coded pictures of the representation of the highest layer are not IDR pictures.

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

`noInterLayerPredFlag` when set to 1, indicates that the members of this group are with no_inter_layer_pred_flag equal to 1 and coded without using inter layer prediction. A value of 0 indicates that the members of this group may have been coded using inter layer prediction.

`useRefBasePicFlag` when set to 1 indicates that the members of this group are with use_ref_base_pic_flag equal to 1 and using decoded base representations for inter prediction such that mismatch due to discarding of NAL units with quality_id greater than 0 is controlled. A value of 0 indicates that the members of this group may have any value of use_ref_base_pic_flag.

`storeBaseRepFlag` when set to 1 indicates that the members of this group are with `store_base_rep_flag` equal to 1 such that the corresponding decoded base representations are stored when the decoding operates at the current tier. A value of 0 indicates that the members of this group may have any value of store_base_rep_flag.

`is_tl_switching_point` when set to 1, indicates that, for the members of this group, those having the highest value of temporal_id as specified in ISO/IEC 14496-10 Annex G are temporal layer switching points. Let the highest value of temporal_id of the members of this group as tId, then the bitstream can be switched at any of the members having temporal_id equal to tId from the temporal layer with temporal_id equal to tId-1 to the temporal layer with temporal_id equal to tId, provided that the members with temporal_id equal to tId-1 indicated by tl_switching_distance have been processed (transmitted and decoded). `is_tl_switching_point` equal to 0 indicates that the members of this group having the highest value of temporal_id as specified in ISO/IEC 14496-10 Annex G may or may not be temporal layer switching points.

`tl_switching_distance` is used when is_tl_switching_point is 1. It indicates the number of samples of the temporal layer with temporal_id equal to tId-1 that must be decoded to ensure decodability of the stream at or above temporal layer tId from the switching point onward. The value 0 indicates a temporal switching point with no dependency on the lower temporal layer. This required distance for a particular sample may be reduced by a temporal layer switching distance statement in the time parallel metadata track for a specific sample.

C.3 Mapping NAL units to map groups and tiers

C.3.1 Introduction

In order to describe the layer representations within an SVC access unit, two kinds of sample groups are used:

a) a group to describe sections of a sample. For each of the groups a ScalableGroupEntry exists which defines the group properties. Note that these describe tiers, not the entire stream, and therefore describe the NAL units belonging to one tier at any instant, not the entire AU. See subsections C.1 and C.2.
b) a map group, that describes the mapping of each NAL unit inside an AU to a map group (of grouping_type 'scnm'). For each different sequence of NAL units belonging to a particular map group, a ScalableNALUMapEntry exists. Within an AU a map group includes all NAL units of a tier.

Defining map groups requires that there is a limited number of map grouping patterns for all access units. If there is a varying number of NAL units in successive access units for a given tier, Aggregators can be used to make these varying structures consistent and to reduce the number of map groups required.

C.3.2 Map group definition

Each sample is associated with a group_description_index in the SampleToGroupBox with grouping_type '`scnm`'. A SampleGroupDescriptionBox with grouping_type '`scnm`' contains a ScalableNALUMapEntry for each group_description_index.

```
class ScalableNALUMapEntry() extends VisualSampleGroupEntry ('scnm') {
   unsigned int(8) reserved = 0;
   unsigned int(8) NALU_count;
   for (i=1; i<= NALU_count; i++)
      unsigned int(8) groupID;
   }
}
```


© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Each sample belonging to a given map group has exactly `NALU_count` NAL units in it (possibly by using aggregators to group together NAL units of the same layer). Each of those NAL units maps to the corresponding scalable group as described by the `groupID`.

NOTE 1 An arbitrarily chosen groupID is used here, rather than the more obvious scalable group index form the sample group description box, so that if scalable groups are deleted or re-ordered these operations can be detected and handled. Note also that there may be one or more scalable groups in a given tier.

NOTE 2 If movie fragments are used, new maps cannot be introduced in the fragments (only the association of the new samples to pre-existing maps). In this case, care should be taken to introduce, in the movie box, all the maps that may be needed.

C.4 Decode re-timing groups

C.4.1 Introduction

When temporal layers are discarded, re-timing the decoding times of some or all samples may be needed to ensure that the stream complies with all buffer and HRD requirements. Also re-timing may improve the transmission and decoding process. Composition times are not affected. If the stream is 'thinned' to tierID X, and there is a re-timing sample grouping for tierID Y, where Y is the largest such tierID less than X that contains re-timing sample grouping, then the adjusted decode time is the time from the time-to-sample table (the original decode time), plus the given re-timing: newDTS = oldDTS + delta. The CTS is given as usual by the composition time to sample table: CTS = oldDTS + compositionOffset, which is CTS = newDTS - delta + compositionOffset.

This re-timing is given as sample groups, which are associated with samples by using the normal sample-to-group structures. Each group provides a set of re-timing deltas and their associated tierIDs. The group definition must be ordered by increasing tierID.

C.4.2 Syntax

```
class DecodeRetimingEntry() extends VisualSampleGroupEntry ('dtrt') {
   unsigned int(8) tierCount;
   for (i=1; i<=tierCount; i++) {
      unsigned int(16) tierID;
      signed int(16) delta;
   }
}
```

C.4.3 Semantics

`tierID` gives the ID of a tier that maps to a temporal level; the tiers with equal or greater tierID, up to the next tierID in this group, use the given decode time delta

`delta` provides an adjustment for the decode time

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Annex D
(normative)

Temporal metadata support

D.1 Introduction

A timed metadata track, as defined in ISO/IEC 14496-12, may be used to provide temporal information about an AVC or SVC track.

This metadata is stored in metadata tracks. These tracks have a handler type 'meta' and are linked to the media track using a track reference of type 'cdsc' as specified in ISO/IEC 14496-12. The metadata is stored in samples, the decoding time of which is equal to the media samples it describes. Composition offsets are permitted but not required in SVC timed metadata tracks, but, if used, the composition timing must match the composition timing of the associated media track.

The metadata is structured using conceptual *statements*. Each statement has a one-byte type field – indicating what it is asserting, and a size, which is the length of its payload in bytes, *not* including the size and type fields. The length of the size field depends on the type field.

There are two important 'structuring' statements, groupOfStatements and sequenceOfStatements.

The statement groupOfStatements allows several statements to be made about one thing, by grouping them. A groupOfStatements contains a set of statements all of which are asserted about the thing described.

The statement type sequenceOfStatements may be used in the description of the entire sample or of a NAL unit in the media stream that is an Aggregator or Extractor, to describe its sequence of NAL units. A sequenceOfStatements contains a set of statements, which are in one-to-one correspondence with the sequence of contained objects in that which is described.

Each metadata sample is a collection (a group or sequence) of one or more statements about the temporally aligned media sample. Each of the statements in the collection may have a default type from the sample entry, or have an explicit single type in each statement. Similarly, the default length may be indicated in the sample entry, or be inline in each sample. The overall sample is a collection of N statements. The sample entry provides the statement type of each sample (group or sequence), and (optionally) the default type and length values of the statements in the sample. It can also supply a statement which is true of every sample described by this metadata (an 'overall' statement).

There is a set of pre-defined statement types defined in this International Standard, and there is explicit provision for extension statements by other bodies.

There are statement types reserved to ISO, and other statement types reserved for dynamic assignment. Dynamic assignment consists of a table in the Sample Entry of the metadata track, containing pairs mapping a local statement ID to URIs.

The allocation of different categories of statement types is as follows:

0	no metadata (empty statement), reserved to ISO
1-95	short, reserved to ISO
96-191	short, user extension
192-223	long, reserved to ISO
224-255	long, user extension

--`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

The URIs are used in the same sense as namespace identifiers in XML; they are not guaranteed to be de-referenceable. If URLs are used, they should contain a month indication in the form yyyymm, indicating that this use of the domain in the URL was authorized by the owner of that domain as of that month. An example may be:

2 maps to http://www.example.com/200610/gateway-types#quality-model

For the purposes of this metadata, a prefix NAL unit and its associated AVC NAL unit are considered as one NAL unit, and the prefix NAL unit provides the NAL unit header values except for the NAL unit type, which is taken from the associated AVC NAL unit.

An aggregator NAL unit may be described, and if it is described by a sequence, the elements in the sequence correspond one-one to the NAL units aggregated by both inclusion and reference. Similarly, an extractor may be described, and if it is described by a sequence, the elements in the sequence correspond one-one to the NAL units in the extracted data.

In all sequences, SEI NAL units are treated as any other NAL unit. If they need to be skipped in sequences, an empty statement or a NAL header statement can be used.

D.2 Connection to the SVC media data

A metadata sample may store a data structure for each NAL unit in the media data stream. The metadata sample must be temporally aligned to the media data sample (in decoding time).

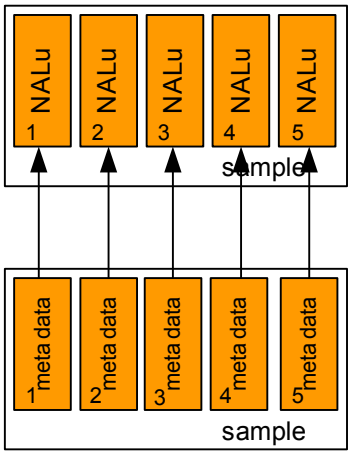

Figure D.1 — Connection between media data and metadata

NOTE As illustrated in Figure D.1 synchronization between the NAL units in the media data stream and the corresponding meta information is done by using the same structure in both streams (e.g. by counting NAL units and metadata statements).

D.3 SVC meta data sample entry

D.3.1 Definition

The SVC Metadata Sample Entry extends the Metadata Sample Entry and includes a configuration box which defines some default values for the samples and statements. It can also supply a statement which is valid for every sample described by this metadata sample entry, and a mapping of user extension statements to URIs.

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

The following example sample entry field values demonstrate different default possibilities:

sample_statement_type = groupOfStatements, default_statement_type=0, default_statement_length=0
— this is the 'normal' case, a group of statements about the entire sample

sample_statement_type = sequenceOfStatements, default_statement_type=0, default_statement_length=0
— where no statement needs to be made about the overall sample

sample_statement_type = sequenceOfStatements, default_statement_type=NALHeaderStatement, default_statement_length=N
— compact samples consisting merely of NAL headers of length N, for each aligned NAL unit

If priority override statements are used, then a priority assignment box may be present, providing the names of the methods referenced. When the box occurs here, the method_count may be greater than 1.

D.3.2 Syntax

```
class statement (default_type, default_length) {
   int st_type, field_size = 0;     // local variables, not fields
   int st_length = 0, body_len = 0; // local variables, not fields

   if (default_type != 0) st_type = default_type;
   else {
         unsigned int(8)   statement_type;
         st_type    = statement_type;
         st_length  = 1;
   }

   if (default_length != 0) body_len = default_length;
   else {
      if   (st_type >= 192) field_size = 4;
      else if (st_type > 0) field_size = 1;
      else  { body_len = 0; field_size = 0; }
      if (field_size > 0) {
         unsigned int(8*field_size) statement_length;
         body_len = statement_length;
         st_length += field_size;
      }
   }

   unsigned int(8) statement_body[body_len];
   st_length += body_len;
}

class SVCMetadataSampleConfigBox extends FullBox('svmC')
{
   int i;      // local variable, not a field
   unsigned int(8) sample_statement_type;  /* normally group, or seq */
   unsigned int(8) default_statement_type;
   unsigned int(8) default_statement_length;
   unsigned int(8) entry_count;
   for (i=1; i<=entry_count; i++) {
      unsigned int(8) statement_type;   // from the user extension ranges
      string statement_namespace;
   }
   statement(0,0) overall_statement; /* NB may be the empty statement, type==0 */
}
```


--`,,```,,,,````-`-`,,`,,`,`,,`---
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
© ISO/IEC 2008 – All rights reserved
Not for Resale

```
class SVCPriorityLayerInfoBox extends Box('qlif'){
   unsigned int(8) pr_layer_num;
   for(j=0; j< pr_layer_num; j++){
      unsigned int(8) pr_layer;
      unsigned int(24) profile_level_idc;
      unsigned int(32) max_bitrate;
      unsigned int(32) avg_bitrate;
   }
}

class SVCMetadataSampleEntry () extends MetadataSampleEntry('svcM')
{
   SVCMetadataSampleConfigBox config;
   SVCPriorityAssignmentBox   methods;     // optional
   SVCPriorityLayerInfoBox    priorities;  // optional
}

class MetaDataSample {
   int totalLength;     // local variable, not a field
   for (totalLength = 0; totalLength<sample_size; ) {
      statement(default_statement_type, default_statement_length) the_statement;
      totalLength += the_statement.st_length;
   }
}
```

D.3.3 Semantics

`statement_type` - an integer identifying a statement type defined in this International Standard, or dynamically defined in the corresponding sample entry of this track. In the SVCMetaDataSampleEntry this entry also defines the namespace mapping for this statement type of a dynamic statement. When used to define a mapping, the statement_type must have a value taken from the ranges reserved for user extensions.

`statement_length` - the length in bytes of the statement body, not including the statement_type and statement_length fields.

`statement_namespace` - gives a valid URI, in null-terminated UTF-8 characters; if a URL, it should contain a month in the form yyyymm, defined or approved by the owner of the domain name in that URL as of the month indicated.

`statement_body` - the contents of the statement, as defined by the statement type.

`sample_statement_type` - describes whether the collection of statements in each sample associated with this sample entry is either a group (each describing the whole sample) or a sequence (each mapped to a NAL unit of the sample), and therefore normally takes the value groupOfStatements or sequenceOfStatements.

`default_statement_type` - in the case where all the first-level statements in all the associated samples have the same type, that type can be supplied here and then not be present in each sample; this would normally only be used when the sample_statement_type is sequenceOfStatements. If no default is needed, then 0 should be supplied in this field.

`default_statement_length` - in the case where all statements in all the associated samples have the same length, that length can be supplied here and then not be present in each sample. If no default is needed, then 0 should be supplied in this field.

`pr_layer_num` specifies the number of the priority layer.

`pr_layer` specifies the identifier of the priority layer. Priority layer identifiers are unique across the stream that is mapped to this metadata stream.

`profile_level_idc` specifies the profile and level compliancy of the bitstream of the priority layer identified by pr_layer. profile_level_idc is the exact copy of the three bytes comprised of profile_idc, constraint_set0_flag, constraint_set1_flag, constraint_set2_flag, constraint_set3_flag and level_idc, if these syntax elements were used to specify the profile and level compliancy of the bitstream of the priority layer.

--`,,```,,,,````-`-`,,`,,`,`,,`---
© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

`max_bitrate` specifies the maximum bit rate, in units of 1000 bits per second, of the bitstream of the priority layer identified by pr_layer in any one-second time window.

`avg_bitrate` specifies the average bit rate, in units of 1000 bits per second, of the bitstream of the priority layer identified by pr_layer.

D.4 Helper Functions

```
function next_NALu (){
      // the next statement is made about the next NAL unit
}
```

D.5 Statement Types

The following statement types, and their required contents, are defined.

0 emptyStatement: when nothing needs to be said about the thing described (allows skipping of an item in a sequence)

192 groupOfStatements: the contents of the statement are exactly a set of statements, all of which apply to the described sample or NAL unit

```
class groupOfStatementsBody(size) {
   int i=0;
   do {
      statement(0,0) the_statement;
      i+=the_statement.st_length;
   } while (i < size);
}
```

193 sequenceOfStatements: the contents of the statement are exactly a set of statements, which describe, in one-to-one correspondence, the contents of a sample, extractor or aggregator (except note that the inlinesequenceOfStatements corresponds to one or more NAL units)

```
class sequenceOfStatementsBody(size) {
   int i=0;
   do {
      statement(0,0) the_statement;
      i+=the_statement.st_length;
      next_NALu();
   } while (i < size);
}
```

194 sequenceOfFixedStatements: same semantics as for sequenceOfStatements, except that the contents of the statement are a one-byte statement type, followed by a one-byte length indication; then follows a number of statements of that single type, each of the same length. The number of statements following is given by (statement_length-2)/fixedLength The fixedLength shall be greater than zero.

```
class sequenceOfFixedStatementsBody(size) {
   int i=2;
   unsigned int(8) fixedType;
   unsigned int(8) fixedLength;
   do {
      statement(fixedType, fixedLength) the_statement;
      i+=fixedLength;
      next_NALu();
   } while (i < size);
}
```


© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

195 inlineGroupOfStatements; this structure can be used to describe a consecutive set of items (NAL units). This structure starts with a one-byte count of the number of items described, and then describes all these items together with a number of statements (like a groupOfStatements). If individual statements about each item are also desired, a sequenceOfStatements may be included in the inlineGroupOfStatements to describe the items individually. In this case the included sequenceOfStatements shall describe as many items as specified by the value of count.

```
class inlineGroupOfStatementsBody(size) {
   int i=1;
   unsigned int(8) count;
   do {
      statement(0,0) the_statement;
      i+=the_statement.st_length;
   } while (i < size);
   for (j=0; j<count; j++)
      next_NALu();
}
```

1 itemLengthStatement: 1, 2, or 4 bytes of payload containing the length of the corresponding item (sample, group, or NAL unit).

2 aggregatorStatement: indicates that the item described is an Aggregator. If contained in a groupOfStatements about the Aggregator, the aggregatorStatement shall be the first statement in the groupOfStatements. The groupOfStatements may contain additional statements about the whole aggregation and a sequenceOfStatements to individually describe the NAL units aggregated. An aggregatorStatement contains no body.

3 extractorStatement: indicates that the item described is an Extractor. If contained in a groupOfStatements about the Extractor, the extractorStatement shall be the first statement in the groupOfStatements. The groupOfStatements may contain additional statements about all NAL units referenced and a sequenceOfStatements to individually describe the NAL units referenced. An extractorStatement contains no body.

4 overridePriorityStatement:

```
class overridePriorityStatementBody(size) {
   unsigned int(8) priority_assignment_method_index;
   bit(2) reserved = 0;
   bit(6) priority_id;
}
```

Contains a value for priority_id which may replace the priority_id value in the NAL unit header of the corresponding NAL unit. The field priority_assignment_method_index identifies the method used to calculate the priorities, as a 1-based index into the priority assignment URI box in the metadata sample entry (if any). There may be more than one of these statements for a given NAL unit; if there are several, they must differ in the value of the priority_assignment_method_index. If a stream is logically or physically re-labelled with priority_id values from these statements, then the same method must be used for all NAL units. If, for a given NAL unit, the priority_id value desired for a given method is the same as the value in the bitstream, then this statement may be omitted for that method for that NAL unit.

5 priorityRangeStatement: This is used when multiple NAL units are described. This contains two bytes, each containing a priority value in their lower 6 bits. The first is the lowest P value and the second the highest in the matching NAL units.

```
class priorityRangeStatementBody(size) {
   bit(2) reserved = 0;
   bit(6) min_priority_id;
   bit(2) reserved = 0;
   bit(6) max_priority_id;
}
```

--`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

6 DTQrangeStatement: This is used when multiple NAL units are described. The fields are defined exactly as for the SVCDependencyRangeBox.

```
class DTQRangeStatementBody(size) {
   bit(3) min_dependency_id;
   bit(3) min_temporal_id;
   bit(6) reserved = 0;
   bit(4) min_quality_id;
   bit(3) max_dependency_id;
   bit(3) max_temporal_id;
   bit(6) reserved = 0;
   bit(4) max_quality_id;
}
```

7 ROIindicationStatement: applies to a set of NAL units and gives IROI information.

```
class iroiStatementBody() {
   unsigned int(16) tierID;
   unsigned int(24) roi_id;
}
```

`tierID` specifies the tier the `roi_id` is defined in.

`roi_id` gives the ID of the ROI to which the NAL units pertaining to this statement belong.

8 scalabilityInfoStatement. This statement contains the header information from the matching NAL unit. The syntax is below, and the fields are as defined in ISO/IEC 14496-10 Annex G for NAL unit headers with NAL unit header SVC extension. The first byte is taken from the matching NAL unit, and the remaining bytes also if it is an SVC VCL NAL unit. If it is an AVC NAL unit, the remaining bytes are taken from the prefix NAL unit, if any, or else filled with zeroes.

```
class scalabilityInfoStatementBody() {
   bit (1)  forbidden_zero_bit;
   bit (2)  nal_ref_idc;
   bit (5)  nal_unit_type;
   bit (1)  reserved_zero_one_bit;
   bit (1)  idr_flag;
   bit (6)  priority_id;
   bit (1)  no_inter_layer_pred_flag;
   bit (3)  dependency_id;
   bit (4)  quality_id;
   bit (3)  temporal_id;
   bit (1)  use_ref_base_pic_flag;
   bit (1)  discardable_flag;
   bit (1)  output_flag;
   bit (2)  reserved_three_2bits;
}
```

9 temporalLayerSwitchingDistanceStatement. This statement provides a smaller value than that supplied in the group for tl_switching_distance for the tier, for switching points where the maximum value indicated in the tier is not needed.

```
   class TLSwitchingDistanceStatementBody() {
      unsigned int(8) groupID;
      unsigned int(8) alt_tl_switching_distance;
   }
```

`alt_tl_switching_distance` specifies a smaller value than tl_switching_distance in the group for the tier that applies to the target samples of the current statement.

10 priorityLayerStatement. This statement provides the priority layer to which the corresponding NAL unit(s) are mapped. Decoding can be performed at consistent quality by selecting the NAL units

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

that are at, or below, a given priority layer. The body is a single 8-bit integer, the priority layer identifier. The characteristics of the priority layer are given in the optional SVCPriorityLayerInfoBox in the metadata sample entry.

```
class PriorityLayerStatementBody() {
   unsigned int(8) priorityLayer;
}
```

`prorityLayer` specifies the identifier of the priority layer the NAL unit(s) are mapped to. These identifiers shall be unique across the stream that is mapped to this metadata stream, i.e., a same value shall not be reused by NAL units with different dependency_id values.

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Annex E
(normative)

File format toolsets

This annex defines what constitutes tools, for the purposes of branding files containing AVC or SVC content. A specific brand may require some or all of the tools indicated here. A brand should be chosen that indicates the full level of support required, including any requirements on other specifications (e.g. support for aspects of the ISO base media file format specification, ISO/IEC 14496-12).

For all toolsets the implementation of the SVC specific definitions from Annex A are required.

The following toolsets are defined:

- *SVCExtractor*: this toolset includes Extractors (Annex B).
- *SVCAggregator:* this toolset includes Aggregators (Annex B).
- *SVCTiers*: this toolset includes map/group/tier implementation (Annex C).
- *SVCTimedMetaData*: this toolset includes the techniques from Annex D.

NOTE The SVCTiers and the SVCTimedMetaData toolsets define descriptive tools; if the file reader does not need this information, these toolsets need not be implemented as the video data can be processed without them. Extractors and Aggregators, however, are in-stream structures and must be implemented under some circumstances to yield the correct video stream. A brand requiring SVCTiers or SVCTimedMetaData might also need to require SVCAggregator.

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

ICS 35.040

Price based on 40 pages

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale