INTERNATIONAL STANDARD

ISO/IEC 15938-4

First edition
2002-06-15

AMENDMENT 1
2004-04-15

Information technology — Multimedia content description interface —

Part 4: **Audio**

AMENDMENT 1: Audio extensions

Technologies de l'information — Interface de description du contenu multimédia —

Partie 4: Audio

AMENDEMENT 1: Extensions audio

Reference number
ISO/IEC 15938-4:2002/Amd.1:2004(E)

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

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 shall not be held responsible for identifying any or all such patent rights.

Amendment 1 to ISO/IEC 15938-4:2002 was prepared by Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

Information technology — Multimedia content description interface —

Part 4: **Audio**

AMENDMENT 1: Audio extensions

Add at the end of subclause 4.2:

4.3 Handling of multi-channel signals

Introduction:

The framework to handle multi-channel signals is given by the `AudioD` and `AudioDS` Types defined in ISO/IEC 15938-5/Amd.1 (MDS). The new additional attribute `channels` gives the channel numbers that are described by the assigned Descriptor or Description Scheme. However, to prevent some misunderstanding, a more detailed description and handling policy is given in this part of ISO/IEC 15938. In particular a recommendation is given to handle typical surround formats, when only tag names like L, C, R, LS, LR, LFE are known.

By using the `channels` attribute, defined in ISO/IEC 15938-5/Amd.1 (MDS) it is possible to specify which channels should be used for e.g. computing the mean with the extraction method. Therefore, the Descriptor and Description Schemes contain information about these channels only. This is useful in order to separate a multi-channel input signal into subgroups that are closely related, e.g. the Left (L), Center (C) and Right (R) signal of a typical surround format. The highest possible channel number is given in the file-format of the audio media file itself. All numbers given in the `channels` attribute higher than the number of channels given by the media file-format should be ignored.

In the case where the numbering of the audio channels is not explicitly given in the file-format (like 5.1 surround signals), the following convention to number the channels is recommended to be used.

When mapping typical surround file-formats, consisting of tags like (L, R, C, LS, RS, LFE), the scheme shown in Figure AMD1-1 should be followed, in order to reduce the ambiguity between scheme and channel number. To define the channel number, the counting should start at an optional center channel and go from left to right, top to bottom and then from front to back (see for example Figure AMD1-2). An optional rear center will get the last channel number for the standard audio channels. The assigned number can be higher if specialised channels are present, like an LFE channel for low frequency effect signals. Two examples are given in Tables 1 and 2. Furthermore, it is recommended that a textual description of the scheme used inside the AudioSegmentD-Framework (defined in ISO/IEC 15938-5) is included. An instantiation example is given in ISO/IEC 15938-5/Amd.1 (MDS), subclause 4.2.4.

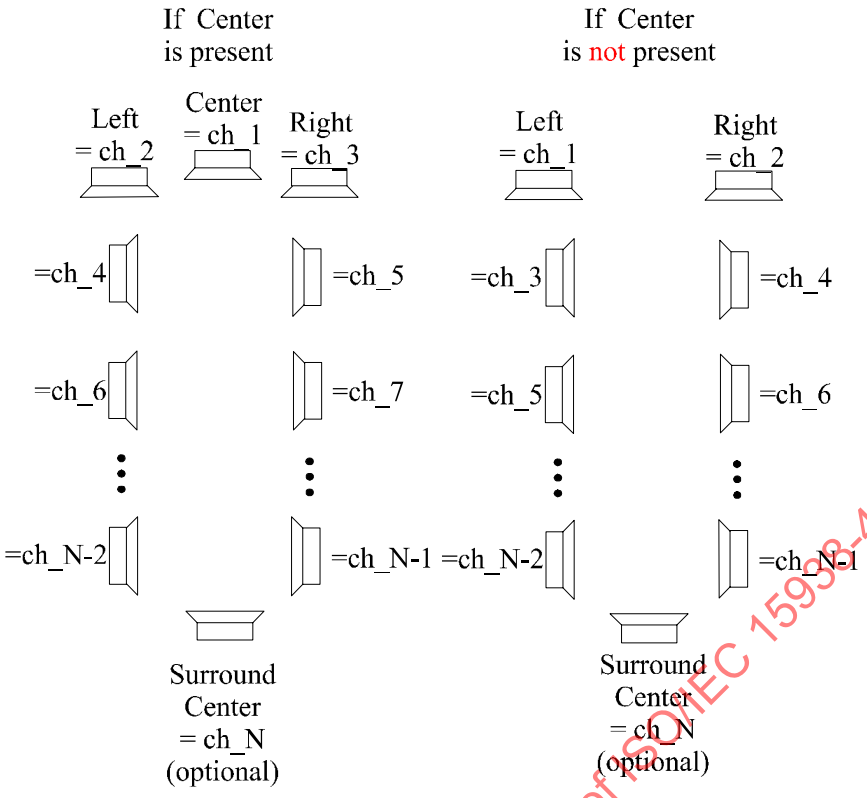

Figure AMD1-1 - Scheme and channel number for typical surround file-formats

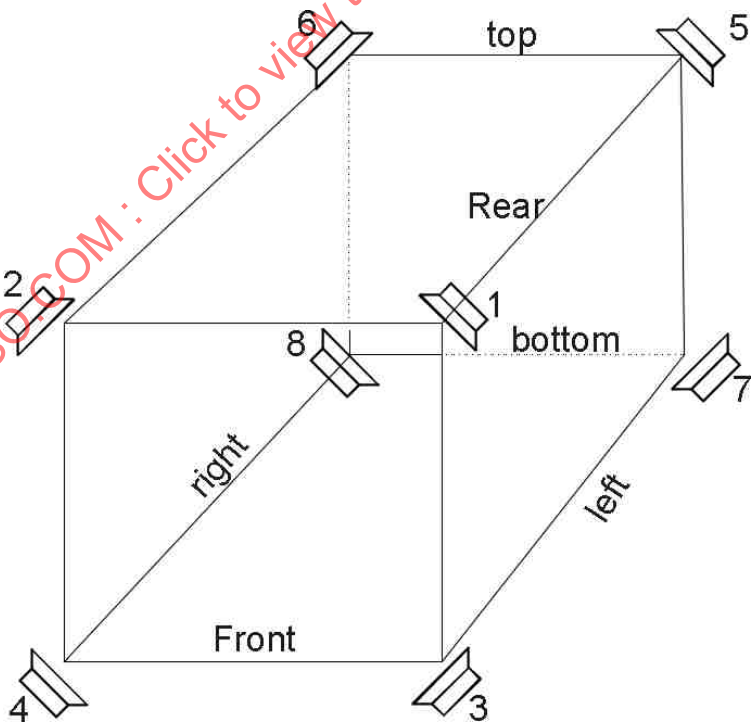

Figure AMD1-2 - Scheme and channel number for a 3D speaker arrangement (example)

Examples for mapping:

Table AMD1-1- Simple Stereo	
Tag name	channel number
Left	1
Right	2

TableAMD1-2- Surround 5.1	
Tag name	Channel number
Center	1
Left	2
Right	3
Left Surround (LS)	4
Right Surround (LR)	5
LFE	6

Replace subclause 6.5.8 by:

6.5.8 WordLexiconType

6.5.8.1 Syntax

```
<!-- ################################################################## -->
<!-- Definition of WordLexiconType                                      -->
<!-- ################################################################## -->
<complexType name="WordLexiconType">
  <complexContent>
    <extension base="mpeg7:LexiconType">
      <sequence>
        <element name="Token" minOccurs ="1" maxOccurs="4294967296">
          <complexType> <!-- New attribute V2 -->
            <simpleContent>
              <extension base="mpeg7:WordType">
                <attribute name="linguisticUnit" use="optional"
                           default="word">
                  <simpleType>
                    <union>
                      <simpleType>
                        <restriction base="NMTOKEN">
                          <enumeration value="word"/>
                          <enumeration value="syllable"/>
                          <enumeration value="morpheme"/>
                          <enumeration value="phrase"/>
                          <enumeration value="component"/>
                          <enumeration value="stem"/>
                          <enumeration value="affix"/>
                          <enumeration value="nonspeech"/>
                          <enumeration value="other"/>
                        </restriction>
                      </simpleType>
                      <simpleType>
                        <restriction base="mpeg7:termReferenceType"/>
                      </simpleType>
```

```
                    </union>
                  </simpleType>
                </attribute>
                <attribute name="representation" use="optional"
                           default="orthographic">
                  <simpleType>
                    <restriction base="NMTOKEN">
                      <enumeration value="orthographic"/>
                      <enumeration value="nonorthographic"/>
                    </restriction>
                  </simpleType>
                </attribute>
              </extension>
            </simpleContent>
          </complexType>
        </element>
      </sequence>
      <attribute name="phoneticAlphabet" type="mpeg7:phoneticAlphabetType"
                 use="optional"/> <!-- New attribute V2 -->
    </extension>
  </complexContent>
</complexType>
```

6.5.8.2 Semantics

Name	Definition
`WordLexiconType`	A list of words (a lexicon). Each entry represents one orthographic representation (spelling) or one non-orthographic representation of a word or linguistic unit. The lexicon is not a phonetic (pronunciation) dictionary.
`phoneticAlphabet`	The name of the encoding scheme of the phone lexicon. Only needed if phonetic representation is used. See 6.5.9 `phoneticAlphabetType`
`Token`	An entry in the lexicon
`linguisticUnit`	Indicates the type of the linguistic unit that is put into the entry of the word lexicon. The linguistic units are defined as follows. • *word* – an unit delimitated by whitespace. This is the default value. (example: psychcoacoustics) • *syllable* – minimal pronouncable unit (example: psy) • *morpheme* – minimal meaning bearing unit (example: psycho) • *stem* – the uninflected base of a word-form, can be polymorphemic. (example: psychoacoustic) • *affix* – needs to be added to a stem to get a word • *component* – a constituent part of a compound word. Important for compounding languages. (example from German: *Forschungs* (in English corresponds to "research-")) • *nonspeech* – noises, both human-produced and background, that are non-linguistic in nature. (example: throat clearing, coughing) • *phrase* - a sequence of words (e.g. “God bless America”) • *other* - a linguistic unit that does not map onto any of the above Other values that are datatype-valid with respect to `mpeg7:termReferenceType` are reserved.

`representation`	Form of representation for a lexicon entry. The kinds of representation are defined as follows. • *orthographic* – representation of an entry by spelling • *nonorthographic* – representation of an entry by an identifier that is not synonymous with the spelling of a word. A non-orthographic representation may, for example, encode the phoneme string corresponding to the pronunciation of the entry.

6.5.8.3 Usage, Extraction and Examples (informative)

6.5.8.3.1 Purpose

The word lexicon makes it possible to store the words contained in the lattice. It is common in both speech recognition and spoken document retrieval to include entries in the word lexicon that are “words” only in a wider sense of the term (e.g. acronyms or abbreviations) or not really words at all (e.g. phrases, syllables, morphemes or the individual components of compound words).

The attribute `linguisticUnit` makes it possible to distinguish between these different types of units. Differentiating these units is useful, for instance, when the retrieval algorithm of an application needs to treat different units in different ways. For example stemming, a pre-processing step applied to words, should not be applied to syllables or morphemes. Similarly, different types of units might receive different weightings in the calculation of the retrieval metric.

In some applications, it is also necessary to know if the entry is given in its human-readable form or not. For example, if the entry is human-interpretable and can potentially be displayed to the user or if certain algorithms are applied which are intended for the orthographic form only (e.g. stemming).

6.5.8.3.2 Extraction

The generation of syllable, morpheme, compound and phrase transcriptions of spoken input is performed in the following ways:

a) The output of the word or phoneme recognizer is mapped to other linguistic units. For example the recognized word can be transformed into syllables using a syllable generation tool.
b) The ASR system produces the desired linguist unit directly during the recognition process. In this case, the linguistic units are parts of the recognition vocabulary of the speech recognition engine. For example, the dictionary used for the speech recognition system could be composed exclusively of syllables.

6.5.8.3.3 Example

The following example shows a lexicon containing six entries. The first two entries represent syllables and the next two entries represent words. The fifth entry also represents a word, but not in its written form. The last entry represents a phrase.

```
<WordLexicon id="lex1" phoneticAlphabet="other">
  <Token linguisticUnit="syllable" representation="nonorthographic">
          6: n
  </Token>
  <Token linguisticUnit="syllable" representation="nonorthographic">
          6:_s_
  </Token>
  <Token>water</Token>
  <Token linguisticUnit="word">draw</Token>
  <Token representation="nonorthographic">Q e: l e: f a n t</Token>
```

```
    <Token linguisticUnit="phrase" representation="orthographic">
        as a rule
    </Token>
  </WordLexicon>
```

Replace subclause 6.5.12 by:

6.5.12 SpokenContentLinkType

6.5.12.1 Syntax

```
<!-- ################################################################ -->
<!-- Definition of SpokenContentLink datatype                         -->
<!-- ################################################################ -->
<complexType name="SpokenContentLinkType">
  <attribute name="probability" type="mpeg7:zeroToOneType" default="1.0"/>
  <attribute name="acousticScore" type="mpeg7:nonNegativeReal"
             use="optional" /> <!-- New attribute V2 -->
  <attribute name="nodeOffset" type="mpeg7:unsigned16" default="1"/>
</complexType>
```

6.5.12.2 Semantics

Name	Definition
SpokenContentLinkType	The structure of a word or phone link in the lattice
probability	The probability of this link. In a crude sense, this is to indicate which links are more likely than others, with larger numbers indicating higher likelihood.
nodeOffset	The node to which this link leads, specified as a relative offset and defaulting to 1. A node offset leading out of the current block implicitly refers to the next block. A node offset cannot span a whole block, i.e., a link from a node in block 3 must lead to a node in block 3 or block 4.
acousticScore	The score assigned by the acoustic models of the speech recognition engine only. It is given in logarithmic scale (base e) and indicates the quality of the match between the acoustic models and the corresponding signal segment. A higher value indicates a better match.

Add a new subclause 6.7:

6.7 Audio Signal Quality

6.7.1 Introduction

If an AudioSegment DS contains a piece of music, several features describing the signal's quality can be computed to describe the quality attributes. The AudioSignalQualityType contains these quality attributes and uses the ErrorEventType to handle typical errors that occur in audio data and in the transfer process from analog audio to the digital domain. However, note that this DS is not applicable to describe the subjective sound quality of audio signals resulting from sophisticated digital signal processing, including the use of noise shaping or other techniques based on perceptual/psychoacoustic considerations.

For example, in the case of searching an audio file on the Internet, quality information could be used to determine which one should be downloaded among several search results. Another application area would be an archiving system. There, it would be possible to browse through the archive using quality information, and also the information could be used to decide if a file is of sufficient quality to be used e.g. for broadcasting.

6.7.2 Conventions

The description of the Descriptors refers to the input signal x. If x is a multi channel signal, then the signal for a certain channel is designated as x_n for the n-th channel. The functions max(), min() and mean() are used as defined in ISO/IEC CD 15938-4 (Audio Part). The function abs() calculates the absolute value.

6.7.3 Audio Signal Quality Description Scheme

The `AudioSignalQualityType` is a set of `AudioQuality` Descriptors and some additional tools for handling and describing audio signal quality information. In particular the handling of single error events in audio streams is considered.

6.7.3.1 Syntax

```
<!-- ################################################ -->
<!--   Definition of AudioSignalQuality DS            -->
<!-- ################################################ -->
<complexType name="AudioSignalQualityType">
  <complexContent>
    <extension base="mpeg7:AudioDSType">
      <sequence>
        <!--  Summaries-->
        <element name="Operator" type="mpeg7:PersonType" minOccurs="0"/>
        <element name="UsedTool" type="mpeg7:CreationToolType" minOccurs="0"/>
        <element name="BackgroundNoiseLevel"
type="mpeg7:BackgroundNoiseLevelType"/>
        <element name="RelativeDelay" type="mpeg7:RelativeDelayType"/>
        <element name="Balance" type="mpeg7:BalanceType"/>
        <element name="DcOffset" type="mpeg7:DcOffsetType"/>
        <element name="CrossChannelCorrelation"
type="mpeg7:CrossChannelCorrelationType"/>
        <element name="Bandwidth" type="mpeg7:BandwidthType"/>
        <!--  High Level Feature-->
        <element name="TransmissionTechnology"
type="mpeg7:TransmissionTechnologyType" minOccurs="0"/>
        <!--  Events-->
        <element name="ErrorEventList" minOccurs="0">
          <complexType>
            <sequence>
              <element name="ErrorEvent" type="mpeg7:ErrorEventType"
minOccurs="0" maxOccurs="unbounded"/>
            </sequence>
          </complexType>
        </element>
      </sequence>
      <!--  Attributes-->
      <attribute name="IsOriginalMono" type="boolean"/>
      <attribute name="BroadcastReady" type="boolean" use="optional"/>
    </extension>
  </complexContent>
</complexType>
```

6.7.3.2 Semantics

Name	Definition
AudioSignalQualityType	The AudioSignalQualityType describes the quality of an AudioSegment. It consists of several quality elements.
Operator	The Operator is the person who is responsible for the audio quality information. Operator is of type PersonType.
UsedTool	The UsedTool is the system that was used by the Operator to create the quality information. UsedTool is of type CreationToolType.
BackgroundNoiseLevel (BNL)	The BackgroundNoiseLevel describes the noise level in an AudioSegment. BackgroundNoiseLevelType is defined in 6.7.4.
RelativeDelay	The RelativeDelay describes the relative delay between two or more channels of an AudioSegment. RelativeDelayType is defined in 6.7.6.
Balance	The Balance describes the relative level between two or more channels of an AudioSegment. BalanceType is defined in 6.7.7.
DcOffset	The DcOffset describes the mean value relative to the maximum of each channel of an AudioSegment. DcOffsetType is defined in 6.7.8.
CrossChannelCorrelation	The CrossChannelCorrelation describes the correlation between two or more channels of an AudioSegment. CrossChannelCorrelation Type is defined in 6.7.5.
Bandwidth	The Bandwidth describes the upper limit of the signal's bandwidth for each channel. BandwidthType is defined in 6.7.9.
IsOriginalMono	The IsOriginalMono attribute describes if a signal was originally Mono, i.e. presently it has more than one channel. IsOriginalMono is of type boolean Possible values: false and true
BroadcastReady	The BroadcastReady attribute describes whether or not the sound material is ready for broadcasting. BroadcastReady is of type boolean Possible values: false and true
TransmissionTechnology	The TransmissionTechnology describes the technology with which the audio file was transmitted or recorded using the TransmissionTechnologyCS classification scheme (see Annex B).
ErrorEventList	The ErrorEventList contains different ErrorEvents.
ErrorEvent	The ErrorEvent describes the event time of a specified error type in the signal. ErrorEventType is defined in 6.7.11.

6.7.3.3 Purpose, Extraction and Usage

6.7.3.3.1 BroadcastReady (Purpose)

This value can be set by an operator. For historical material the quality may be quite bad, but it may be ready for broadcasting since no improvement in quality is possible. The subjective measurement should be considered.

6.7.3.3.2 IsOriginalMono

6.7.3.3.2.1 Purpose

The `IsOriginalMonoType` describes whether or not a signal was originally recorded as a mono signal. Since modern playback devices will mostly produce a stereo output, the mono signal is now stereo since the noise components are not the same as the two output channels. Therefore, a check to see if the original sound file was mono is useful to describe the audio material.

6.7.3.3.2.2 Extraction

To extract the IsOriginalMonoType the following method is suggested:

The calculation of `IsOriginalMonoType` is similar to the calculation of the CrossChannelCorrelation coefficient. Normalize each channel to its maximum value, then calculate the cross correlations c_{x1xn} between the first channel and the other N-1 channels, and normalize them all to the mean maximum of the first channel's autocorrelation c_{x1x1} and the n-th channel's autocorrelation c_{xnxn}, which is defined as sqrt($c_{x1x1}(0)$ $c_{xnxn}(0)$). Take the 11 middle coefficients of these normalized cross correlations $c_{x1xn}(-5...+5)$ to regard a maximum delay of 5 samples between the channels. If any of these coefficients is greater than a threshold, which is fixed at 0.99, describing the correlation between the channels, `IsOriginalMonoType` is set to true. Otherwise `IsOriginalMonoType` is set to false.

This produces a single IsOriginalMonoType value.

6.7.4 Background Noise Level

The `BackgroundNoiseLevelType` describes the noise level in an `AudioSegment`. In order to be independent of level scaling a normalization to the signal level is recommended. This value is computed for all channels separately. Furthermore, it is recommended to extract the `BackgroundNoiseLevelType` for the complete `AudioSegment` only, in order to get a summary description.

6.7.4.1 Syntax

```
<!-- ################################################ -->
<!--    Definition of BackgroundNoiseLevel D          -->
<!-- ################################################ -->
<complexType name="BackgroundNoiseLevelType">
  <complexContent>
     <extension base="mpeg7:AudioLLDVectorType"/>
  </complexContent>
</complexType>
```

6.7.4.2 Semantics

Name	Definition
`BackgroundNoiseLevelType` (BNL)	The `BackgroundNoiseLevelType` describes the noise level in an `AudioSegment`. Unit: [dB] Range: [-∞, 0]

6.7.4.3 Extraction (Informative)

To extract the Background-Noise-Level for an N-channel signal the following method is suggested:

For every channel calculate the absolute maximum maxPeak_dB, then divide the signal into blocks of 5ms, and find the block with the minimum mean power minPow_dB. The difference between these two values is the Background-Noise-Level for this channel. The normalization to the maximum amplitude is calculated to be independent of the recording level of the signal.

For n = 1..N

maxPeak_dB = $20 \log_{10} (\max(\text{abs}(x_n)))$

minPow_dB = $10 \log_{10} (\min(b_n))$

$BNL_n = minPow\_dB_n - maxPeak\_dB_n$

where

x_n is the signal of the n-th channel

b_n is the power of the 5ms blocks of the n-th channel with

$$b_n(j) = \frac{1}{bs} \sum_{k=(j-1)bs+1}^{j \cdot bs} x_n^2(k)$$ for the j-th block and the blocksize bs = $f_s \cdot 5ms$

This gives N values, which are stored in the `AudioLLDVectorType`, as a summary of one `AudioSegment`..

6.7.5 CrossChannelCorrelation

The normalized CrossChannelCorrelation of a multi channel signal is a measurement of the relationship between the first channel and the N-1 other channels, independent of their levels. The CrossChannelCorrelation r ranges between –1 and +1 (1 = completely correlated, 0 = uncorrelated, -1 = out of phase). In the case of two sine signals, the CrossChannelCorrelation is defined as r = cos φ, with φ as the phase shift between the two sine signals. This value indicates mono compatibility of multi channel signals.It is recommended to extract the `CrossChannelCorrelationType` for the complete `AudioSegment` only, in order to get a summary description.

6.7.5.1 Syntax

```
<!-- ################################################ -->
<!--   Definition of CrossChannelCorrelation D        -->
<!-- ################################################ -->
<complexType name="CrossChannelCorrelationType">
  <complexContent>
    <extension base="mpeg7:AudioLLDVectorType"/>
  </complexContent>
</complexType>
```

6.7.5.2 Semantics

Name	Definition
CrossChannelCorrelationType	The CrossChannelCorrelationType describes the correlation between two or more channels of an AudioSegment. Size: N-1 Unit: [-] Range: [-1, 1] Where N is the number of channels.

6.7.5.3 Extraction (informative)

To extract the CrossChannelCorrelation for an N-channel signal, the following method is suggested:

Normalize each channel to its maximum value. The following calculation has to be done N-1 times for an N-channel signal. Calculate the cross correlation c_{x1xn} between the first channel and the N-1 other channels, and normalize them all to the geometric mean maximum of the first channel's autocorrelation c_{x1x1} and the n-th channel's autocorrelation c_{xnxn}, which is defined as sqrt($c_{x1x1}(0)$ $c_{xnxn}(0)$). Take the middle coefficient of these normalized cross correlations as Correlation r.

For n = 2..N

$$r_{n-1} = \frac{c_{x1xn}(0)}{\sqrt{c_{x1x1}(0)c_{xnxn}(0)}}$$

This produces N-1 values, which are stored in the AudioLLDVectorType, as a summary of one AudioSegment.

6.7.6 Relative Delay

The RelativeDelayType describes the relative delay between the first channel and the N-1 other channels of an AudioSegment. For mono signals this value is zero. In order to be independent of the sampling frequency, all values are given in milliseconds [ms]. Furthermore, the relative delay measurement is restricted to ±0.5ms, in order to prevent ambiguity with pitch or other correlations in the signal. It is recommended to extract the RelativeDelayType for the complete AudioSegment only, in order to get a summary description.

6.7.6.1 Syntax

```
<!-- ################################################ -->
<!--   Definition of RelativeDelay D                  -->
<!-- ################################################ -->
<complexType name="RelativeDelayType">
  <complexContent>
    <extension base="mpeg7:AudioLLDVectorType">
        <attribute name="Reliability" type="mpeg7:zeroToOneType"
              use="optional"/>
    </extension>
  </complexContent>
</complexType>
```

6.7.6.2 Semantics

Name	Definition
`RelativeDelayType`	The `RelativeDelayType` describes the relative delay between two or more channels of an `AudioSegment`. Size: N-1 Unit: [ms] Range: [-0.5, 0.5] Where N is the number of channels. The confidence attribute of the `AudioLLDVectorType` may be used to indicate the reliability of the estimates
`Reliability`	The Reliability attribute describes the estimation quality of the `RelativeDelayType`. `Reliability` is of type `mpeg7:zeroToOne`.

6.7.6.3 Extraction (informative)

To extract the relative delay for an N-channel signal the following method is suggested:

The calculation has to be done N-1 times for an N-channel signal. Calculate the cross correlation c_{xy} between the first channel and the N-1 other channels. The unscaled cross correlation c_{xy} between two signals x and y is defined as:

$$c_{xy}(m) = \begin{cases} \sum_{k=0}^{K-|m|-1} x(k)\,y(k+m) & m \geq 0 \\ c_{xy}(-m) & m < 0 \end{cases}$$

If the input signals have the length K, then the cross correlation has the length 2K-1.

To calculate the relative delay find the position of the maximum of the cross correlation in the search region corresponding to ±0.5ms for the current sampling frequency. The `RelativeDelayType` for each channel is then estimated by taking the time difference corresponding to the position of the maximum (position_of_maximum) and the position of the middle coefficient (0).

`RelativeDelay` = (position_ of_maximum) / f_s

where f_s is the sample frequency

This produces N-1 values, which are stored in the `AudioLLDVectorType`, as summary values for one AudioSegment.

The reliability of the estimation is given by the magnitude squared normalized cross-correlation for this maximum value, given by

$$MSC_{n-1}(\max) = \frac{\left|c_{x1xn}(\max)\right|^2}{c_{x1x1}(\max)c_{xnxn}(\max)}$$

6.7.7 Balance

The BalanceType describes the relative levels between the first channel and the N-1 other channels. For mono signals this value is zero. It is recommended to extract the `BalanceType` for the complete `AudioSegment` only, in order to get a summary description.

6.7.7.1 Syntax

```
<!-- ################################################ -->
<!--   Definition of Balance D                        -->
<!-- ################################################ -->
<complexType name="BalanceType">
  <complexContent>
    <extension base="mpeg7:AudioLLDVectorType"/>
  </complexContent>
</complexType>
```

6.7.7.2 Semantics

Name	Definition
`BalanceType`	The `BalanceType` describes the relative level between two or more channels of an `AudioSegment`. Size: N-1 Unit: [dB] Range: [-100, 100] Where N is the number of channels.

6.7.7.3 Extraction (informative)

The balance B for an N-channel signal may be extracted using the following algorithm:

Calculate the mean power for each channel and then compute the relation between the first channel power and the power of the N-1 other channels in dB.

For n = 2..N

$B_{n-1} = 10*\log_{10}(\text{meanpower\_ch\_1}/\text{meanpower\_ch\_n})$

where

meanpower_ch_1 is the mean power of the first channel = $\frac{1}{K}\sum_{k=0}^{K} x_1^2(k)$

meanpower_ch_n is the mean power of the n-th channel = $\frac{1}{K}\sum_{k=0}^{K} x_n^2(k)$

This produces N-1 values which are stored in the `AudioLLDVectorType`.

6.7.8 DC Offset

The DcOffsetType describes the mean relative to the maximum of each channel of an AudioSegment. Audio signals should have a zero mean. A DC-Offset can indicate bad analog-digital conversion. In order to be independent of level scaling a normalization to the signal level is recommended. It is recommended to extract the DcOffsetType for the complete AudioSegment only, in order to get a summary description.

6.7.8.1 Syntax

```
<!-- ################################################ -->
<!--   Definition of DcOffset D                       -->
<!-- ################################################ -->
<complexType name="DcOffsetType">
  <complexContent>
    <extension base="mpeg7:AudioLLDVectorType"/>
  </complexContent>
</complexType>
```

6.7.8.2 Semantics

Name	Definition
DcOffsetType	The DcOffsetType describes the mean relative to the maximum of each channel of an AudioSegment. Audio signals should have zero mean. Size: N Unit: [-] Range: [-1, 1] Where N is the number of channels.

6.7.8.3 Extraction (informative)

The DC-Offset DC for an N-channel signal may be extracted using the following algorithm:

Calculate the mean amplitude for each channel and normalize it to the maximum of the absolute magnitude value of each channel.

For n = 1..N

$DC_n = mean(x_n)/max(abs(x_n))$

This produces N values, which are stored in the AudioLLDVectorType.

6.7.9 Bandwidth

The BandwidthType describes an estimate of the original signal bandwidth for each channel. This value is an indicator of the technical quality of the original recording. Due to errors such as clicks, some robustness issues have to be considered. Figure AMD1-3 shows a typical spectrogram for a short segment (2s) of an old recording.

The bandwidth of the original signal is just 4 kHz, since the higher frequency components are connected to clicks and background noise. It is recommended to extract the BandwidthType for the complete AudioSegment only, in order to get a summary description.

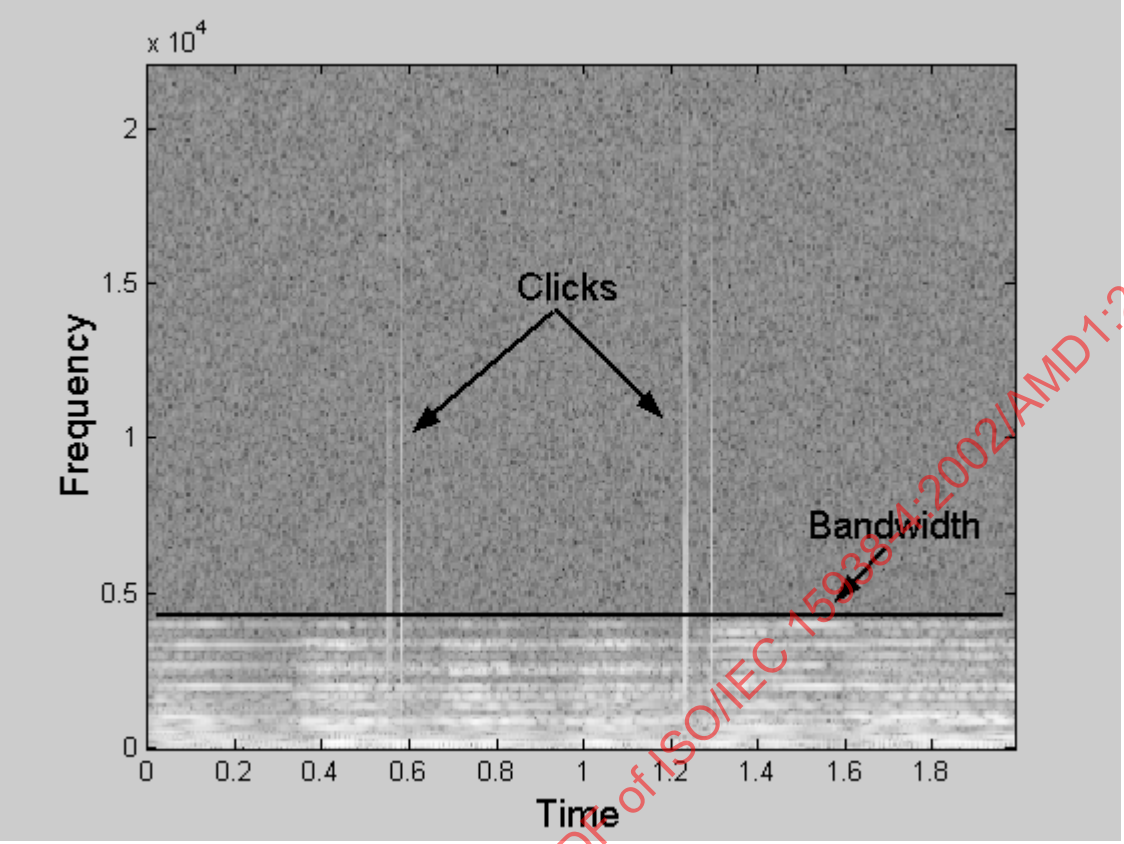

Figure AMD1-3 - Spectrogram of a degraded audio file

6.7.9.1 Syntax

```
<!-- ################################################ -->
<!--   Definition of Bandwidth D                      -->
<!-- ################################################ -->
<complexType name="BandwidthType">
  <complexContent>
    <extension base="mpeg7:AudioLLDVectorType"/>
  </complexContent>
</complexType>
```

6.7.9.2 Semantics

Name	Definition
BandwidthType	The BandwidthType describes the upper limit of the signal's bandwidth for each channel. The lower limit is set to 0 Hz. Size: N Unit: [Hz] Range: [0,sr/2), that is not including sr/2 Where sr stands for sampling rate. Where N is the number of channels.

6.7.9.3 Extraction (informative)

To extract the `BandwidthType` for an N-channel signal, the following method is suggested. However, this will not lead to accurate results under all conditions.

The calculation of the bandwidth is similar to the calculation of the `AudioSpectrumEnvelopeType` (see extraction of `AudioSpectrumEnvelopeType` in ISO/IEC 15938-4 subclause 5.3.7.3.4). The audio segment is divided into parts 30ms in length, separated by 10ms gaps. The power spectrum is then calculated for each part using a Hamming window and a 2048 point FFT (see extraction of `AudioSpectrumEnvelopeType`). Following this, a maximum filter is used over the spectra of the 30ms parts to get a maximum power spectrum for each channel. After that the logarithmic maximum power spectrum (LMPS) is calculated.

LMPS = 10 $\log_{10}$ (maximum power spectrum)

The method to find the upper limit of the LMPS is shown in Figure AMD1-4. A boundary is used to find the edge of the bandwidth of the LMPS for each channel. Therefore, the maximum and the minimum of the LMPS is calculated (see Figure AMD1-4). The boundary for the upper limit of the bandwidth is set to 70% below the maximum of the logarithmic maximum power spectrum, corresponding to the range between maximum and minimum. The upper edge of the bandwidth is set as the point where the power spectrum falls below the border for the last time.

Note: For signals which have been through some source coding (e.g. MP3) this extraction method will not work properly.

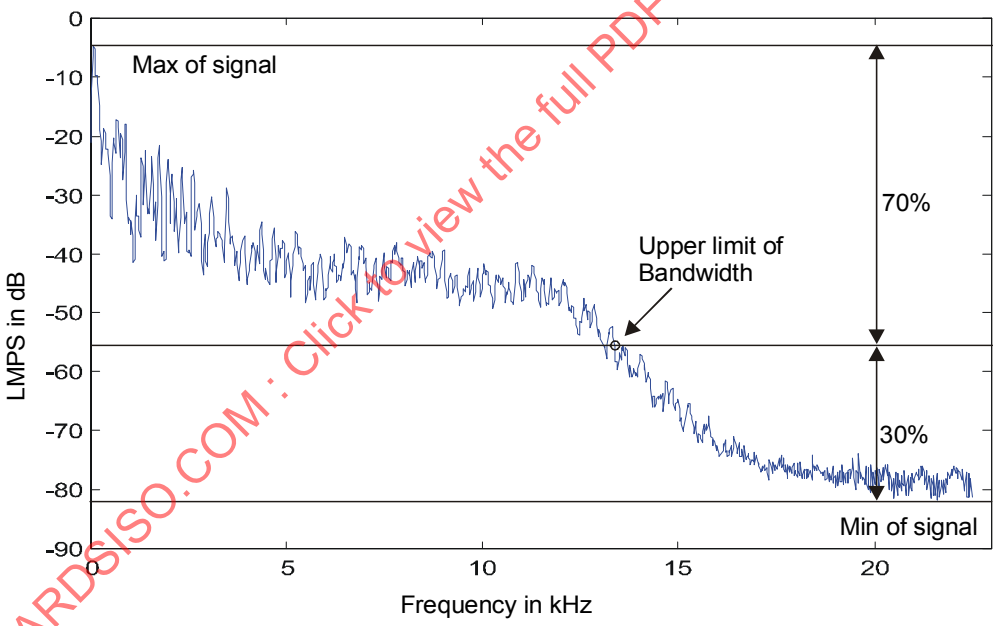

Figure AMD1-4 - Calculation of upper limit in bandwidth

6.7.10 Transmission Technology

The `TransmissionTechnologyType` describes the technology in which the audio file was transmitted or recorded. For the purpose of the migration of analogue sound material e.g. stored on wax-cylinder, shellac or vinyl, ten categories of transmission and recording technologies are defined in the `TransmissionTechnologyCS` classification scheme.

This value has to be set by an operator manually, since knowledge about the transfer process is necessary.

6.7.10.1 Syntax

```
<!-- ################################################ -->
<!--   Definition of Transmission Technology Type     -->
<!-- ################################################ -->
<complexType name="TransmissionTechnologyType">
  <complexContent>
    <extension base="mpeg7:ControlledTermUseType"/>
  </complexContent>
</complexType>
```

6.7.10.2 Semantics

Name	Definition
`TransmissionTechnologyType`	The `TransmissionTechnologyType` describes the technology in which the audio file was transmitted or recorded using the `TransmissionTechnologyCS` classification scheme (Annex B).

6.7.10.3 Extraction (informative)

In order to describe the `TransmissionTechnologyType` the `TransmissionTechnologyCS` Classification Scheme can be used. This Classification Scheme contains 10 categories with different transmission types and recording technologies. Although the transmission or recording technology is not directly linked to the signal quality, or related to the time of recording, it can be used, and should be used, to classify similar quality of signals to one category. Therefore, different kinds of technology are packed together in order to define one category. The editor of the `TransmissionTechnology` descriptor should be familiar with the different transmission or recording technologies, and should manually choose a suitable category. To determine the category it is necessary to know the weakest link from the signal transmission path. For example, a shellac recording from 1927 republished on SACD belongs to category 3, and not to category 9. However, by using high-end restoration software it may be possible to enhance the final quality, and therefore, category 4 or 5 is the right choice.

To give a rough overview the following examples with well-known audio standards are suggested for the different categories.

Category 0: - no specified transmission or recording technology

Category 1: - wax cylinder record recordings

Category 2: - shellac recordings before 1925

Category 3: - shellac recordings after 1925

movie sound recordings of the 1930s

Category 4: - speech over telephone with bandwidth 300Hz to 3.4kHz (bad quality)

Category 5: - speech over telephone with bandwidth 300Hz to 3.4kHz (good quality)

AM radio (bad quality)

Category 6: - speech over telephone with bandwidth 50Hz to 8kHz (ISDN, bad quality)

vinyl before 1960 (bad quality)

Category 7: - speech over telephone with bandwidth 50Hz to 8kHz (ISDN, good quality)

movie sound recordings of the 1960s

vinyl before 1960 (good quality)

vinyl after 1960 (bad quality)

FM radio (bad quality)

Category 8: - vinyl after 1960 (good quality)

FM radio (good quality)

Category 9: - PCM & ΣΔ modulated digital data

e.g.:

CD, DVD-A, SACD

6.7.11 Error Event Type

The `ErrorEventType` is used to describe errors in an `AudioSegment`.

6.7.11.1 Syntax

```
<!-- ################################################ -->
<!--   Definition of ErrorEvent DS                    -->
<!-- ################################################ -->
<complexType name="ErrorEventType">
  <complexContent>
    <extension base="mpeg7:AudioDSType">
      <sequence>
        <element name="ErrorClass" type="mpeg7:ControlledTermUseType" />
        <element name="ChannelNo" type="positiveInteger" />
        <element name="TimeStamp" type= "mpeg7:MediaTimeType" />
        <element name="Relevance" type="mpeg7:unsigned4" />
        <element name="DetectionProcess">
          <simpleType>
            <restriction base="string">
              <enumeration value="manual"/>
              <enumeration value="automatic"/>
            </restriction>
          </simpleType>
        </element>
        <element name="Status">
          <simpleType>
            <restriction base="string">
              <enumeration value="undefined"/>
              <enumeration value="checked"/>
              <enumeration value="needs restoration"/>
              <enumeration value="restored"/>
              <enumeration value="deleted"/>
            </restriction>
          </simpleType>
        </element>
        <element name ="Comment" type ="mpeg7:TextAnnotationType"/>
      </sequence>
    </extension>
    </complexContent>
</complexType>
```

6.7.11.2 Semantics

Name	Definition
ErrorEventType	The ErrorEventType describes the event time of a specified error type in the signal.
ErrorClass	The ErrorClass describes the error type using the ErrorClassCS classification scheme (Annex B).
ChannelNo	This element specifies the channel in which an error occurs.
TimeStamp	This element specifies the temporal location of an error.
Relevance	Describes the relevance of an error. Possible values: 0 = Not specified, 1 = low to 7 = high An error with low Relevance (value = 1) is hardly audible, and an error with high Relevance (value = 7) is very disturbing. In the case of a SampleHold for example, an error with 5 successive samples with the same value has more relevance than an error with two successive samples with the same value.
DetectionProcess	Describes the process of detection. Possible entries: Manual, Automatic
Status	Describes the status of an error. The entry of the Status is set automatically or by a listener, its default entry is “Undefined”. After a check of the error the Status may be set to several other entries like “checked” which denotes that the error has been checked; “needs restoration” denotes that the error needs to be restored; “restored” denotes that the error has been restored and “deleted” denotes that the detected error was a false alarm. Possible entries: Undefined, checked, needs restoration, restored, deleted
Comment	The Comment contains any comment about the detected error

6.7.11.3 Event Description and Extraction (informative examples)

6.7.11.3.1 Click

6.7.11.3.1.1 Description

A Click is a pulse shaped error which can occur e.g. when an audio signal originates from an old gramophone record. The occurrence of this error is stochastic. In the frequency domain it can be seen that the main power of this error lies in the higher frequencies of the spectrum. The Click value includes the channel and the position of this error.

6.7.11.3.1.2 Extraction (informative)

Click detection is a very challenging problem, and has not yet been completely resolved. The following method is suggested for finding clicks in a signal.

Firstly, a high pass filter is used in order to suppress the part of the spectrum where the main power of the audio signal lies. The absolute value is then taken.

d(k) = abs (Highpass (s(k)))

where s(k) is the original signal, and d(k) is the filtered signal.

Following this, a median filter is used to smooth the signal, in order to get a robust estimation of the energy in the high frequencies.

m(k) = median (d(k))

For a further smoothing, a mean filter is used on the computed signal that calculates the mean of the median filtered signal over short blocks.

r(k) = meanfilter (m(k))

To determine the position of the Clicks, the calculated signal r(k) is weighted with a threshold *Thresh* and then compared to the filtered signal d(k). The position where d(k) is greater than *Thresh* * r(k) is the position of a Click.

Clickpos = d(k) > Thresh * r(k)

All clicks that occur within a range of 200 samples are taken as one click, with the position taken as the first occurrence of these clicks.

6.7.11.3.2 Digital Clip

6.7.11.3.2.1 Description

`DigitalClip` is an error, which occurs when a signal is clipped. In this case two or more subsequent samples have a value of ±1. The `DigitalClip` includes the channel, the position and the duration of this error.

6.7.11.3.2.2 Extraction (informative)

To find `DigitalClips` in a signal, two or more successive samples with a value of ±1 must be found. The position of the first of these samples is taken as the position of the `DigitalClip`. The number of the subsequent samples with a value of ±1 is the duration of the `DigitalClip`.

6.7.11.3.3 Sample & Hold

6.7.11.3.3.1 Description

`SampleHold` is an error which occurs when three or more successive samples have the same value. The `SampleHold` includes the channel, the position and the duration of this error.

6.7.11.3.3.2 Extraction (informative)

To find `SampleHolds` in a signal, three or more successive samples with the same value must be found. The position of the first of these samples is taken as the position of the `SampleHold`. The number of the subsequent samples with the same value is the duration of the `SampleHold`.

6.7.11.3.4 Digital Zero

6.7.11.3.4.1 Description

`DigitalZero` describes an error which occurs when two or more subsequent samples have the value 0. `DigitalZero` includes the channel, the position and the duration of this error.

6.7.11.3.4.2 Extraction (informative)

To find `DigitalZeros` in a signal, two or more successive samples with a value of 0 must be found. The position of the first of these samples is the position of the `DigitalZero`. The number of the subsequent samples with a value of 0 is taken as the duration of the `DigitalZero`.

6.7.12 Example

The following example is an instantiation of the `AudioSignalQualityDS`. All low level descriptors are instantiated as summary results for a complete `AudioSegment`.

```
   <DescriptionUnit xsi:type="AudioSignalQualityType" IsOriginalMono ="false">
      <Operator>
         <Name>
            <GivenName>Joerg Bitzer</GivenName>
         </Name>
      </Operator>
      <UsedTool>
         <Tool>
            <Name>
               Quadriga TapeModul 1.1.0
            </Name>
         </Tool>
      </UsedTool>
      <BackgroundNoiseLevel channels="1 2">
         <Vector>
            -37.2281     -37.6493
         </Vector>
      </BackgroundNoiseLevel>
      <RelativeDelay channels="1 2">
         <Vector>
            -0.29478
         </Vector>
      </RelativeDelay>
      <Balance channels="1 2">
         <Vector>
            -4.9338
         </Vector>
      </Balance>
      <DcOffset channels="1 2">
         <Vector>
            0.012083    0.010522
         </Vector>
      </DcOffset>
      <CrossChannelCorrelation channels="1 2">
         <Vector>
            0.32628
         </Vector>
      </CrossChannelCorrelation>
      <Bandwidth channels="1 2">
         <Vector>
            7988.8184      5878.5645
         </Vector>
      </Bandwidth>
      <TransmissionTechnology
href="urn:mpeg:mpeg7:cs:TransmissionTechnologyCS:category0">
         <Name>Category0</Name>
      </TransmissionTechnology>
      <ErrorEventList>
         <ErrorEvent>
```

```
            <ErrorClass href="urn:mpeg:mpeg7:cs:ErrorClassCS:digitalzero">
               <Name>DigitalZero</Name>
            </ErrorClass>
            <ChannelNo>2</ChannelNo>
            <TimeStamp>
               <MediaRelIncrTimePoint mediaTimeUnit="PT1N44100F"
mediaTimeBase="../../MediaLocator[1]">1</MediaRelIncrTimePoint>
               <MediaIncrDuration
mediaTimeUnit="PT1N44100F">13</MediaIncrDuration>
            </TimeStamp>
            <Relevance>1</Relevance>
            <DetectionProcess>automatic</DetectionProcess>
            <Status>undefined</Status>
            <Comment>
               <FreeTextAnnotation>
                  any Comment to ErrorEvent
               </FreeTextAnnotation>
            </Comment>
         </ErrorEvent>
      </ErrorEventList>
   </DescriptionUnit>
```

Add a new subclause 6.8:

6.8 Audio Tempo

6.8.1 Introduction

The musical tempo is a higher level semantic concept to characterize the underlying temporal structure of musical material. Musical tempo information may be used as an efficient search criterion to find musical content for various purposes (e.g. dancing) or belonging to certain musical genres.

AudioTempo describes the tempo of a musical item according to standard musical notation. Its scope is limited to describing musical material with a dominant musical tempo and only one tempo at a time. The tempo information consists of two components:

- The frequency of beats is expressed in units of beats per minute (bpm) by AudioBPMType.
- The meter defines the unit of measurement of beats (whole note, half-note, quarter-note, dotted quarter note etc.) and is described using MeterType. Please note that, although MeterType has been initially defined in a different context, it is used here to represent the unit of measurement of beats in a more flexible way, thus allowing to also express non-elementary values (e.g. dotted half-note). By combining Bpm and Meter the information about the musical tempo is expressed in terms of standard musical notation. Figure AMD1-5 illustrates this aspect with the help of a musical example score that contains all needed information for a musician to interpret the notation correctly. The Meter is assigned to a quarter note and the Tempo to 120 Bpm (thus resulting in an equivalent of 120 quarter notes per minute).

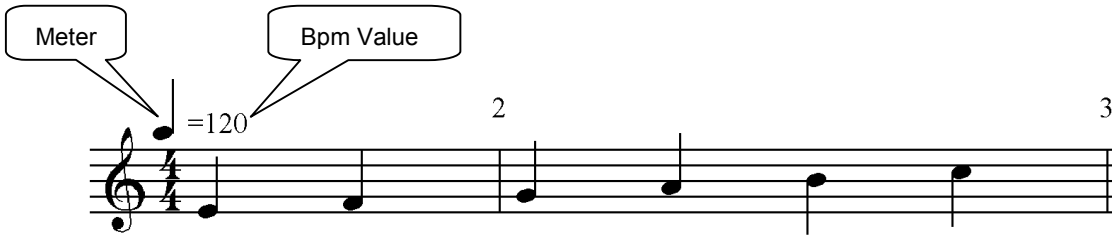

Figure AMD1-5 - Example of a musical score

6.8.2 Audio TempoType

6.8.2.1 Syntax

```
<!-- ################################################ -->
<!--   Definition of Audio Tempo DS                   -->
<!-- ################################################ -->
<complexType name="AudioTempoType">
  <complexContent>
    <extension base="mpeg7:AudioDSType">
      <sequence>
        <element name="BPM" type="mpeg7:AudioBPMType"/>
        <element name="Meter" type="mpeg7:MeterType" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

6.8.2.2 Semantics

Name	Definition
`AudioTempoType`	A structure describing musical tempo information
`BPM`	The bpm (beats per minute) information of the audio signal of type `AudioBPMType`.
`Meter`	The information of the current unit of measurement of beats in `MeterType`

6.8.3 Audio BPMType

AudioBPMType describes the frequency of beats of an audio signal representing a musical item in units of beats per minute (bpm).

6.8.3.1 Syntax

```
<!-- ################################################################ -->
<!-- Definition of AudioBPMType                                       -->
<!-- ################################################################ -->
<complexType name="AudioBPMType">
  <complexContent>
    <extension base="mpeg7:AudioLLDScalarType">
      <attribute name="loLimit" type="float" use="optional"/>
      <attribute name="hiLimit" type="float" use="optional"/>
    </extension>
  </complexContent>
</complexType>
```

6.8.3.2 Semantics

Name	Definition
AudioBPMType	The bpm (beats per minute) information of the audio signal of type AudioBPMType
loLimit	Indicates the smallest valid bpm value for this description and defines the corresponding limit for an extraction mechanism calculating the bpm information.
hiLimit	Indicates the biggest valid bpm value for this description and defines the lower limit for an extraction mechanism calculating the bpm information.

A default hopSize of 2 seconds (PT2000N1000F) is assumed.

6.8.3.3 Usage, extraction and examples (informative)

6.8.3.3.1 Purpose

The AudioBPMType describes the frequency of beats of an audio signal representing musical content. The beat frequency information is given in units of beats per minute (bpm), together with optional weights indicating the reliability of this measurement.

6.8.3.3.2 Extraction

The extraction of the beat frequency can be implemented by many different algorithms. One illustrative example is described here and comprises several steps:

The incoming signal is decomposed and pre-processed in a number of spectral bands. As an example, a split into 6 frequency bands may be used with transition frequencies of 200 Hz, 400 Hz, 800 Hz, 1600 Hz, and 3200 Hz, respectively. The following processing steps are carried out for each frequency band:

1. The band limited signal is derived from the input signal by means of bandpass filtering (lowpass filtering for the first frequency band, highpass filtering for the last frequency band).

2. The band limited signal is two-way rectified (i.e. the absolute values are taken) and smoothed over time with a time constant around 100 ms to calculate an envelope signal. At this point, the signal may be decimated in order to reduce computational complexity.

3. The envelope signal is differentiated (i.e. the differences between subsequent samples are calculated) and the result is limited to non-negative values, thus corresponding to the onset portions of the signal. Each differentiated envelope is normalized by its maximum value.