INTERNATIONAL STANDARD

ISO/IEC 23005-5

Fourth edition
2019-02

Information technology — Media context and control —

Part 5: Data formats for interaction devices

Technologies de l'information — Contrôle et contexte de supports —
Partie 5: Formats des données pour dispositifs d'interaction

Reference number
ISO/IEC 23005-5:2019(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

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.

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

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. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

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

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

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

This fourth edition cancels and replaces the third edition (ISO/IEC 23005-5:2016), which has been technically revised.

The main changes compared to the previous edition are as follows:

— added new device command type of ThreeDPrinterType;

— added new device command type of SoundDisplaySettingType;

— added new device command type of ThreeDPrintingColorReproductionType;

— added new device command type of ArrayLightType;

— added new sensed information type of RADARSensorType;

— added new sensed information type of RADARSensorType;

— added new sensed information type of ArrayCameraType;

— added new sensed information type of MicrophoneSensorType;

— added new sensed information type of E-NoseSensorType.

A list of all parts in the ISO 23005 series can be found on the ISO website.

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

Introduction

The ISO/IEC 23005 series provides an architecture and specifies information representation of data flowing in and out of the real world and virtual worlds.

The data for the real world are communicated through sensors and actuators. The data for virtual worlds consist of properties of virtual objects and multi-sensorial data embedded in audio-visual content. The ISO/IEC 23005 series specifies data formats for sensors, actuators, virtual objects, and audio-visual content.

Data captured from the real world may need to be adapted for use in a virtual world and data from virtual worlds may also need to be adapted for use in the real world. The standard does not specify how the adaptation is carried out but only specifies the interfaces.

Data for sensors are sensor capabilities, sensed data, and sensor adaptation preferences.

Data for actuators are sensory device capabilities, sensory device commands, and sensory effect preferences.

Data for virtual objects are characteristics of avatars and virtual world objects.

Data for audio-visual content are sensory effects.

This document contains the tools for exchanging information for interaction devices. To be specific, it specifies normative command formats for controlling actuators (e.g., actuator commands for sensory devices) and data formats for receiving information from sensors (e.g., sensed information from sensors) as illustrated as the yellow boxes in Figure 1. It also specifies some non-normative examples. The adaptation engine is not within the scope.

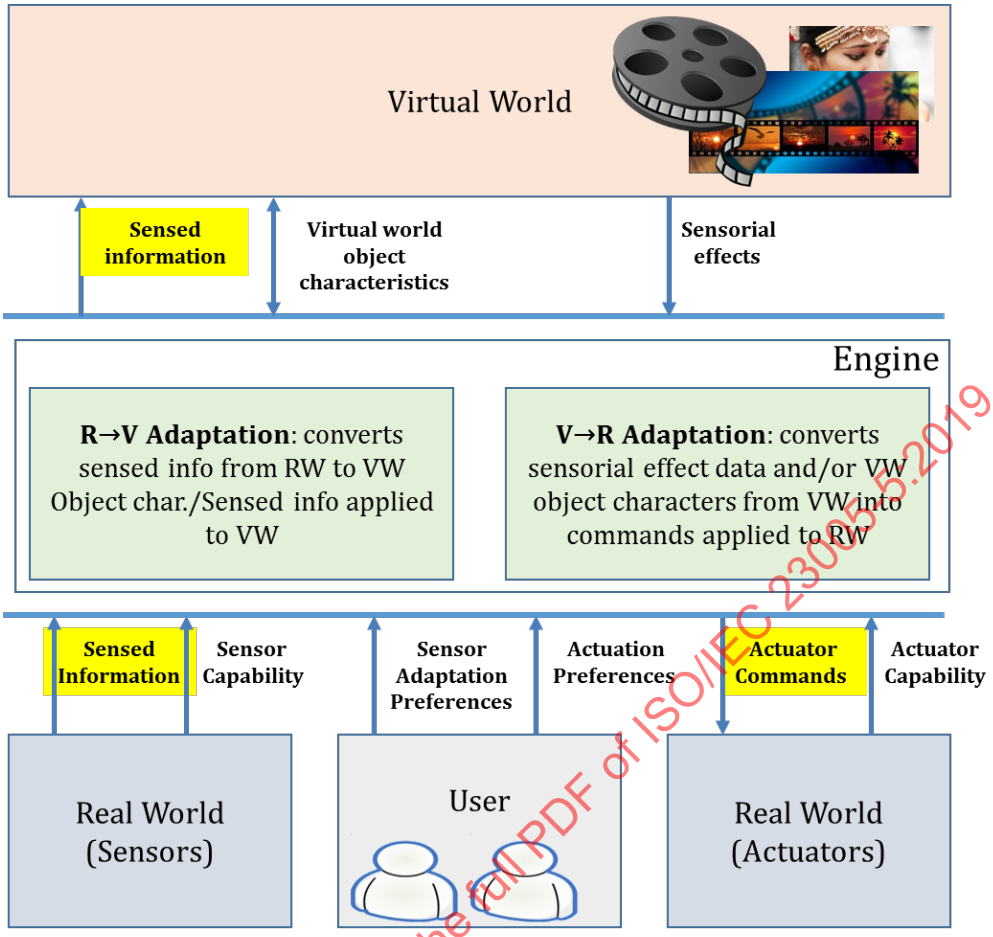


Figure 1 — Scope of the data formats for interaction devices

When this document is used, the adaptation engine (RV or VR engine), which is not within the scope of standardization, performs bi-directional communications using data formats specified in this document. The adaptation engine can also utilize other tools defined in ISO/IEC 23005-2, which are user's sensory preferences (USP), sensory device capabilities (SDC), sensor capabilities (SC), and sensor adaptation preferences (SAP) for fine controlling devices in both real and virtual worlds.

The International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of patents.

ISO and the IEC take no position concerning the evidence, validity and scope of these patent rights. The holders of these patent rights have assured ISO and the IEC that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statements of the holders of these patent rights are registered with ISO and the IEC. Information may be obtained from the companies listed below.

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

ISO (www.iso.org/patents) and IEC (http://patents.iec.ch) maintain online databases of patents relevant to their standards. Users are encouraged to consult the databases for the most up to date information concerning patents.

Company	Address
Samsung Electronics Co.Ltd.	416, Maetan-dong, Yeongtong-gu, Suwon-si, Gyeonggi-do, 152-848, Korea
Gwangju Institute of Science and Technology	261 Cheomdan - gwagiro (Oryong-dong), Buk-gu, Gwangju 500-712, Korea
Electronics and Telecommunications Research Institute (ETRI)	218 Gajeongno, Yuseong-gu, Daejeon, 305-700, Korea
Konkuk University	1 Hwayang-dong, Gwangjin-gu, Seoul, 143-701, Korea

Information technology — Media context and control —

Part 5:
Data formats for interaction devices

1 Scope

This document specifies syntax and semantics of the data formats for interaction devices by providing a standardized format for interfacing actuators and sensors by defining XML schema-based language named Interaction Information Description Language (IIDL). IIDL provides a basic structure with common information for communication with various actuators and sensors in consistency. Device Command Vocabulary (DCV) is defined to provide a standardized format for commanding individual actuator, and Sensed Information Vocabulary (SIV) is defined to provide a standardized format for holding information from individual sensors either to get environmental information from real world or to influence virtual world objects using the acquired information on the basis of IIDL.

2 Normative references

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

ISO/IEC 15938-5:2003, *Information technology — Multimedia content description interface — Part 5: Multimedia description schemes*

ISO/IEC FDIS 23005-6:—[1], *Information technology — Media context and control — Part 6: Common types and tools*

ISO/IEC 10646, *Information technology — Universal Coded Character Set (UCS)*

3 Terms, definitions and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 23005-6 apply.

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

— IEC Electropedia: available at http://www.electropedia.org/

— ISO Online browsing platform: available at https://www.iso.org/obp

3.2 Abbreviated terms

CS classification scheme (see ISO/IEC 15938-5)

DAC digital-to-analog conversion

[1] Fourth Edition under preparation. Stage at time of publication: ISO/IEC FDIS 23005-6:2018.

RV real to virtual

URI Uniform Resource Identifier (see RFC 2396)

VR virtual to real

XML Extensible Markup Language (W3C, http://www.w3.org/XML/)

RW real world

VW virtual world

4 Interaction information description language

4.1 General

This Clause describes basic structure of the tools in this document in the form of interaction information description language including the schema wrapper conventions, basic data types, root element, and top-level elements.

4.2 Schema wrapper conventions

The syntax defined in this Clause assumes the following schema wrapper to form a valid XML schema document.

```
<schema xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004" xmlns:iidl="urn:mpeg:mpeg-v:2018:01-
IIDL-NS" xmlns:mpegvct="urn:mpeg:mpeg-v:2018:01-CT-NS" xmlns:dcv="urn:mpeg:mpeg-
v:2018:01-DCV-NS" xmlns:siv="urn:mpeg:mpeg-v:2018:01-SIV-NS"
targetNamespace="urn:mpeg:mpeg-v:2018:01-IIDL-NS" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="ISO/IEC 23005-5" id="MPEG-V-
IIDL.xsd">
   <import namespace="urn:mpeg:mpeg7:schema:2004"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
7_schema_files/mpeg7-v2.xsd"/>
   <import namespace="urn:mpeg:mpeg-v:2018:01-CT-NS" schemaLocation="
http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-CT.xsd"/>
   <import namespace="urn:mpeg:mpeg-v:2018:01-DCV-NS"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-DCV.xsd"/>
   <import namespace="urn:mpeg:mpeg-v:2018:01-SIV-NS"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-SIV.xsd"/>
```

Additionally, the following line should be appended to the resulting schema document in order to obtain a well-formed XML document.

```
</schema>
```

For clarity, throughout this document, consistent namespace prefixes are used.

"`xsi:`" prefix is not normative. It is a naming convention in this document to refer to an element of the `http://www.w3.org/2001/XMLSchema-instance` namespace.

"xml:" and "xmlns:" are normative prefixes defined in Reference [10]. The prefix "xml:" is by definition bound to "http://www.w3.org/XML/1998/namespace". The prefix "xmlns:" is used only for namespace bindings and is not itself bound to any namespace name.

All other prefixes used in either the text or examples of this specification are not normative, e.g., "sedl:", "sev:", "dia:", "si:", "mpeg7:".

In particular, most of the informative examples in this specification are provided as XML fragments without the normally required XML document declaration and, thus, miss a correct namespace binding context declaration. In these descriptions fragments the different prefixes are bound to the namespaces as given in Table 1.

The XML schema defined in this standard is available through appendix 1.

Table 1 — Mapping of prefixes to namespaces in examples and text

Prefix	Corresponding namespace
mpegvct	urn:mpeg:mpeg-v:2018:01-CT-NS
iidl	urn:mpeg:mpeg-v:2018:01-IIDL-NS
dcv	urn:mpeg:mpeg-v:2018:01-DCV-NS
sedl	urn:mpeg:mpeg-v:2018:01-SEDL-NS
sev	urn:mpeg:mpeg-v:2018:01-SEV-NS
dia	urn:mpeg:mpeg21:2003:01-DIA-NS
Si	urn:mpeg:mpeg21:2003:01-DIA-XSI-NS
mpeg7	urn:mpeg:mpeg7:schema:2004
xsi	http://www.w3.org/2001/XMLSchema-instance
xsd	http://www.w3.org/2001/XMLSchema

4.3 Root element and top-level tools

4.3.1 General

This subclause specifies the root elements and the top-level tools which can follow a root element in interactive information. The root elements are the only elements, one of which can appear as the topmost element when the interactive information specified in this document is instantiated. The top-level tools are defined as the elements which are allowed to appear as the topmost element within the root element.

4.3.2 Syntax

```
<!-- ################################################ -->
<!--  Root and Top-Level Elements                     -->
<!-- ################################################ -->
<element name="InteractionInfo" type="iidl:InteractionInfoType"/>
<element name="DeviceCommand" type="iidl:DeviceCommandBaseType"/>
<element name="SensedInfo" type="iidl:SensedInfoBaseType"/>

<complexType name="InteractionInfoType">
   <choice>
      <element name="DeviceCommandList" type="iidl:DeviceCmdListType"/>
      <element name="SensedInfoList" type="iidl:SensedInfoListType"/>
   </choice>
</complexType>

<complexType name="DeviceCmdListType">
   <sequence>
      <element ref="iidl:DeviceCommand" maxOccurs="unbounded"/>
   </sequence>
```

```
</complexType>

<complexType name="SensedInfoListType">
    <sequence>
        <element ref="iidl:SensedInfo" maxOccurs="unbounded"/>
    </sequence>
</complexType>
```

4.3.3 Binary representation syntax

InteractionInfo {	***Number of bits***	***Mnemonic***
InteractionType	1	bslbf
If (InteractionType){		
DeviceCommandList		DeviceCmdListType
}else{		
SensedInfoList		SensedInfoListType
}		
}		
SensedInfoListType{		
NumOfSensedInfo	32	uimsbf
for(i=1;i<NumOfSensedInfo;i++){		
IndividualSensedInfoType	8	bslbf
SensedInfo		SensedInfoType specified by IndividualSensedInfoType
}		
}		
}		
DeviceCmdListType{		
NumOfDeviceCmd	32	uimsbf
for(i=1;i<NumOfDeviceCmd;i++){		
IndividualDeviceCmdType	8	bslbf
DeviceCmd		DeviceCmdType specified by

		IndividualDeviceCmdType
}		
}		

4.3.4 Semantics

Semantics of the `InteractionInfo` type:

Name	*Definition*
`InteractionInfo`	One of the root elements that serve as the topmost element in the interaction information description. This element may have `DeviceCommandList` and `SensedInfoList` as its subelements.
`DeviceCommand`	One of the root elements that serve as the topmost element in the interaction information description. It specifies a single command for a certain device. This element can be instantiated as a root element or subelements of `DeviceCommandList`.
`SensedInfo`	One of the root elements that serve as the topmost element in the interaction information description. It specifies a single description of information acquired through a sensor. This element can be instantiated as a root element or subelements of `SensedInfoList`.
`InteractionInfoType`	The root type provides basic structure that the interaction information description should follow through the root element.
`DeviceCommandList`	Optional wrapper element that serves as the placeholder for the sequence of device commands.
`SensedInfoList`	Optional wrapper element that serves as the placeholder for the list of information acquired through sensors (`SensedInfo`).
`DeviceCommandBaseType`	`DeviceCommandBaseType` is an abstract type providing a base for individual command (`DeviceCommand`).
`SensedInfoBaseType`	`SensedInfoBaseType` is an abstract type providing a base for description of individual type of sensor.
`InteractionType`	This field, which is only present in the binary representation, indicates the type of the `InteractionInfo` element. If it is "1" then the `DeviceCommandList` element is present, otherwise the `SensedInfoList` element is present.
`SensedInfoListType`	A type that serves as the placeholder for the list of information acquired through sensors.
`NumOfSensedInfo`	This field, which is only present in the binary representation, specifies the number of `SensedInfo` instances accommodated in the `SensedInfoList`.

Name	*Definition*
`IndividualSensedInfoType`	This field, which is only present in the binary representation, describes which SenseInfo type shall be used. In the binary description, the following mapping table is used,

Terms of sensor	Binary representation for sensor type (8 bits)
Light sensor	00000000
Ambient noise sensor	00000001
Temperature sensor	00000010
Humidity sensor	00000011
Distance sensor	00000100
Atmospheric pressure sensor	00000101
Position sensor	00000110
Velocity sensor	00000111
Acceleration sensor	00001000
Orientation sensor	00001001
Angular velocity sensor	00001010
Angular acceleration sensor	00001011
Force sensor	00001100
Torque sensor	00001101
Pressure sensor	00001110
Motion sensor	00001111
Intelligent camera sensor	00010000
MultiInteraction point sensor	00010001
Gaze tacking sensor	00010010
Wind sensor	00010011
Global position sensor	00010100
Altitude sensor	00010101
Bend sensor	00010110
Gas sensor	00010111
Dust sensor	00011000
Body height sensor	00011001

Name	*Definition*	
	Body weight sensor	00011010
	Body temperature sensor	00011011
	Body fat sensor	00011100
	Blood type sensor	00011101
	Blood pressure sensor	00011110
	Blood sugar sensor	00011111
	Blood oxygen sensor	00100000
	Heart rate sensor	00100001
	Electrograph sensor	00100010
	EEG sensor	00100011
	ECG sensor	00100100
	EMG sensor	00100101
	EOG sensor	00100110
	GSR sensor	00100111
	Bio sensor	00101000
	Weather sensor	00101001
	Facial expression sensor	00101010
	Facial morphology sensor	00101011
	Facial expression characteristics sensor	00101100
	Geomagnetic sensor	00101101
	Proximity sensor	00101110
	Switch sensor	00101111
	Camera sensor	00110001
	Spectrum camera sensor	00110010
	Color camera sensor	00110011
	Depth camera sensor	00110100
	Stereo camera sensor	00110101
	Thermographic camera sensor	00110111
	Engine oil temperature sensor	00111000

Name	*Definition*
	Intake air temperature sensor: 00111001 Tire pressure monitor system sensor: 00111010 Distance traveled sensor: 00111011 Speed sensor: 00111100 Vehicle speed sensor: 00111101 Mass air flow sensor: 00111110 Percentage sensor: 00111111 Fuel level sensor: 01000000 Manifold absolute pressure sensor: 01000001 Engine RPM sensor: 01000010 CoM sensor: 01000011 Radar sensor: 01000100 Array camera sensor: 01000101 Microphone sensor: 01000110 E-nose sensor: 01000111 Reserved: 01001000-11111111
`DeviceCommandListType`	A type that serves as the placeholder for the sequence of device commands.
`NumOfDeviceCmd`	This field, which is only present in the binary representation, specifies the number of `DeviceCmd` instances accommodated in the `DeviceCommandList`.
`IndividualDeviceCmdType`	This field, which is only present in the binary representation, describes which DeviceCmd type shall be used. In the binary description, the following mapping table is used, Terms of device / Binary representation for device type (8bits): Light device: 00000000 Flash device: 00000001 Heating device: 00000010 Cooling device: 00000011 Wind device: 00000100

Name	*Definition*	
	Vibration device	00000101
	Sprayer device	00000110
	Scent device	00000111
	Fog device	00001000
	Color correction device	00001001
	Initialize color correction parameter device	00001010
	Rigid body motion device	00001011
	Tactile device	00001100
	Kinesthetic device	00001101
	Global position command device	00001110
	Bubble device	00001111
	3D printer device	00010000
	Sound display setting device	00010001
	3D printing color reproduction device	00010010
	Array light device	00010011
	Reserved	00010100-11111111

4.3.5 Examples

The following shows two use cases of `InteractionInfo` element, which are for listing device commands and for listing sensed informations.

The first example shows the case when the `InteractionInfo` is used for `DeviceCommandList`.

```
<iidl:InteractionInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004" xmlns:mpegvct="urn:mpeg:mpeg-v:2017:01-
CT-NS" xmlns:iidl="urn:mpeg:mpeg-v:2018:01-IIDL-NS" xmlns:dcv="urn:mpeg:mpeg-
v:2018:01-DCV-NS" xsi:schemaLocation="urn:mpeg:mpeg-v:2018:01-DCV-NS
http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-DCV.xsd">
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:FogType" activate="true"
deviceIdRef="fdc1" id="command1" intensity="20"/>
      ...
      <iidl:DeviceCommand xsi:type="dcv:..." .../>
      ...
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

The second example shows the case when the InteractionInfo is used for SensedInfoList.

```
<iidl:InteractionInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004" xmlns:mpegvct="urn:mpeg:mpeg-v:2018:01-
CT-NS" xmlns:iidl="urn:mpeg:mpeg-v:2018:01-IIDL-NS" xmlns:siv="urn:mpeg:mpeg-
v:2018:01-SIV-NS" xsi:schemaLocation="urn:mpeg:mpeg-v:2018:01-SIV-NS
http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-SIV.xsd">
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="a_type_derived_from_SensedInfoBaseType" .../>
      ...
      <iidl:SensedInfo xsi:type="siv:AccelerationSensorType" .../>
      ...
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

The third example shows the case when the DeviceCommand is used directly as the root element.

```
<iidl:DeviceCommand xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004" xmlns:mpegvct="urn:mpeg:mpeg-v:2018:01-
CT-NS" xmlns:dcv="urn:mpeg:mpeg-v:2018:01-DCV-NS" xmlns:iidl="urn:mpeg:mpeg-
v:2018:01-IIDL-NS" xsi:schemaLocation="urn:mpeg:mpeg-v:2018:01-DCV-NS
http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-DCV.xsd" xsi:type="dcv:LightType" id="light1"
color="urn:mpeg:mpeg-v:01-SI-ColorCS-NS:red" intensity="5">
   <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
</iidl:DeviceCommand>
```

The fourth example shows the case when the SensedInfo is used directly as the root element.

```
<iidl:SensedInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004" xmlns:mpegvct="urn:mpeg:mpeg-v:2018:01-
CT-NS" xmlns:iidl="urn:mpeg:mpeg-v:2018:01-IIDL-NS" xmlns:siv="urn:mpeg:mpeg-
v:2018:01-SIV-NS" xsi:schemaLocation="urn:mpeg:mpeg-v:2018:01-SIV-NS
http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-SIV.xsd" xsi:type="siv:LightSensorType" id="LS001"
sensorIdRef="LSID001" activate="true" value="200" color="#FF0000">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="1000"
pts="600000"/>
</iidl:SensedInfo>
```


Note that these examples are only showing a part of the complete XML description to show the use of the root element, InteractionInfo, with the choice of DeviceCommandList or SensedInfoList.

4.4 Device commands

4.4.1 General

This subclause specifies tools for describing actions that each individual device (actuators) is supposed to take. Instances of following device commands defined in this Clause may be generated as an output of the VR engine and used to drive actuators. In 4.4.3, an abstract complex type of DeviceCommandBaseType is defined, which the device command types of individual device should inherit.

4.4.2 Reference coordinate system

The origin of the reference coordinate for actuators is located at the position of the user as depicted in Figure 2. Each axis is defined as follows. X-axis is in the direction of the left-hand side of the user facing the screen. Y-axis is in the reverse direction of gravity. Z-axis is in the direction of the user's facing the screen. Rotating clockwise along the X-axis is defined as pitch, rotating clockwise along the Y-axis is defined as yaw, and rotating clockwise along the Z-axis is defined as roll.

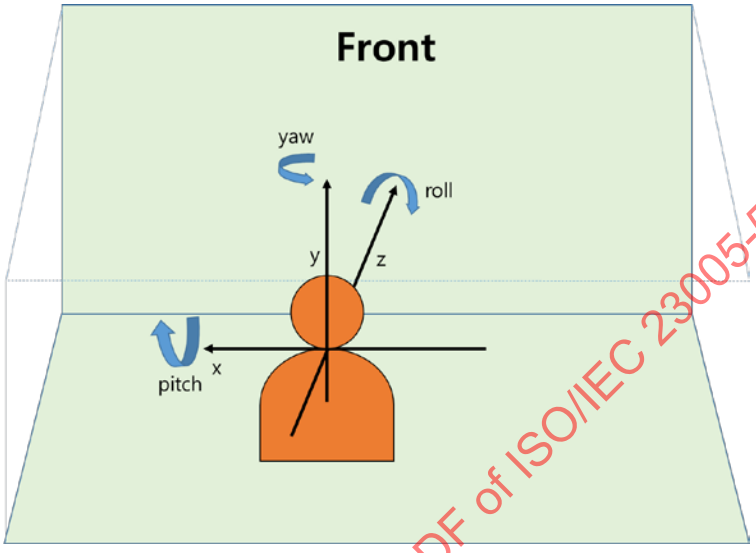


Figure 2 — Reference coordinate system for actuators

4.4.3 Device command base type

4.4.3.1 Syntax

```
<!-- ################################################ -->
<!--  Device command base type                        -->
<!-- ################################################ -->
<complexType name="DeviceCommandBaseType" abstract="true">
   <sequence>
      <element name="TimeStamp" type="mpegvct:TimeStampType"/>
   </sequence>
   <attributeGroup ref="iidl:DeviceCmdBaseAttributes"/>
</complexType>
```

4.4.3.2 Binary representation syntax

DeviceCommandBaseType{	***Number of bits***	***Mnemonic***
TimeStamp		TimeStampType
DeviceCmdBaseAttributes		DeviceCmdBaseAttributesType
}		

4.4.3.3 Semantics

Semantics of the `DeviceCommandBaseType`:

Name	*Definition*
`DeviceCommandBaseType`	Provides the topmost type of the base type hierarchy which each individual device command can inherit.
`TimeStamp`	Provides the timing information for the device command to be executed. As defined in ISO/IEC 23005-6:—, 4.7[2], there is a choice of selection among three timing schemes, which are absolute time, clocktick time, and delta of clock tick time.
`DeviceCmdBaseAttributes`	Describes a group of attributes for the commands.
`TimeStampType`	This field, which is only present in the binary representation, describes which time stamp scheme shall be used. “1” means that the absolute time stamp type shall be used, “2” means that the clock tick time stamp type shall be used, and “3” means that the clock tick time delta stamp type shall be used. “0” is reserved.
`AbsoluteTimeType`	The absolute time stamp is defined in ISO/IEC 23005-6:—, 4.7.
`ClockTickTimeType`	The clock tick time stamp is defined in ISO/IEC 23005-6:—, 4.7.
`ClockTickTimeDeltaType`	The clock tick time delta stamp, which value is the time delta between the present and the past time, is defined in ISO/IEC 23005-6:—, 4.7.

TimeStampSelect	Type Stamp Type
00	Reserved
01	AbsoluteTimeType
10	ClockTickTimeType
11	ClockTickTimeDeltaType

4.4.3.4 Examples

For the examples of the `DeviceCommandBaseType`, please see the examples of individual types of device commands.

4.4.4 Device command base attributes

4.4.4.1 Syntax

```
<!-- ################################################ -->
<!--  Definition of Device Command Base Attributes    -->
<!-- ################################################ -->
<attributeGroup name="DeviceCmdBaseAttributes">
   <attribute name="id" type="ID" use="optional"/>
   <attribute name="deviceIdRef" type="anyURI" use="optional"/>
   <attribute name="activate" type="boolean" use="optional" default="true"/>
</attributeGroup>
```

[2] Fourth Edition under preparation. Stage at time of publication: ISO/IEC FDIS 23005-6:2018.

4.4.4.2 Binary representation syntax

DeviceCmdBaseAttributesType{	Number of bits	Mnemonic
idFlag	1	Bslbf
deviceIdRefFlag	1	Bslbf
activateFlag	1	Bslbf
If(idFlag) {		
id	See ISO/IEC 10646	UTF-8
}		
if(deviceIdRefFlag) {		
deviceIdRef	See ISO/IEC 10646	UTF-8
}		
if(activateFlag) {		
activate	1	bslbf
}		
}		

4.4.4.3 Semantics

Semantics of the `DeviceCmdBaseAttributes`:

Name	*Definition*
`DeviceCmdBaseAttributes`	Specifies the common attributes for any type inherits from the `DeviceCommandBaseType`.
`id`	To be used to identify each individual device command.
`deviceIdRef`	To specify an individual device to which the command is associated.
`activate`	Describes whether the effect shall be activated. A value of `true` means the device shall be activated (switch on) and `false` means the device shall be deactivated (switch off).
`DeviceCmdBaseAttributesType`	Provides the topmost type of the base type hierarchy which the attributes of each individual device command can inherit.
`idFlag`	This field, which is only present in the binary representation, signals the presence of the id attribute. A value of ”1” means the attribute shall be used and ”0” means the attribute shall not be used.

Name	*Definition*
`deviceIdRefFlag`	This field, which is only present in the binary representation, signals the presence of the sensor ID reference attribute. A value of ”1” means the attribute shall be used and ”0” means the attribute shall not be used.
`activateFlag`	This field, which is only present in the binary representation, signals the presence of the activation attribute. A value of ”1” means the attribute shall be used and ”0” means the attribute shall not be used.

4.4.4.4 Examples

The following is a snippet of an XML document showing the use of the DeviceCmdBaseAttributes. It shows a device command specified by the identifier of `command1` to activate a device type *`any_specific_device_command_type`* with device identifier `fdc1`.

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:any_specific_device_command_type"
activate="true" deviceIdRef="fdc1" id="command1"/>
      ...
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

4.5 Sensed information description tools

4.5.1 General

This subclause specifies tools for describing information acquired through each individual sensor. Instances of following sensed information defined in this Clause may be generated as an output of the sensors. The following subclause defines an abstract complex type of `SensedInfoBaseType`, which the sensed information types for each individual sensor should inherit.

4.5.2 Global coordinate for sensors

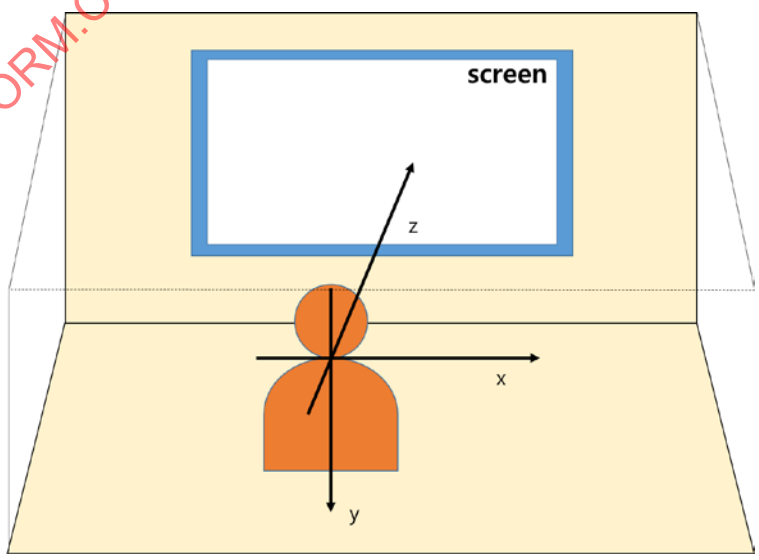


Figure 3 — Reference coordinate for sensors

The reference coordinate for sensors is defined adapting the right-handed coordinate system. Each axis is defined as follows: Y-axis is in the direction of gravity; Z-axis is in the direction of user's front (in common sense) which is orthogonal to the y-axis; X-axis is in the direction of user's right side which is also orthogonal to both y-axis and z-axis. The x-, y-, and z-axis are depicted in Figure 3. The default origin of the reference coordinate for sensors is the position of the user. The origin of the coordinate system differs depending on the type of the sensor.

4.5.3 Sensed information base type

4.5.3.1 Syntax

```
<!-- ################################################ -->
<!--  Sensed information base type                    -->
<!-- ################################################ -->
<complexType name="SensedInfoBaseType" abstract="true">
   <sequence>
      <element name="TimeStamp" type="mpegvct:TimeStampType" minOccurs="0"/>
   </sequence>
   <attributeGroup ref="iidl:sensedInfoBaseAttributes"/>
</complexType>
```

4.5.3.2 Binary representation syntax

SensedInfoBaseTypeType{	***Number of bits***	***Mnemonic***
TimeStampFlag	1	bslbf
SensedInfoBaseAttributes		SensedInfoBaseAttributesType
If(TimeStampFlag){		
TimeStamp		TimeStampType
}		
}		

4.5.3.3 Semantics

Semantics of the `SensedInfoListType`:

Name	*Definition*
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.
`sensedInfoBaseAttributes`	Describes a group of attributes for the sensed information.
`TimeStamp`	Provides the time information at which the sensed information is acquired. As defined in ISO/IEC 23005-6, there is a choice of selection among three timing schemes, which are absolute time, clocktick time, and delta of clock tick time.

Name	*Definition*
`TimeStampFlag`	This field, which is only present in the binary representation, signals the presence of the TimeStamp element. A value of ”1” means the element shall be used and ”0” means the element shall not be used.

4.5.4 Sensed information base attributes

4.5.4.1 Syntax

```
<!-- ################################################### -->
<!--  Definition of Sensed Information Base Attributes   -->
<!-- ################################################### -->
<attributeGroup name="sensedInfoBaseAttributes">
   <attribute name="id" type="ID" use="optional"/>
   <attribute name="sensorIdRef" type="anyURI" use="optional"/>
   <attribute name="linkedlist" type="anyURI" use="optional"/>
   <attribute name="groupID" type="anyURI" use="optional"/>
   <attribute name="activate" type="boolean" use="optional"/>
   <attribute name="priority" type="nonNegativeInteger" use="optional"
default="0"/>
</attributeGroup>
```

4.5.4.2 Binary representation syntax

SensedInfoBaseAttributesType{	***Number of bits***	***Mnemonic***
idFlag	1	bslbf
sensorIdRefFlag	1	bslbf
linkedlistFlag	1	bslbf
groupIDFlag	1	bslbf
priorityFlag	1	bslbf
activateFlag	1	bslbf
If(idFlag) {		
id	See ISO/IEC 10646	UTF-8
}		
if(sensorIdRefFlag) {		
sensorIdRef	See ISO/IEC 10646	UTF-8
}		
if(linkedlistFlag) {		

linkedlist	See ISO/IEC 10646	UTF-8
}		
if(groupIDFlag) {		
groupID	See ISO/IEC 10646	UTF-8
}		
If(priorityFlag) {		
priority	32	uimsbf
}		
if(activateFlag) {		
activate	1	bslbf
}		
}		

4.5.4.3 Semantics

Semantics of the `sensedInfoBaseAttributes`:

Name	*Definition*
`sensedInfoBase Attributes`	Describes a group of attributes for the commands.
`id`	Unique identifier for identifying individual sensed information
`sensorIdRef`	References a sensor that has generated the information included in this specific sensed information.
`linkedlist`	Describes the multi-sensor structure that consists of a group of sensors in a way that each record contains a reference to the ID of the next sensor.
`groupID`	Identifier for a group multi-sensor structure to which this specific sensor belongs.
`activate`	Describes whether the sensor shall be activated. A value of `"true"` means the sensor shall be activated and `"false"` means the sensor shall be deactivated. In the binary representation, A value of “1” means the sensor shall be activated and ”0” means the sensor shall be deactivated.

Name	*Definition*
priority	Describes a priority for sensed information with respect to other sensed information sharing the same point in time when the sensed information becomes adapted. A value of one indicates the highest priority and larger values indicate lower priorities. The default value of the priority is one. If there are more than one sensed information with the same priority, the order of process can be determined by the Adaptation engine itself. NOTE The priority might be used to apply the sensed information on the virtual world object characteristics – defined within a group of sensors – according to the capabilities of the adaptation VR. EXAMPLE The adaptation RV processes the individual sensed information of a group of sensors according to their priority in descending order due to its limited capabilities. That is, the sensed information with the lower priority might get lost.
SensedInfoBaseAttributesType	Tool for describing sensed information base attributes.
IDFlag	This field, which is only present in the binary representation, signals the presence of the ID attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
sensorIdRefFlag	This field, which is only present in the binary representation, signals the presence of the sensor ID reference attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
linkedlistFlag	This field, which is only present in the binary representation, signals the presence of the linked list attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
groupIDFlag	This field, which is only present in the binary representation, signals the presence of the group ID attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
priorityFlag	This field, which is only present in the binary representation, signals the presence of the priority attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
activateFlag	This field, which is only present in the binary representation, signals the presence of the activation attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.

4.5.4.4 Examples

The example of the BaseAttributes is given in the examples of sensed information vocabulary.

5 Device command vocabulary

5.1 General

This Clause describes syntax and semantics of the device command vocabulary to implement commanding of individual devices.

This Clause also describes the binary representation of each individual device command. There are two possible modes for the devices requiring a high speed update rate and large data, such as color correction type, rigid body motion type, and tactile type, can utilize the update mode in addition to the normal mode. The device commands with the update mode parse the elements, which values are different from their corresponding values in the previous device command.

5.2 Schema wrapper conventions

The syntax defined in this Clause assumes the following schema wrapper to form a valid XML schema document.

```
<schema xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004" xmlns:dcv="urn:mpeg:mpeg-v:2018:01-DCV-
NS" xmlns:iidl="urn:mpeg:mpeg-v:2018:01-IIDL-NS" xmlns:mpegvct="urn:mpeg:mpeg-
v:2017:01-CT-NS" targetNamespace="urn:mpeg:mpeg-v:2018:01-DCV-NS"
elementFormDefault="qualified" attributeFormDefault="unqualified"
version="ISO/IEC 23005-5" id="MPEG-V-DCV.xsd">
   <import namespace="urn:mpeg:mpeg-v:2018:01-IIDL-NS"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-IIDL.xsd"/>
   <import namespace="urn:mpeg:mpeg7:schema:2004"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
7_schema_files/mpeg7-v2.xsd"/>
   <import namespace="urn:mpeg:mpeg-v:2018:01-CT-NS"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-CT.xsd"/>
```

Additionally, the following line should be appended to the resulting schema document in order to obtain a well-formed XML document.

```
</schema>
```

5.3 Light type

5.3.1 General

This subclause specifies a device command type which can generate a light effect. The properties of the command can be generated by the adaptation engine, which is combining the light effect specified by ISO/IEC 23005-3 with the user preference toward the light effect and the light device capabilities specified by ISO/IEC 23005-2.

5.3.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of DCV light type                    -->
<!-- ################################################ -->
<complexType name="LightType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <attribute name="color" type="mpegvct:colorType" use="optional"/>
         <attribute name="intensity" type="integer" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

5.3.3 Binary representation syntax

LightType{	***Number of bits***	***Mnemonic***
colorFlag	1	bslbf
intensityFlag	1	bslbf
DeviceCommandBase		DeviceCommandBaseType
if(colorFlag) {		
color	8	colorType
}		
if(intensityFlag) {		
intensity	7	uimsbf
}		
}		

colorType {	***Number of bits***	***Mnemonic***
NamedcolorFlag	1	
If(NamedcolorFlag) {		
NamedColorType	9	bslbf
} else {		
colorRGBType	6	Bslbf
}		
}		

5.3.4 Semantics

Semantics of the `LightType`:

Name	*Definition*
`LightType`	Tool for describing a command for a lighting device to follow.
`color`	Describes the list of colors, which the lighting device can provide, that shall be used either as a reference to a classification scheme term using the mpeg7:termReferenceType defined in ISO/IEC 15938-5:2003, 7.6 or as RGB value. A CS that may be used for this purpose is the ColorCS defined in ISO/IEC 23005-6:—, A.2.2. The binary representation of the ColorCS is defined in ISO/IEC 23005-6:—, A.2.2. EXAMPLE 1 urn:mpeg:mpeg-v:01-SI-ColorCS-NS:alice_blue would describe the color Alice blue. EXAMPLE 2 The RGB representation of the color Alice blue is `#F0F8FF`.
	Describes the intensity that the lighting device shall emit in percentage with respect to the maximum intensity that the specific device can generate.
`colorFlag`	This field, which is only present in the binary representation, signals the presence of color attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`intensityFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`DeviceCommandBase`	Provides the topmost type of the base type hierarchy which each individual device command can inherit.
`NamedcolorFlag`	This field, which is only present in the binary representation, indicates a choice of the color descriptions. If it is 1 then the color is described by mpeg7:termReferenceType, otherwise the color is described by colorRGBType.
`NamedColorType`	This field, which is only present in the binary representation, describes color in terms of ColorCS Flag defined in ISO/IEC 23005-6:—, A.2.2.
`colorRGBType`	This field, which is only present in the binary representation, describes color in terms of colorRGBType.

5.3.5 Examples

This example shows the description of a device command of light effect with the following semantics. The dispaying device for the light effect is “light1”. The intensity shall be 5 % with the color “red” from the classification scheme described in ISO/IEC 23005-6:—, A.2.2.

```
<iidl:InteractionInfo>
  <iidl:DeviceCommandList>
    <iidl:DeviceCommand xsi:type="dcv:LightType" id="light1"
color="urn:mpeg:mpeg-v:01-SI-ColorCS-NS:red" intensity="5">
      <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
    </iidl:DeviceCommand>
  </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.4 Flash type

5.4.1 General

This subclause specifies a device command type which can generate a flash effect. The properties of the command can be generated by the adaptation engine, which is combining the flash effect specified by ISO/IEC 23005-3 with the user preference toward the flash effect and the flash device capabilities specified by ISO/IEC 23005-2.

5.4.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of DCV flash type                    -->
<!-- ################################################ -->
<complexType name="FlashType">
   <complexContent>
      <extension base="dcv:LightType">
         <attribute name="frequency" type="positiveInteger" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

5.4.3 Binary representation syntax

FlashType{	***Number of bits***	***Mnemonic***
frequencyFlag	1	Bslbf
Light		LightType
if(frequencyFlag) {		
frequency	7	Uimsbf
}		
}		

5.4.4 Semantics

Semantics of the `FlashType`:

Name	*Definition*
`FlashType`	Tool for describing a flash device command.
`intensity`	Describes the intensity of the flash effect in terms of illumination in percentage with respect to the maximum light intensity that the specific device can generate.
`frequency`	Describes the number of flickering in percentage with respect to the maximum frequency that the specific flash device can generate.
`Color`	Describes the color of the flash effect as a reference to a classification scheme term or as RGB. A CS that may be used for this purpose is the ColorCS defined in ISO/IEC 23005-6:—, A.2.2. The binary representation of the ColorCS is defined in ISO/IEC 23005-6:—, A.2.2. EXAMPLE 1 urn:mpeg:mpeg-v:01-SI-ColorCS-NS:alice_blue would describe the color Alice blue. EXAMPLE 2 The RGB representation of the color Alice blue is `#F0F8FF`.
`frequencyFlag`	This field, which is only present in the binary representation, signals the presence of `frequency`. A value of "1" means the attribute shall be used and "0" means the attribute shall not be used.

5.4.5 Examples

This example shows the description of a device command of flash effect with the following semantics. The displaying device for the flash effect is "flash1". The intensity shall be 5% of the maximum intensity of "flash1", while the frequency is 10 % of the maximum frequency of "flash1".

```
<InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:FlashType" id="flash1"
         color="urn:mpeg:mpeg-v:01-SI-ColorCS-NS:red" intensity="5"
         frequency="10">
           <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
      </iidl:DeviceCommand>
   </iidl:DeviceCommandList>
</InteractionInfo>
```

5.5 Heating type

5.5.1 General

This subclause specifies a device command type which can generate a heating effect. The properties of the command can be generated by the adaptation engine, which is combining the heating effect specified by ISO/IEC 23005-3 with the user preference toward the heating effect and the heating device capabilities specified by ISO/IEC 23005-2.

5.5.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of DCV heating type                  -->
<!-- ################################################ -->
<complexType name="HeatingType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <attribute name="intensity" type="integer" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

5.5.3 Binary representation syntax

HeatingType{	***Number of bits***	***Mnemonic***
intensityFlag	1	bslbf
DeviceCommandBase		DeviceCommandBaseType
if(intensityFlag) {		
intensity	7	uimsbf
}		
}		

5.5.4 Semantics

Semantics of the `HeatingType`:

Name	*Definition*
`HeatingType`	Tool for describing a command for heating device.
`intensity`	Describes the intensity of the temperature effect of heating in percentage with respect to the capable range of temperature control. If the intensity is not specified, this command shall be interpreted as turning on at the maximum intensity. EXAMPLE If the device can control temperature between 20 and 40 °C, intensity of 20 % means the intensity of 24 °C.
`intensityFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`DeviceCommandBase`	Provides the topmost type of the base type hierarchy which each individual device command can inherit.

5.5.5 Examples

This example shows the description of a device command of heating effect with the following semantics. The identifier for this command is “heater1” and the identifier for the heating device for which this command is issued is “heater001”. The intensity shall be 28 % of the maximum intensity of “heater001.”

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:HeatingType" id="heater1"
         deviceIdRef="heater001" activate="true" intensity="28">
         <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
      </iidl:DeviceCommand>
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.6 Cooling type

5.6.1 General

This subclause specifies a device command type which can generate a cooling effect. The properties of the command can be generated by the adaptation engine, which is combining the cooling effect specified by ISO/IEC 23005-3 with the user preference toward the cooling effect and the cooling device capabilities specified by ISO/IEC 23005-2.

5.6.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of DCV cooling type                  -->
<!-- ################################################ -->
<complexType name="CoolingType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <attribute name="intensity" type="integer" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

5.6.3 Binary representation syntax

CoolingType{	***Number of bits***	***Mnemonic***
intensityFlag	1	bslbf
DeviceCommandBase		DeviceCommandBaseType
if(intensityFlag) {		
intensity	7	uimsbf
}		
}		

5.6.4 Semantics

Semantics of the `CoolingType`:

Name	*Definition*
`CoolingType`	Tool for describing a command for cooling device
`intensity`	Describes the intensity of the temperature effect of cooling in percentage with respect to the capable range of temperature control. The larger intensity value the `CoolingType` description has, the cooler the environment is intended to be driven. If the intensity is not specified, this command shall be interpreted as turning on at the maximum intensity. EXAMPLE If the device can create cooling effect from 30 to 10 °C, intensity of 20 % means the intensity of 26 °C.
`intensityFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`DeviceCommandBase`	Provides the topmost type of the base type hierarchy which each individual device command can inherit.

5.6.5 Examples

This example shows the description of a device command of cooling effect with the following semantics. The identifier for this command is “cooling01” and the identifier for the cooling device for which this command is issued is “cooler001”. The intensity shall be 40 % of the maximum intensity of “cooler001.”

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:CoolingType" id="cooling01"
         deviceIdRef="cooler001" activate="true" intensity="40">
         <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
      </iidl:DeviceCommand>
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.7 Wind type

5.7.1 General

This subclause specifies a device command type which can generate a wind effect. The properties of the command can be generated by the adaptation engine, which is combining the wind effect specified by ISO/IEC 23005-3 with the user preference toward the wind effect and the wind device capabilities specified by ISO/IEC 23005-2.

5.7.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of DCV wind type                     -->
<!-- ################################################ -->
<complexType name="WindType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <attribute name="intensity" type="integer" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

5.7.3 Binary representation syntax

WindType{	***Number of bits***	***Mnemonic***
intensityFlag	1	bslbf
DeviceCommandBase		DeviceCommandBaseType
if(intensityFlag) {		
intensity	7	uimsbf
}		
}		

5.7.4 Semantics

Semantics of the `WindType`:

Name	*Definition*
`WindType`	Tool for describing a wind device command.
`intensity`	Describes the intensity of the wind effect in terms of strength in percentage with respect to the maximum intensity of the specified device. If the intensity is not specified, this command shall be interpreted as turning on at the maximum intensity.
`intensityFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`DeviceCommandBase`	Provides the topmost type of the base type hierarchy which each individual device command can inherit.

5.7.5 Examples

This example shows the description of a device command of wind effect with the following semantics. The identifier for this command is “wind01” and the identifier for the wind device for which this command is issued is “wind001”. The intensity shall be 30 % of the maximum intensity of “wind001”.

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:WindType" id="wind01"
         deviceIdRef="wind001" activate="true" intensity="30">
         <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
      </iidl:DeviceCommand>
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.8 Vibration type

5.8.1 General

This subclause specifies a device command type which can generate a vibration effect. The properties of the command can be generated by the adaptation engine, which is combining the vibration effect specified by ISO/IEC 23005-3 with the user preference toward the vibration effect and the vibration device capabilities specified by ISO/IEC 23005-2.

5.8.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of DCV vibration type                -->
<!-- ################################################ -->
<complexType name="VibrationType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <attribute name="intensity" type="integer" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

5.8.3 Binary representation syntax

VibrationType{	***Number of bits***	***Mnemonic***
intensityFlag	1	bslbf
DeviceCommandBase		DeviceCommandBaseType
if(intensityFlag) {		
intensity	7	uimsbf
}		
}		

5.8.4 Semantics

Semantics of the `VibrationType`:

Name	*Definition*
`VibrationType`	Tool for describing a vibration device command.
`intensity`	Describes the intensity of the vibration effect in terms of strength in percentage with respect to the maximum intensity of the specified device. If the intensity is not specified, this command shall be interpreted as turning on at the maximum intensity.
`intensityFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of "1" means the attribute shall be used and "0" means the attribute shall not be used.
`DeviceCommandBase`	Provides the topmost type of the base type hierarchy which each individual device command can inherit.

5.8.5 Examples

This example shows the description of a device command of vibration effect with the following semantics. The identifier for this command is "vibe01" and the identifier for the vibration device for which this command is issued is "vibe001". The intensity shall be 60 % of the maximum intensity of "vibe001".

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:VibrationType" id="vibe01"
         deviceIdRef="vibe001" activate="true" intensity="60">
         <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
      </iidl:DeviceCommand>
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```


5.9 Sprayer type

5.9.1 General

This Subclause specifies a device command type which can generate a spraying effect. The properties of the command can be generated by the adaptation engine, which is combining the spraying effect specified by ISO/IEC 23005-3 with the user preference toward the spraying effect and the spraying device capabilities specified by ISO/IEC 23005-2.

5.9.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of DCV sprayer type                  -->
<!-- ################################################ -->
<complexType name="SprayerType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <attribute name="sprayingType" type="mpeg7:termReferenceType"/>
         <attribute name="intensity" type="integer" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

5.9.3 Binary representation syntax

SprayerType{	*Number of bits*	*Mnemonic*
sprayingFlag	1	bslbf
intensityFlag	1	bslbf
DeviceCommandBase		DeviceCommandBaseType
if(sprayingFlag) {		
sprayingType	8	bsblf
}		
if(intensityFlag) {		
intensity	7	Uimsbf
}		
}		

5.9.4 Semantics

Semantics of the `SprayerType`:

Name	*Definition*
`SprayerType`	Tool for describing a liquid spraying device command.
`sprayingType`	Describes the type of the sprayed material that shall be used as a reference to a classification scheme term using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7,6. A CS that may be used for this purpose is the `SprayingTypeCS` defined in ISO/IEC 23005-6:—, A.2.7. The binary representation of the `SprayingTypeCS` is defined in ISO/IEC 23005-6:—, A.2.7.
`intensity`	Describes the intensity that the liquid is sprayed in percentage with respect to the maximum intensity described in the device capability. If the intensity is not specified, this command shall be interpreted as turning on at the maximum intensity.
`sprayingFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`intensityFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`DeviceCommandBase`	Provides the topmost type of the base type hierarchy which each individual device command can inherit.

5.9.5 Examples

This example shows the description of a device command of sprayer effect with the following semantics. The identifier for this command is “sprayer01” and the identifier for the sprayer device for which this command is issued is “sprayer001”. The intensity shall be 50 % of the maximum intensity of “sprayer001.” The material to be sprayed is pure water as specified in the SprayingTypeCS of ISO/IEC 23005-6:—, A.2.7.

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:SprayerType" id="sprayer01"
         deviceIdRef="sprayer001" activate="true" intensity="50"
         sprayingType="urn:mpeg:mpeg-v:01-SI-SprayingTypeCS-NS:water">
         <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="T1:30:23"/>
      </iidl:DeviceCommand>
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.10 Scent type

5.10.1 General

This Subclause specifies a device command type which can generate a scent effect. The properties of the command can be generated by the adaptation engine, which is combining the scent effect specified by ISO/IEC 23005-3 with the user preference toward the scent effect and the scent device capabilities specified by ISO/IEC 23005-2.

5.10.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of DCV scent type                    -->
<!-- ################################################ -->
<complexType name="ScentType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <attribute name="scent" type="mpeg7:termReferenceType" use="optional"/>
         <attribute name="intensity" type="integer" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

5.10.3 Binary representation syntax

ScentType{	***Number of bits***	***Mnemonic***
scentFlag	1	bslbf
intensityFlag	1	bslbf
DeviceCommandBase		DeviceCommandBaseType
if(scentFlag) {		
scent	9	bslbf

}		
if(intensityFlag) {		
intensity	7	uimsbf
}		
}		

5.10.4 Semantics

Semantics of the ScentType:

Name	*Definition*
ScentType	Tool for describing a scent device command.
intensity	Describes the intensity of the scent effect in percentage with respect to the maximum intensity described in the device capability. If the intensity is not specified, this command shall be interpreted as turning on at the maximum intensity.
scent	Describes the scent that shall be used as a reference to a classification scheme term using the mpeg7:termReferenceType defined in ISO/IEC 15938-5:2003, 7.6. A CS that may be used for this purpose is the ScentCS defined in ISO/IEC 23005-6:—, A.2.4. The binary representation of the ScentCS is defined in ISO/IEC 23005-6:—, A.2.4.
scentFlag	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
intensityFlag	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
DeviceCommandBase	Provides the topmost type of the base type hierarchy which each individual device command can inherit.

5.10.5 Examples

This example shows the description of a device command of scent effect with the following semantics. The identifier for this command is “scent01” and the identifier for the sprayer device for which this command is issued is “scentdevice001”. The intensity shall be 30 % of the maximum intensity of “scentdevice001”. The scent is defined to be the scent of acacia according to the definition of ScentCS inISO/IEC 23005-6:—, A.2.4.

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:ScentType" id="scent01"
         deviceIdRef="scentdevice001" activate="true" intensity="30"
         scent="urn:mpeg:mpeg-v:01-SI-ScentCS-NS:acacia">
         <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
      </iidl:DeviceCommand>
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.11 Fog type

5.11.1 General

This Subclause specifies a device command type which can generate a fog effect. The properties of the command can be generated by the adaptation engine, which is combining the fog effect specified by ISO/IEC 23005-3 with the user preference toward the fog effect and the fog device capabilities specified by ISO/IEC 23005-2.

5.11.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of DCV fog type                      -->
<!-- ################################################ -->
<complexType name="FogType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <attribute name="intensity" type="integer" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

5.11.3 Binary representation syntax

FogType{	*Number of bits*	*Mnemonic*
intensityFlag	1	bslbf
DeviceCommandBase		DeviceCommandBaseType
if(intensityFlag) {		
intensity	7	uimsbf
}		
}		

5.11.4 Semantics

Semantics of the FogType:

Name	*Definition*
FogType	Tool for describing a fog device command.
intensity	Describes the intensity of the fog effect in percentage with respect to the maximum intensity described in the device capability. If the intensity is not specified, this command shall be interpreted as turning on at the maximum intensity.
intensityFlag	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
DeviceCommandBase	Provides the topmost type of the base type hierarchy which each individual device command can inherit.

5.11.5 Examples

This example shows the description of a device command of fog effect with the following semantics. The identifier for this command is “fog01” and the identifier for the sprayer device for which this command is issued is “fog001”. The intensity shall be 50 % of the maximum intensity of “fog001”.

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:FogType" id="fog01" deviceIdRef="fog001"
         activate="true" intensity="50">
         <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
      </iidl:DeviceCommand>
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.12 Color correction type

5.12.1 General

This subclause specifies a device command type which can generate a color correction effect. The properties of the command can be generated by the adaptation engine, which is combining the color correction effect specified by ISO/IEC 23005-3 with the user preference toward the color correction effect and the color correction device capabilities specified by ISO/IEC 23005-2.

5.12.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of DCV color correction type         -->
<!-- ################################################ -->
<complexType name="ColorCorrectionType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <sequence minOccurs="0" maxOccurs="unbounded">
            <element name="SpatialLocator" type="mpeg7:RegionLocatorType"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>
```

5.12.3 Binary representation syntax

ColorCorrectionType{	***Number of bits***	***Mnemonic***
SpatialLocatorFlag	1	bslbf
DeviceCommandBase		DeviceCommandBaseType
if (SpatialLocatorFlag) {		
LoopSpatialLocator		vluimsbf5
for(k=0;k< LoopSpatialLocator;k++){		
SpatialLocator[k]		mpeg7:RegionLocatorType
}		

}		
}		

5.12.4 Semantics

Semantics of the `ColorCorrectionType`:

Name	*Definition*
`ColorCorrectionType`	Tool for commanding a display device to perform color correction.
`SpatialLocator`	Describes the spatial localization of the still region using `SpatialLocatorType` (optional), which indicates the regions in a video segment where the color correction effect is applied. The `SpatialLocatorType` shall be used as defined in ISO/IEC 15938-5.
`DeviceCommandBase`	Provides the topmost type of the base type hierarchy which each individual device command can inherit.
`LoopSpatialLocator`	This field, which is only present in the binary representation, specifies the number of SpatialLocator contained in the description.
`SpatialLocatorFlag`	This field, which is only present in the binary representation, specifies if `SpatialLocator` field is present in this descrption.

5.12.5 Examples

This example shows the description of a device command of color correction effect with the following semantics. The displaying device for the color correction effect is “tv1”. The color correction effect is applied only to the region defined by the `SpatialLocator`.

```
<iidl:InteractionInfo>
  <iidl:DeviceCommandList>
    <iidl:DeviceCommand xsi:type="dcv:ColorCorrectionType" id="tv1"
       activate="true">
      <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
      <dcv:SpatialLocator>
         <mpeg7:Polygon>
            <mpeg7:Coords mpeg7:dim="8">5 25 0 -2 15 0 0 2 </mpeg7:Coords>
         </mpeg7:Polygon>
      </dcv:SpatialLocator>
    </iidl:DeviceCommand>
  </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.13 Initialize color correction parameter type

5.13.1 General

This command delivers the parameters supporting the color correction effect to devices.

5.13.2 Syntax

```
<!-- ################################################################ -->
<!--  Definition of initialize color correction parameter Type -->
<!-- ################################################################ -->
<complexType name="InitializeColorCorrectionParameterType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <sequence>
            <element name="ToneReproductionCurves"
               type="mpegvct:ToneReproductionCurvesType" minOccurs="0"/>
            <element name="ConversionLUT" type="mpegvct:ConversionLUTType"/>
            <element name="ColorTemperature" type="mpegvct:IlluminantType"
               minOccurs="0"/>
            <element name="InputDeviceColorGamut"
               type="mpegvct:InputDeviceColorGamutType" minOccurs="0"/>
            <element name="IlluminanceOfSurround" type="mpeg7:unsigned12"
               minOccurs="0"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>
```

5.13.3 Binary representation syntax

InitializeColorCorrectinParameterType{	***Number of bits***	***Mnemonic***
ToneReproductionCurvesFlag	1	bslbf
ConversionLUTFlag	1	bslbf
ColorTemperatureFlag	1	bslbf
InputDeviceColorGamutFlag	1	bslbf
IlluminanceOfSurroundFlag	1	bslbf
DeviceCommandBase		DeviceCommandBaseType
if(ToneReproductionCurvesFlag) {		
ToneReproductionCurves		ToneReproductionCurvesType
}		
ConversionLUT		ConversionLUTType
if(ColorTemperatureFlag) {		
ColorTemperature		IlluminantType

}		
if(InputDeviceColorGamutFlag) {		
InputDeviceColorGamut		InputDeviceColorGamutType
}		
if(IlluminanceOfSurroundFlag) {		
IlluminanceOfSurround	12	uimsbf
}		
}		

ToneReproductionCurvesType {	***Number of bits***	***Mnemonic***
NumOfRecords	8	uimsbf
for(i=0;i< NumOfRecords;i++){		
DAC_Value	8	mpeg7:unsigned8
RGB_Value	32*3	mpeg7:doubleVector
}		
}		

ConversionLUTType {	***Number of bits***	***Mnemonic***
RGB2XYZ_LUT	32*3*3	mpeg7:DoubleMatrixType
RGBScalar_Max	32*3	mpeg7:doubleVector
Offset_Value	32*3	mpeg7:doubleVector
Gain_Offset_Gamma	32*3*3	mpeg7:DoubleMatrixType
InverseLUT	32*3*3	mpeg7:DoubleMatrixType
}		

IlluminantType {	***Number of bits***	***Mnemonic***
ElementType	1	bslbf

if(ElementType==00){		
xy_Value	32*2	dia:ChromaticityType
Y_Value	7	uimsbf
}else if(ElementType==01){		
Correlated_CT	8	uimsbf
}		
}		

InputDeviceColorGamutType {	***Number of bits***	***Mnemonic***
typeLength		vluimsbf5
IDCG_Type	8 * typeLength	bslbf
IDCG_Value	32*3*2	mpeg7:DoubleMatrixType
}		

5.13.4 Semantics

Semantics of the `InitializeColorCorrectionParameterType`:

Name	*Definition*
`InitializeColorCorrectionParameterType`	Tool for describing an initialize color correction parameter command.
`ToneReproductionCurves`	This curve shows the characteristics (e.g., gamma curves for R, G and B channels) of the input display device.
`ConversionLUT`	A look-up table (matrix) converting an image between an image color space (e.g. RGB) and a standard connection space (e.g. CIE XYZ).
`ColorTemperature`	An element describing a white point setting (e.g., D65, D93) of the input display device.
`InputDeviceColorGamut`	An element describing an input display device color gamut, which is represented by chromaticity values of R, G, and B channels at maximum DAC values.
`IlluminanceOfSurround`	An element describing an illuminance level of viewing environment. The illuminance is represented by lux.
`ToneReproductionCurvesFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the

Name	*Definition*
	attribute shall not be used.
`ConversionLUTFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of "1" means the attribute shall be used and "0" means the attribute shall not be used.
`ColorTemperatureFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of "1" means the attribute shall be used and "0" means the attribute shall not be used.
`InputDeviceColorGamutFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of "1" means the attribute shall be used and "0" means the attribute shall not be used.
`IlluminanceOfSurroundFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of "1" means the attribute shall be used and "0" means the attribute shall not be used.
`DeviceCommandBase`	Provides the topmost type of the base type hierarchy which each individual device command can inherit.

Semantics of the `ToneReproductionCurvesType`:

Names	*Description*
`NumOfRecords`	This field, which is only present in the binary representation, specifies the number of record (DAC and RGB value) instances accommodated in the ToneReproductionCurves.
`DAC_Value`	An element describing discrete DAC values of input device.
`RGB_Value`	An element describing normalized gamma curve values with respect to DAC values. The order of describing the RGB_Value is R_n, G_n, B_n.

Semantics of the `ConversionLUTType`:

Names	*Description*
`RGB2XYZ_LUT`	This look-up table (matrix) converts an image from RGB to CIE XYZ. The size of the conversion matrix is 3×3 such as $\begin{pmatrix} R_x & G_x & B_x \\ R_y & G_y & B_y \\ R_z & G_z & B_z \end{pmatrix}$. The way of describing the values in the binary representation is in the order of [R_x, G_x, B_x, R_y, G_y, B_y, R_z, G_z, B_z].
`RGBScalar_Max`	An element describing maximum RGB scalar values for GOG transformation.

	The order of describing the RGBScalar_Max is R_{max}, G_{max}, B_{max}.
`Offset_Value`	An element describing offset values of input display device when the DAC is 0. The value is described in CIE XYZ form. The order of describing the Offset_Value is X, Y, Z.
`Gain_Offset_Gamma`	An element describing the gain, offset, gamma of RGB channels for GOG transformation. The size of the Gain_Offset_Gamma matrix is 3×3 such as $\begin{bmatrix} Gain_r & Gain_g & Gain_b \\ Offset_r & Offset_g & Offset_b \\ Gamma_r & Gamma_g & Gamma_b \end{bmatrix}$. The way of describing the values in the binary representation is in the order of [$Gain_r$, $Gain_g$, $Gain_b$; $Offset_r$, $Offset_g$, $Offset_b$; $Gamma_r$, $Gamma_g$, $Gamma_b$].
`InverseLUT`	This look-up table (matrix) converts an image form CIE XYZ to RGB. The size of the conversion matrix is 3×3 such as $\begin{pmatrix} R_x^{\mid} & G_x^{\mid} & B_x^{\mid} \\ R_y^{\mid} & G_y^{\mid} & B_y^{\mid} \\ R_z^{\mid} & G_z^{\mid} & B_z^{\mid} \end{pmatrix}$. The way of describing the values in the binary representation is in the order of [$R_x^{\mid}$, $G_x^{\mid}$, $B_x^{\mid}$; $R_y^{\mid}$, $G_y^{\mid}$, $B_y^{\mid}$; $R_z^{\mid}$, $G_z^{\mid}$, $B_z^{\mid}$].

Semantics of the `IlluminantType`:

Names	*Description*
`ElementType`	This field, which is only present in the binary representation, describes which Illuminant scheme shall be used. In the binary description, the following mapping table is used, Illuminant / IlluminantType 00 / xy and Y value 01 / Correlated_CT
`xy_Value`	An element describing the chromaticity of the light source. The ChromaticityType is specified in ISO/IEC 21000-7.
`Y_Value`	An element describing the luminance of the light source between 0 and 100.
`Correlated_CT`	Indicates the correlated color temperature of the overall illumination. The value expression is obtained through quantizing the range [1667, 25000] into 28 bins in a non-uniform way as specified in ISO/IEC 15938-5.

Semantics of the InputDeviceColorGamutType:

Names	*Description*
typeLength	This field, which is only present in the binary representation, specifies the length of each IDCG_Type instance in bytes. The value of this element is the size of the largest IDCG_Type instance, aligned to a byte boundary by bit stuffing using 0-7 '1' bits.
IDCG_Type	An element describing the type of input device color gamut (e.g., NTSC, SMPTE).
IDCG_Value	An element describing the chromaticity values of RGB channels when the DAC values are maximum. The size of the IDCG_Value matrix is 3×2 such as $\begin{bmatrix} x_r & y_r \\ x_g & y_g \\ x_b & y_b \end{bmatrix}$. The way of describing the values in the binary representation is in the order of $[x_r, y_r, x_g, y_g, x_b, y_b]$.

5.13.5 Exmaples

Examples of the color correction parameters.

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:InitializeColorCorrectionParameterType">
         <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
         <dcv:ToneReproductionCurves>
            <mpegvct:DAC_Value>0</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.0000 0.0000 0.0000</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>16</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.0093 0.0087 0.0076</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>32</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.0304 0.0312 0.0274</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>48</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.0595 0.0633 0.0557</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>64</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.0947 0.1026 0.0957</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>80</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.1391 0.1486 0.1388</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>96</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.1864 0.1974 0.1863</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>112</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.2400 0.2555 0.2426</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>125</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.2907 0.3082 0.2960</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>144</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.3759 0.3951 0.3841</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>160</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.4582 0.4778 0.4673</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>176</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.5491 0.5666 0.5576</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>192</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.6510 0.6653 0.6528</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>208</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.7503 0.7644 0.7635</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>224</mpegvct:DAC_Value>
```

```
            <mpegvct:RGB_Value>0.8483 0.8644 0.8654</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>240</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>0.9445 0.9546 0.9438</mpegvct:RGB_Value>
            <mpegvct:DAC_Value>255</mpegvct:DAC_Value>
            <mpegvct:RGB_Value>1.0000 1.0000 1.0000</mpegvct:RGB_Value>
         </dcv:ToneReproductionCurves>
         <dcv:ConversionLUT>
            <mpegvct:RGB2XYZ_LUT mpeg7:dim="3 3">
               .6000 67.6000 38.0000
               .0000 137.0000 16.5000
               .3650 19.4100 203.9000
            </mpegvct:RGB2XYZ_LUT>
            <mpegvct:RGBScalar_Max>0.9910 0.9860 0.9820</mpegvct:RGBScalar_Max>
            <mpegvct:Offset_Value>0.2150 0.2050 0.4250</mpegvct:Offset_Value>
            <mpegvct:Gain_Offset_Gamma mpeg7:dim="3 3">
               .0228 -0.0228 1.6222
               .0242 -0.0242 1.5624
               .0220 -0.0220 1.6180
            </mpegvct:Gain_Offset_Gamma>
            <mpegvct:InverseLUT mpeg7:dim="3 3">
               .0155 -0.0073 -0.0023
               .0052 0.0099 0.0002
               .0003 -0.0009 0.0049
            </mpegvct:InverseLUT>
         </dcv:ConversionLUT>
         <dcv:ColorTemperature>
            <mpegvct:xy_Value x="0.3127" y="0.3290"/>
            <mpegvct:Y_Value>100</mpegvct:Y_Value>
         </dcv:ColorTemperature>
         <dcv:InputDeviceColorGamut>
            <mpegvct:IDCG_Type>NTSC</mpegvct:IDCG_Type>
            <mpegvct:IDCG_Value mpeg7:dim="3 2">
               .6700 0.3300
               .2100 0.7100
               .1400 0.0800
            </mpegvct:IDCG_Value>
         </dcv:InputDeviceColorGamut>
         <dcv:IlluminanceOfSurround>180</dcv:IlluminanceOfSurround>
      </iidl:DeviceCommand>
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.14 Rigid body motion type

5.14.1 General

This subclause specifies a device command type which can generate a regid body motion effect. The properties of the command can be generated by the adaptation engine, which is combining the regid body motion effect specified by ISO/IEC 23005-3 with the user preference toward the regid body motion effect and the regid body motion device capabilities specified by ISO/IEC 23005-2.

5.14.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of rigid body motion type            -->
<!-- ################################################ -->
<complexType name="RigidBodyMotionType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <sequence>
```

```
            <element name="MoveToward" type="dcv:MoveTowardType"
minOccurs="0"/>
            <element name="Incline" type="dcv:InclineType" minOccurs="0"/>
          </sequence>
          <attribute name="duration" type="float"/>
       </extension>
    </complexContent>
</complexType>

<complexType name="MoveTowardType">
    <attribute name="directionX" type="float"/>
    <attribute name="directionY" type="float"/>
    <attribute name="directionZ" type="float"/>
    <attribute name="speedX" type="float"/>
    <attribute name="speedY" type="float"/>
    <attribute name="speedZ" type="float"/>
    <attribute name="accelerationX" type="float"/>
    <attribute name="accelerationY" type="float"/>
    <attribute name="accelerationZ" type="float"/>
</complexType>

<complexType name="InclineType">
    <attribute name="pitchAngle" type="mpegvct:InclineAngleType" use="optional"/>
    <attribute name="yawAngle" type="mpegvct:InclineAngleType" use="optional"/>
    <attribute name="rollAngle" type="mpegvct:InclineAngleType" use="optional"/>
    <attribute name="pitchSpeed" type="float" use="optional"/>
    <attribute name="yawSpeed" type="float" use="optional"/>
    <attribute name="rollSpeed" type="float" use="optional"/>
    <attribute name="pitchAcceleration" type="float" use="optional"/>
    <attribute name="yawAcceleration" type="float" use="optional"/>
    <attribute name="rollAcceleration" type="float" use="optional"/>
</complexType>
```


5.14.3 Binary representation syntax

RigidBodyMotionType{	***Number of bits***	***Mnemonic***
UpdateMode	1	bslbf
if(UpdateMode ==0){		
RigidBodyMotionNormal		RigidBodyMotionNormalType
}else{		
RigidBodyMotionUpdate		RigidBodyMotionUpdateType
}		
}		
RigidBodyMotionNormalType{	***Number of bits***	***Mnemonic***
MoveTowardFlag	1	bslbf

RigidBodyMotionType{	***Number of bits***	***Mnemonic***
InclineFlag	1	bslbf
durationFlag	1	bslbf
DeviceCommandBase		DeviceCommandBaseType
if(MoveTowardFlag) {		
MoveToward		MoveTowardTypes
}		
if(InclineFlag) {		
Incline		InclineType
}		
if(durationFlag) {		
duration	32	fsbf
}		
}		
MoveTowardType{		
directionXFlag	1	bslbf
directionYFlag	1	bslbf
directionZFlag	1	bslbf
speedXFlag	1	bslbf
speedYFlag	1	bslbf
speedZFlag	1	bslbf
accelerationXFlag	1	bslbf
accelerationYFlag	1	bslbf
accelerationZFlag	1	bslbf
if(directionXFlag){		
directionX	32	fsbf
}		
if(directionYFlag){		

RigidBodyMotionType{	***Number of bits***	***Mnemonic***
directionY	32	fsbf
}		
if(directionZFlag){		
directionZ	32	fsbf
}		
if(speedXFlag){		
speedX	32	fsbf
}		
if(speedYFlag){		
speedY	32	fsbf
}		
if(speedZFlag){		
speedZ	32	fsbf
}		
if(accelerationXFlag){		
accelerationX	32	fsbf
}		
if(accelerationYFlag){		
accelerationY	32	fsbf
}		
if(accelerationZFlag){		
accelerationZ	32	fsbf
}		
}		
InclineType{		
pitchAngleFlag	1	bslbf
yawAngleFlag	1	bslbf

RigidBodyMotionType{	***Number of bits***	***Mnemonic***
rollAngleFlag	1	bslbf
pitchSpeedFlag	1	bslbf
yawSpeedFlag	1	bslbf
rollSpeedFlag	1	bslbf
pitchAccelerationFlag	1	bslbf
YawAccelerationFlag	1	bslbf
rollAccelerationFlag	1	bslbf
if(pitchAngleFlag){		
pitchAngle		InclineAngleType
}		
if(yawAngleFlag){		
yawAngle		InclineAngleType
}		
if(rollAngleFlag){		
rollAngle		InclineAngleType
}		
if(pitchSpeedFlag){		
pitchSpeed	32	fsbf
}		
if(yawSpeedFlag){		
yawSpeed	32	fsbf
}		
if(rollSpeedFlag){		
rollSpeed	32	fsbf
}		
if(pitchAccelerationFlag){		
pitchAcceleration	32	fsbf

RigidBodyMotionType{	***Number of bits***	***Mnemonic***
}		
if(yawAccelerationFlag){		
yawAcceleration	32	fsbf
}		
if(rollAccelerationFlag){		
rollAcceleration	32	fsbf
}		
}		

RigidBodyMotionUpdateModeType{	***Number of bits***	***Mnemonic***
idFlag	1	bslbf
deviceIdRefFlag	1	bslbf
activateFlag	1	bslbf
MoveTowardFlag	1	bslbf
directionXFlag	1	bslbf
directionYFlag	1	bslbf
directionZFlag	1	bslbf
speedXFlag	1	bslbf
speedYFlag	1	bslbf
speedZFlag	1	bslbf
accelerationXFlag	1	bslbf
accelerationYFlag	1	bslbf
accelerationZFlag	1	bslbf
InclineFlag	1	bslbf
pitchAngleFlag	1	bslbf
yawAngleFlag	1	bslbf
rollAngleFlag	1	bslbf

pitchSpeedFlag	1	bslbf
yawSpeedFlag	1	bslbf
rollSpeedFlag	1	bslbf
pitchAccelerationFlag	1	bslbf
yawAccelerationFlag	1	bslbf
rollAccelerationFlag	1	bslbf
durationFlag	1	bslbf
ListUpdate	idFlag+ deviceIdRefFlag+ activateFlag+ MoveTowardFlag+ directionXFlag + directionYFlag + directionZFlag + speedXFlag + speedYFlag + speedZFlag + accelerationXFlag + accelerationYFlag + accelerationZFlag + InclineFlag + PitchAngleFlag + YawAngleFlag + RollAngleFlag + PitchSpeedFlag + YawSpeedFlag + RollSpeedFlag + PitchAccelerationFlag + YawAccelerationFlag + RollAccelerationFlag + durationFlag	bslbf
ListItemNum = 0		
if(idFlag){		
if(ListUpdate[ListItemNum]){		
id	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(deviceIdRefFlag){		
if(ListUpdate[ListItemNum]){		
deviceIdRef	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(activateFlag){		
if(ListUpdate[ListItemNum]){		

activate	1	bslbf
}		
ListItemNum++		
}		
TimeStamp		TimeStamp Type
if(MoveTowardFlag) {		
if(ListUpdate[ListItemNum]){		
if(directionXFlag){		
if(ListUpdate[ListItemNum]){		
directionX	32	fsbf
}		
ListItemNum++		
}		
if(directionYFlag){		
if(ListUpdate[ListItemNum]){		
directionY	32	fsbf
}		
ListItemNum++		
}		
if(directionZFlag){		
if(ListUpdate[ListItemNum]){		
directionZ	32	fsbf
}		
ListItemNum++		
}		
if(speedXFlag){		
if(ListUpdate[ListItemNum]){		
speedX	32	fsbf

}		
ListItemNum++		
}		
if(speedYFlag){		
if(ListUpdate[ListItemNum]){		
speedY	32	fsbf
}		
ListItemNum++		
}		
if(speedZFlag){		
if(ListUpdate[ListItemNum]){		
speedZ	32	fsbf
}		
ListItemNum++		
}		
if(accelerationXFlag){		
if(ListUpdate[ListItemNum]){		
accelerationX	32	fsbf
}		
ListItemNum++		
}		
if(accelerationYFlag){		
if(ListUpdate[ListItemNum]){		
accelerationY	32	fsbf
}		
ListItemNum++		
}		
if(accelerationZFlag){		
if(ListUpdate[ListItemNum]){		

accelerationZ	32	fsbf
}		
ListItemNum++		
}		
}		
ListItemNum++		
}		
if(InclineTypeFlag) {		
if(ListUpdate[ListItemNum]){		
if(pitchAngleFlag){		
if(ListUpdate[ListItemNum]){		
pitchAngle		InclineAngleType
}		
ListItemNum++		
}		
if(yawAngleFlag){		
if(ListUpdate[ListItemNum]){		
yawAngle		InclineAngleType
}		
ListItemNum++		
}		
if(rollAngleFlag){		
if(ListUpdate[ListItemNum]){		
rollAngle		InclineAngleType
}		
ListItemNum++		
}		

if(pitchSpeedFlag){		
if(ListUpdate[ListItemNum]){		
pitchSpeed	32	fsbf
}		
ListItemNum++		
}		
if(yawSpeedFlag){		
if(ListUpdate[ListItemNum]){		
yawSpeed	32	fsbf
}		
ListItemNum++		
}		
if(rollSpeedFlag){		
if(ListUpdate[ListItemNum]){		
rollSpeed	32	fsbf
}		
ListItemNum++		
}		
if(pitchAccelerationFlag){		
if(ListUpdate[ListItemNum]){		
pitchAcceleration	32	fsbf
}		
ListItemNum++		
}		
if(yawAccelerationFlag){		
if(ListUpdate[ListItemNum]){		
yawAcceleration	32	fsbf
}		
ListItemNum++		

}		
if(rollAccelerationFlag){		
if(ListUpdate[ListItemNum]){		
rollAcceleration	32	fsbf
}		
ListItemNum++		
}		
}		
ListItemNum++		
}		
if(durationFlag){		
if(ListUpdate[ListItemNum]){		
duration	32	fsbf
}		
}		
}		

5.14.4 Semantics

Semantics of the `RigidBodyMotionType`:

Name	*Definition*
`RigidBodyMotionType`	Tool for describing a rigid body motion device command.
`MoveToward`	Describes the destination axis values of move toward effect. The type is defined by `dcv:MoveTowardType`.
`Incline`	Describes the rotation angle of incline effect. The type is defined by `dcv:InclineType`.
`duration`	Describes time period during which the rigid body object should continuously move. The object which reaches the destination described by the description of `RigidBodyMotionType` should stay at the destination until it receives another command with activate="false".
`MoveTowardType`	Tool for describing `MoveToward` commands for each axis
`directionX`	Describes the position command on x-axis in terms of centimetre with

Name	*Definition*
	respect to the current position.
directionY	Describes the position command on y-axis in terms of centimetre with respect to the current position.
directionZ	Describes the position command on z-axis in terms of centimetre with respect to the current position.
speedX	Describes the desired speed of the rigid body object on the x-axis in terms of percentage with respect to the maximum speed of the specific device which also be described in the device capability as defined in ISO/IEC 23005-2.
speedY	Describes the desired speed of the rigid body object on the y-axis in terms of percentage with respect to the maximum speed of the specific device which also be described in the device capability as defined in ISO/IEC 23005-2.
speedZ	Describes the desired speed of the rigid body object on the z-axis in terms of percentage with respect to the maximum speed of the specific device which also be described in the device capability as defined in ISO/IEC 23005-2.
accelerationX	Describes the desired acceleration of the rigid body object on the x-axis in terms of percentage with respect to the maximum acceleration of the specific device which may be described in the device capability as defined in ISO/IEC 23005-2.
accelerationY	Describes the desired acceleration of the rigid body object on the y-axis in terms of percentage with respect to the maximum acceleration of the specific device which may be described in the device capability as defined in ISO/IEC 23005-2.
accelerationZ	Describes the desired acceleration of the rigid body object on the z-axis in terms of percentage with respect to the maximum acceleration of the specific device which may be described in the device capability as defined in ISO/IEC 23005-2.
InclineType	Tool for describing Incline commands for each axis.
pitchAngle	Describes the angle to rotate in y-axis, Θ(pitch) in degrees between -180 and 180. NOTE The pitch angle is increased with counter-clock wise.
yawAngle	Describes the angle to rotate in z-axis, Ψ(yaw) in degrees between -180 and 180. NOTE The yaw angle is increased with counter-clock wise.
rollAngle	Describes the angle to rotate in x-axis, φ (roll), in degrees between -180 and 180. NOTE The roll angle is increased with counter-clock wise.
pitchSpeed	Describes the desired speed (command) of rotation for pitch in terms of percentage with respect to the maximum angular speed of the specific device which may be described in the device capability as

Name	*Definition*
	defined in ISO/IEC 23005-2.
`yawSpeed`	Describes the desired speed (command) of rotation for yaw in terms of percentage with respect to the maximum angular speed of the specific device which may be described in the device capability as defined in ISO/IEC 23005-2.
`rollSpeed`	Describes the desired speed (command) of rotation for roll in terms of percentage with respect to the maximum angular speed of the specific device which may be described in the device capability as defined in ISO/IEC 3005-2.
`pitchAcceleration`	Describes the desired acceleration (command) of rotation for pitch in terms of percentage with respect to the maximum angular acceleration of the specific device which may be described in the device capability as defined in ISO/IEC 23005-2.
`yawAcceleration`	Describes the desired acceleration (command) of rotation for yaw in terms of percentage with respect to the maximum angular acceleration of the specific device which may be described in the device capability as defined in ISO/IEC 23005-2.
`rollAcceleration`	Describes the desired acceleration (command) of rotation for roll in terms of percentage with respect to the maximum angular acceleration of the specific device which may be described in the device capability as defined in ISO/IEC 23005-2.
`UpdateMode`	This field, which is only present in the binary representation, signals whether the command is on the normal mode or on the update mode. A value of "1" means the update mode shall be used and "0" means the normal mode shall be used.
`RigidBodyMotionNormal`	This field is used to command a rigid body motion device to perform color correction.
`RigidBodyMotionUpdate`	This field is used to command a rigid body motion device to perform color correction only for the updated elements.
`RigidBodyMotionNormalType`	Tool for commanding a rigid body motion device to perform color correction on the normal mode.
`MoveTowardFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of "1" means the attribute shall be used and "0" means the attribute shall not be used.
`InclineFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of "1" means the attribute shall be used and "0" means the attribute shall not be used.
`durationFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of "1" means the attribute shall be used and "0" means the attribute shall not be used.
`DeviceCommandBase`	Provides the topmost type of the base type hierarchy which each individual device command can inherit.

Name	*Definition*
`directionXFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`directionYFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`directionZFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`speedXFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`speedYFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`speedZFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`accelerationXFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`accelerationYFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`accelerationZFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`pitchAngleFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`yawAngleFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`rollAngleFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`pitchSpeedFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`yawSpeedFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.

Name	*Definition*
`rollSpeedFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`pitchAccelerationFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`yawAccelerationFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`rollAccelerationFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`rigidBodyMotionUpdateType`	Tool for commanding a rigid body motion device to perform color correction on the update mode.
`ListUpdate`	Describes the updated list among all the active elements in the command.

5.14.5 Examples

This example shows the description of a MoveToward device command. This device will be moved 10cm on x-axis with 2cm/sec speed and no acceleration.

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:RigidBodyMotionType">
         <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
         <dcv:MoveToward directionX="10" speedX="2" accelerationX="0"/>
      </iidl:DeviceCommand>
   <iidl:InteractionInfo>
<iidl:DeviceCommandList>
```

This example shows the description of an Incline device command. This device will be rotated 60degree on y-axis with constant 10% of its maximum speed.

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:RigidBodyMotionType">
         <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
         <dcv:Incline yawAngle="60" yawSpeed="10" yawAcceleration="0"/>
      </iidl:DeviceCommand>
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.15 Tactile type

5.15.1 General

This subclause specifies a device command type which can generate a tactile effect. The properties of the command can be generated by the adaptation engine, which is combining the tactile effect specified by

ISO/IEC 23005-3 with the user preference toward the tactile effect and the tactile device capabilities specified by ISO/IEC 23005-2.

5.15.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of DCV tactile type                  -->
<!-- ################################################ -->
<complexType name="TactileType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <sequence>
            <element name="ArrayIntensity" type="mpeg7:FloatMatrixType"
               minOccurs="0"/>
         </sequence>
         <attribute name="tactileDisplay" type="mpeg7:termReferenceType"
               use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

5.15.3 Binary representation syntax

TactileType{	***Number of bits***	***Mnemonic***
UpdateMode	1	bslbf
if(UpdateMode ==0){		
TactileNormal		TactileNormalType
}else{		
TactileUpdate		TactileUpdateType
}		
}		

TactileNormalType{	***Number of bits***	***Mnemonic***
DeviceCommandBase		DeviceCommandBaseType
ArrayIntensityFlag	1	bslbf
tactileDisplayFlag	1	bslbf
if (ArrayIntensityFlag) {		
dimX	4	uimsbf
dimY	16	uimsbf
for (k=0;k<dimX*dimY;k++) {		

ArrayIntensity[k]	32	fsbf
}		
}		
if (tactileDisplayFlag) {		
tactileDisplay	3	bslbf
}		
}		
TactileUpdateType{	***Number of bits***	***Mnemonic***
idFlag	1	bslbf
deviceIdRefFlag	1	bslbf
activateFlag	1	bslbf
ArrayIntensityFlag	1	bslbf
tactileDisplayFlag	1	bslbf
ListUpdate	idFlag + deviceId RefFlag + activate Flag + arrayIntens ityFlag + tactileDi splayFlag + 2	bslbf
ListItemNum = 0		
if(idFlag){		
if(ListUpdate[ListItemNum]){		
IdLength		vluimsbf5
Id	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(deviceIdRefFlag){		
if(ListUpdate[ListItemNum]){		
deviceIdRef	See ISO/IEC 10646	UTF-8
}		

ListItemNum++		
}		
if(activateFlag){		
if(ListUpdate[ListItemNum]){		
Activate	1	bslbf
}		
ListItemNum++		
}		
TimeStamp		TimeStampType
if (tactileDisplayFlag) {		
if(ListUpdate[ListItemNum]){		
dimX	4	uimsbf
}		
ListItemNum++		
if(ListUpdate[ListItemNum]){		
dimY	16	uimsbf
}		
ListItemNum++		
if(ListUpdate[ListItemNum]){		
Array_intensityMask	dimX*dimY	bslbf
for(k=0;k<dimX*dimY;k++){		
if(Array_intensityMask[k]){		
ArrayIntensity[k]	32	fsbf
}		
}		
}		
ListItemNum++		
}		
if (tactileDisplayFlag) {		

if(ListUpdate[ListItemNum]) {		
tactileDisplay	3	bslbf
}		
ListItemNum++		
}		
}		

5.15.4 Semantics

Semantics of the `TactileType`:

Name	*Definition*
`TactileType`	Tool for describing array-type tactile device command. A tactile device is composed of an array of actuators.
`ArrayIntensity`	Describes the intensities of array actuators in percentage with respect to the maximum intensity described in the device capability. If the intensity is not specified, this command shall be interpreted as turning on at the maximum intensity.
`tactileDisplay`	Describes the `tactileDisplay` that shall be used as a reference to a classification scheme term using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. A CS that may be used for this purpose is the `TactileDisplayCS` defined in ISO/IEC 23005-6:—, A.2.11.
`UpdateMode`	This field, which is only present in the binary representation, signals whether the command is on the normal mode or on the update mode. A value of “1” means the update mode shall be used and “0“ means the normal mode shall be used.
`TactileNormal`	This field is used to command an array-type tactile device to perform a tactile effect.
`TactileUpdate`	This field is used to command an array-type tactile device to perform a tactile effect only for the updated elements.
`TactileNormalType`	Tool for commanding an array-type tactile device to perform a tactile effect on the normal mode.
`dimX`	This field, which is only present in the binary representation, specifies the x-direction size of ArrayIntensity.
`dimY`	This field, which is only present in the binary representation, specifies the y-direction size of ArrayIntensity.
`TactileUpdateType`	Tool for commanding an array-type tactile device to perform a tactile effect on the update mode.
`ListUpdate`	This field, which is only present in the update mode of binary representation, describes the list of active elements only, among all the active elements in the command.
`Array_intensityMask`	This field, which is only present in the binary syntax, specifies a bit-field that indicates whether the updated value is assigned to the

Name	*Definition*
	corresponding partition.

5.15.5 Examples

An example of the TactileType to provide commands to actuate a tactile device is provided as follows. When tactile data formed as an array are given, these data are mapped to tactile devices with 3-by-3 array (note that tactile data are formed with i-by-j array can be resized to map with the 3-by-3 array of a tactile device. The command data are given as MxN matrix as shown in the example.

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:TactileType">
         <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
         <dcv:ArrayIntensity mpeg7:dim="3 3">
            25 25 25 0 0 0 15 12 15
         </dcv:ArrayIntensity>
      </iidl:DeviceCommand>
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.16 Kinesthetic type

5.16.1 General

This subclause specifies a device command type which can generate a kinesthetic effect. The properties of the command can be generated by the adaptation engine, which is combining the kinesthetic effect specified by ISO/IEC 23005-3 with the user preference toward the kinesthetic effect and the kinesthetic device capabilities specified by ISO/IEC 23005-2.

5.16.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of DCV kinesthetic type              -->
<!-- ################################################ -->
<complexType name="KinestheticType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <sequence>
            <element name="Position" type="mpegvct:Float3DVectorType"
               minOccurs="0"/>
            <element name="Orientation" type="mpegvct:Float3DVectorType"
               minOccurs="0"/>
            <element name="Force" type="mpegvct:Float3DVectorType"
               minOccurs="0"/>
            <element name="Torque" type="mpegvct:Float3DVectorType"
               minOccurs="0"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>
```

5.16.3 Binary representation syntax

KinestheticType {	***Number of bits***	***Mnemonic***
UpdateMode	1	bslbf
if(UpdateMode ==0){		
KinestheticNormal		KinestheticNormalType
}else{		
KinestheticUpdate		KinestheticUpdateType
}		
}		

KinestheticNormalType{	***Number of bits***	***Mnemonic***
PositionFlag	1	bslbf
OrientationFlag	1	bslbf
ForceFlag	1	bslbf
TorqueFlag	1	bslbf
DeviceCommandBase		DeviceCommandBaseType
if(PositionFlag){		
Position		Float3DVectorType
}		
if(OrientationFlag){		
Orientation		Float3DVectorType
}		
if(ForceFlag){		
Force		Float3DVectorType
}		
if(TorqueFlag){		
Torque		Float3DVectorType
}		

}		
Float3DVectorType {		
X	32	fsbf
Y	32	fsbf
Z	32	fsbf
}		

KinestheticUpdateType{	***Number of bits***	***Mnemonic***
idFlag	1	bslbf
deviceIdRefFlag	1	bslbf
activateFlag	1	bslbf
PositionFlag	1	bslbf
OrientationFlag	1	bslbf
ForceFlag	1	bslbf
TorqueFlag	1	bslbf
ListUpdate	idFlag + deviceIdRef Flag + activateFlag + PositionFlag +Orient ationFlag + ForceFla g + TorqueFlag	bslbf
ListItemNum = 0		
if(idFlag){		
if(ListUpdate[ListItemNum]){		
id	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(deviceIdRefFlag){		
if(ListUpdate[ListItemNum]){		
deviceIdRef	See ISO/IEC 10646	UTF-8
}		

ListItemNum++		
}		
if(activateFlag){		
if(ListUpdate[ListItemNum]){		
activate	1	bslbf
}		
ListItemNum++		
}		
TimeStamp		TimeStampType
if(PositionFlag){		
if(ListUpdate[ListItemNum]){		
Position		Float3DVectorType
}		
ListItemNum++		
}		
if(OrientationFlag){		
if(ListUpdate[ListItemNum]){		
Orientation		Float3DVectorType
}		
ListItemNum++		
}		
if(ForceFlag){		
if(ListUpdate[ListItemNum]){		
Force		Float3DVectorType
}		
ListItemNum++		
}		
if(TorqueFlag){		
if(ListUpdate[ListItemNum]){		

Torque		Float3DVectorType
}		
}		
}		

5.16.4 Semantics

Semantics of the `KinestheticType`:

Name	*Definition*
`KinestheticType`	Describes a command for a kinesthetic device.
`Position`	Describes the position that a kinesthetic device shall take in millimetres along each axis of X, Y, and Z, with respect to the home position of the device.
`Orientation`	Describes the orientation that a kinesthetic device shall take in degrees along each axis of X, Y, and Z, with respect to the home orientation of the device.
`Force`	Describes the force of kinesthetic effect in percentage with respect to the maximum force described in the device capability. If the `Force` is not specified, this command shall be interpreted as turning on at the maximum force. This element takes `Float3DVectorType` type defined in ISO/IEC 23005-6.
`Torque`	Describes the torque of kinesthetic effect in percentage with respect to the maximum torque described in the device capability. If the `Torque` is not specified, this command shall be interpreted as turning on at the maximum torque. This element takes `Float3DVectorType` type defined in ISO/IEC 23005-6.
`UpdateMode`	This field, which is only present in the binary representation, signals whether the command is on the normal mode or on the update mode. A value of “1” means the update mode shall be used and “0“ means the normal mode shall be used.
`TactileNormal`	This field is used to command a kinesthetic device to perform kinesthetic effects.
`TactileUpdate`	This field is used to command a kinesthetic device to perform kinesthetic effects only for the updated elements.
`TactileNormalType`	Tool for commanding a kinesthetic device to perform kinesthetic effects on the normal mode.
`PositionFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`OrientationFlag`	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the

Name	*Definition*
	attribute shall be used and “0” means the attribute shall not be used.
ForceFlag	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
TorqueFlag	This field, which is only present in the binary representation, signals the presence of device command attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
DeviceCommandBase	Provides the topmost type of the base type hierarchy which each individual device command can inherit.
Float3DVectorType	Tool for describing a 3D vector
X	Describes the sensed value in x-axis.
Y	Describes the sensed value in y-axis.
Z	Describes the sensed value in z-axis.
KinestheticUpdateType	Tool for commanding a kinesthetic device to perform kinesthetic effects on the normal mode.
ListUpdate	Describes the updated list among all the active elements in the command.

5.16.5 Examples

An example of the KinestheticType, to provide commands to actuate a kinesthetic device, is provided as follows. This example provide position, $P_{x,y,z}$(40 mm, 60 mm, 120 mm) and orientation, $O_{x,y,z}$(5 degree, 7 degree, 19 degree) information that a kinesthetic device may be moved to.

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:KinestheticType">
         <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
         <dcv:Position>
            <mpegvct:X>40</mpegvct:X>
            <mpegvct:Y>60</mpegvct:Y>
            <mpegvct:Z>120</mpegvct:Z>
         </dcv:Position>
         <dcv:Orientation>
            <mpegvct:X>5</mpegvct:X>
            <mpegvct:Y>7</mpegvct:Y>
            <mpegvct:Z>19</mpegvct:Z>
         </dcv:Orientation>
      </iidl:DeviceCommand>
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.17 Global position command type

5.17.1 General

This subclause specifies XML syntax, binary representation syntax, and semantics of the GlobalPositionCommandType command with an example instantiation of the command. This command is intended to command an unmanned mobile vehicle/device, such as an unmanned aerial vehicle or an unmanned automobile, to move to a certain position specified by the global position coordinates. The altitude attribute may not be applicable to automobiles as they cannot change their altitude. On the other hand, the altitude attribute should be accepted as a command by the aerial vehicles as they can change its altitude as commanded.

5.17.2 Syntax

```
<!--############################################ -->
<!--Definition of global position command type  -->
<!--############################################ -->
<complexType name="GlobalPositionCommandType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <attribute name="crs" type="anyURI"
default="urn:ogc:def:crs:EPSG::4979"/>
         <attribute name="longitude" use="required">
            <simpleType>
               <restriction base="double">
                  <minInclusive value="-180.0"/>
                  <maxInclusive value="180.0"/>
               </restriction>
            </simpleType>
         </attribute>
         <attribute name="latitude" use="required">
            <simpleType>
               <restriction base="double">
                  <minInclusive value="-90.0"/>
                  <maxInclusive value="90.0"/>
               </restriction>
            </simpleType>
         </attribute>
         <attribute name="altitude" type="double" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```


5.17.3 Binary representation syntax

GlobalPositionCommandType{	***Number of bits***	***Mnemonic***
DeviceCommandBaseType	See above	DeviceCommmandBaseType
altitudeFlag	1	
crs	See ISO/IEC 10646	UTF-8
longitude	32	fsfb

latitude	32	fsfb
if (altitudeFlag) {		
altitude	32	fsfb
}		
}		

5.17.4 Semantics

Semantics of the `GlobalPositionCommandType`:

Name	*Definition*
`GlobalPositionCommmandType`	Tool for commanding mobile device to move to the destination designated by the description.
`TimeStamp`	Describes the time that the command is issued.
`crs`	Specifies the URI of the coordinate reference system based on which the values of longitude, latitude and altitude are given. The default is urn:ogc:def:crs:EPSG::4979 specifying the Coordinate Reference System (CRS) with code 4979 specified in the EPSG database available at http://www.epsg.org/.
`longitude`	Describes the destination point in degrees of longitude. Positive values represent eastern longitude and negative values represent western longitude. EXAMPLE -132.236 represents 132.236 degrees West.
`latitude`	Describes the destination point in degrees of latitude. Positive value represents northern latitude and negative value represents southern latitude. EXAMPLE 37.103 represents 37.103 degrees North.
`altitude`	Describes the destination altitude in terms of metres above the geoid. When this attribute is not specified, it implies that the device is requested to maintain the current altitude.

5.17.5 Examples

This example shows the description of a global position command with the following semantics. The mobile device of id “FLY001” is command to go to the latitude of 37.23456 degrees N, the longitude of 131.23456 degrees E, and the altitude of 252.7 metres above the geoid. The command is issued at system clock tick of 600000 where there are 1000 ticks per second. The id of this command is “GPC001.”

```
<iidl:DeviceCommand xsi:type="dcv:GlobalPositionCommandType" id="GPC001"
deviceIdRef="FLY001" activate="true" longitude="131.23456" latitude="37.23456"
altitude="252.7">
<iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
</iidl:DeviceCommand>
```

5.18 Bubble type

5.18.1 General

This subclause specifies XML syntax, binary representation syntax, and semantics of the BubbleType command with an example instantiation of the command. This command is intended to command a bubble effect generator.

5.18.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of Bubble type                       -->
<!-- ################################################ -->
<complexType name="BubbleType">
  <complexContent>
    <extension base="iidl:DeviceCommandBaseType"/>
  </complexContent>
</complexType>
```

5.18.3 Binary representation

BubbleType {	***Number of bits***	***Mnemonic***
DeviceCommandBase		DeviceCommandBaseType
}		

5.18.4 Semantics

Semantics of the BubbleType:

Name	*Definition*
BubbleType	Tool for describing a bubble device command.

5.18.5 Examples

This example shows the description of a device command of bubble effect with the following semantics. The device for the bubble effect is “bubble1”.

This example shows the description of a device command of bubble effect with the following semantics. The identifier for this command is “bubble01” and the identifier for the bubble device for which this command issued is “bubble001”. The activate value is true of “bubble001.”

```
<iidl:InteractionInfo>
  <iidl:DeviceCommandList>
    <iidl:DeviceCommand          xsi:type="dcv:BubbleType"          id="bubble01"
deviceIdRef="bubble001" activate="true">
      <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
    </iidl:DeviceCommand>
  </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.19 3D printer type

5.19.1 General

This subclause specifies a device command type which can control a 3D printer.

5.19.2 Syntax

```
<!-- ################################################ -->
<!--    Definition of 3D Printer Type                 -->
<!-- ################################################ -->
<complexType name="ThreeDPrinterType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
         <sequence>
            <element name="PrintingMaterial"
type="dcdv:ThreeDPrintingMaterialType" minOccurs="0" maxOccurs="unbounded"/>
            <element name="ObjectToPrint" type="anyURI"/>
         </sequence>
         <attribute name="scalingFactor" type="float" use="optional"/>
         <attribute name="insideFillDensity" type="float" use="optional"/>
         <attribute name="surfaceThickness" type="integer" use="optional"/>
         <attribute name="surfaceThicknessUnit" type="mpeg7:termReferenceType"
use="optional"/>
         <attribute name="useSupporter" type="boolean" default="true"/>
         <attribute name="usePlatformAdhesion" type="boolean" default="true"/>
      </extension>
   </complexContent>
</complexType>
```

5.19.3 Binary representation

ThreeDPrinterType {	***Number of bits***	***Mnemonic***
DeviceCommandBase		DeviceCommandBaseType
scalingFactorFlag	1	bslbf
insideFillDensityFlag	1	bslbf
surfaceThicknessFlag	1	bslbf
surfaceThicknessUnitFlag	1	bslbf
useSupporterFlag	1	bslbf

usePlatformAdhesionFlag	1	bslbf
PrintingMaterialCount	32	uimsbf
for(i=0;i<PrintingMaterialCount;i++) {		
PrintingMaterial[i]		ThreeDPrintingMaterialType
}		
ObjectToPrint	See ISO/IEC 10646	UTF-8
if(scalingFactorFlag) {		
scalingFactor	32	fsbf
}		
if(insideFillDensityFlag) {		
insideFillDensity	32	fsbf
}		
if(surfaceThicknessFlag) {		
surfaceThickness	32	fsbf
}		
if(surfaceThicknessUnitFlag) {		
surfaceThicknessUnit	8	UnitTypeCS
}		
useSupporter	1	bslbf
usePlatformAdhesion	1	bslbf
}		

5.19.4 Semantics

Semantics of the `ThreeDPrinterType`:

Name	*Definition*
`ThreeDPrinterType`	Tool for describing a command for a 3D printer
`PrintingMaterial`	Specifies the material to be used in printing the given object as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in 7.6 of ISO/IEC 15938-5:2003. The CS that may be used for this purpose is the `PrintingMaterialTypeCS` defined in ISO/IEC 23005-6:—, A.2.19. The binary representation of the `PrintingMaterialTypeCS` is also defined in ISO/IEC 23005-6:—, A.2.19.

Name	*Definition*
ObjectToPrint	Specifies the actual location of the file in which the 3D object to be printed is specified in one of the printable format by the selected printer.
scalingFactor	Specifies the scaling factor of the object to be printed, if the object is not to be printed in actual size.
insideFillDensity	Specifies how much the inside is filled with printing material. This attribute is in the range of 0~100%. If leaveInsideEmpty is "true", this attribute is not specified.
surfaceThickness	Specifies the thickness of the object surface, when inside of a solid object is left empty, in mm (millimetre) by default. If the value is given in unit other that the millimetre, the unit should be given in the surfaceThicknessUnit attribute.
surfaceThicknessUnit	Specifies the unit of the surface thickness, as a reference to a classification scheme term that shall be using the mpeg7:termReferenceType defined in 7.6 of ISO/IEC 15938-5:2003. If this attribute is not specified, the default unit of mm (millimetre) is used. The CS that may be used for this purpose is the UnitTypeCS defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the UnitTypeCS is also defined in ISO/IEC 23005-6:—, A.2.1.
useSupporter	Specifies whether to use the printing supporter or not. The default value is “true”. The supporter type may be determined by the dedicated printing software.
usePlatformAdhesion	Specifies whether to use the printing platform adhesion or not. The default value is “true”. The adhesion type may be determined by the dedicated printing software.

5.19.5 Examples

An example of the ThreeDPrinterType to provide a command to actuate a 3D printer device is provided as follows. Printing material with various properties, such as thermal operation range between 200 to 300 degree, material class=”metal clay”, and its provider=“posco”, and its name=“ultra super alloy”, is provided. Its color is set to dark blue. Printing object file is also provided in the form of URI, “http://3dp.etri.re.kr/wheel.obj”.

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand xsi:type="dcv:ThreeDPrinterType">
         <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="1:30:23"/>
         <dcv:PrintingMaterial minThermalOperatingRange="200"
maxThermalOperatingRange="300" thermalOperatingRangeUnit="degree"
printingMaterialClass="metalc" printingMaterialProvider="Posco"
printingMaterialName="Ultra super alloy">
                  <dcdv:Color>dark_blue</dcdv:Color>
               <dcdv:Description>wheel</dcdv:Description>
            </dcv:PrintingMaterial>
               <dcv:ObjectToPrint>http://3dp.etri.re.kr/wheel.obj</dcv:ObjectT
oPrint>
      </iidl:DeviceCommand>
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.20 Sound display setting type

5.20.1 General

Sound display unit, e.g. an active speaker set with embedded equalizer, can be set to fulfil specific aural need of a user. This command enables setting of a sound display unit depending on the need of the user, for example a user with auditory impairments.

5.20.2 Syntax

```
<complexType name="SoundDisplayUnitSettingType">
 <complexContent>
  <extension base="iidl:DeviceCommandBaseType">
   <sequence>
    <element name="BandSettingInformation" minOccurs="1" maxOccurs= "unbounded">
     <complexType>
      <attribute name="bandIDRef" type="string" use="required"/>
      <attribute name="intensity" type="float" use="required"/>
     </complexType>
    </element>
   </sequence>
  </extension>
 </complexContent>
</complexType>
```

5.20.3 Binary representation

SoundDisplaySettingType {	***Number of bits***	***Mnemonic***
DeviceCommandBaseType		DeviceCommandBaseType
bandCount	8	uimsbf
for(k=0;k< banCount;k++) {		
bandIDRef	See ISO/IEC 10646	UTF-8
intensity	8	uimsbf
}		
}		

5.20.4 Semantics

Semantics of the `SoundDisplaySettingType`:

Name	*Definition*
`SoundDisplaySettingType`	Tool for describing sound display unit setting based on `DeviceCommandBasedType` including setting specific band ranges and its intensities.

Name	*Definition*
DeviceCommandBassedType	SoundType is inherited from DeviceCommandBassedType. This type already contains devicedURL, activate, id and TimeStamp for general device command descriptions.
BandSettingInformation	Specifies the information for setting a particular band range. This element contains a bandID and its intensity as attirubes
bandCount	Specifies the number of bands to be set in the target sound display unit. This field is only present in the binary format.
bandIDRef	References a target band ID to be set.
itensity	Describes the intensity of a particular band range in percentile.

5.20.5 Examples

This example shows a device command description in order to handle the setting of sound display unit. The intensities of two band ranges of id “band1” and “band2” are scaled down to 50% and 67% of the maximum range of each subband for a speaker identified with device id “speaker001”. The “spk0001” is the id of the description itself.

```
<iidl:InteractionInfo xmlns="urn:mpeg:mpeg-v:2014:01-IIDL-NS"
xmlns:mpegvct="urn:mpeg:mpeg-v:2017:01-CT-NS" xmlns:dcv="urn:mpeg:mpeg-v:2017:01-
DCV-NS" xmlns:xsi= "http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:mpeg:mpeg-v:2017:01-IIDL-NS MPEG-V-IIDL.xsd">
 <DeviceCommandList>
  <DeviceCommand xsi:type="SoundDisplayUnitSettingType" id="spk0001"
deviceIdRef="speaker001" activate= "true">
   <TimeStamp xsi:type="mpegvct:ClockTickTimeDeltaType" timeScale="100"/>
   <BandSetingInformation bandIDRef="band1" intensity="50"/>
   <BandSetingInformation bandIDRef="band2" intensity="67"/>
  </DeviceCommand>
 </DeviceCommandList>
</iidl:InteractionInfo>
```

5.21 3D printing color reproduction type

5.21.1 General

This subclause specifies XML syntax, binary representation syntax, and semantics of the 3D printing color reproduction type command with an example instantiation of the command. This command is intended to command a 3D printer.

5.21.2 XML representation syntax

```
<complexType name="ThreeDPrintingColorReproductionType">
   <complexContent>
      <extension base="iidl:DeviceCommandBaseType">
   </complexContent>
</complexType>
```

5.21.3 Binary representation syntax

ThreeDPrintingColorReproductionType {	**Number of bits**	**Mnemonic**
DeviceCommandBaseType		DeviceCommandBaseType
}		

5.21.4 Semantics

Semantics of the `ThreeDPrintingColorReproductionType` type:

Name	*Definition*
`ThreeDPrintingColorReproductionType`	Tool for commanding a 3D printer to perform color reproduction.

5.21.5 Examples

This example shows the description of a device command of 3D printing color reproduction with the following semantics. The identifier of this command is “cr01” and the identifier of the 3D printer for which this command is issued is “3dp001”. The activation of the color reproduction function is true on the “3dp001” device.

```
<iidl:InteractionInfo>
   <iidl:DeviceCommandList>
      <iidl:DeviceCommand      xsi:type="dcv:ThreeDPrintingColorReproductionType"
id="cr01" deviceIdRef="3dp001" activate="true">
      </iidl:DeviceCommand>
   </iidl:DeviceCommandList>
</iidl:InteractionInfo>
```

5.22 Arrayed light type

5.22.1 General

The device command of an arrayed light is specified by the following syntax. The attribute of the command is inherited from `DeviceCommandBaseType`. The commands for the arrayed light are explained by `LightSamples` having a color (RGB) matrix to represent.

5.22.2 Syntax

```
<complexType name="ArrayedLightType">
  <complexContent>
    <extension base="iidl:DeviceCommandBaseType">
      <element name="LightSamples" type="mpegvct:colorMatrixType"
minOccurs="0"/>
    </extension>
  </complexContent>
</complexType>
```

5.22.3 Binary representation syntax

ArrayedLightType {	*Number of bits*	*Mnemonic*
lightSamplesFlag	1	bslbf
intensityFlag	1	bslbf
DeviceCommandBase		DeviceCommandBaseType
if(lightSamplesFlag) {		
SizeOfLightSamplesRow	16	uimsbf
SizeOfLightSamplesColumn	16	uimsbf
for(k=0;k<(SizeOfLightSamplesRow * SizeOfLightSamplesColumn);k++) {		
LightSamples[k]		ColorType
}		
}		

5.22.4 Semantics

Semantics of the `ArrayedLightType`:

Name	*Definition*
`ArrayedLightType`	Tool for describing a command for an arrayed light device.
`LightSamples`	Describes a sequences of colors, which the light device supports, that shall be used either as a reference to a classification scheme term using the mpeg7:termReferenceType defined in 7.6 of ISO/IEC 15938-5:2003 or as RGB value.

5.22.5 Examples

The example shows the description of a device command for an arrayed light with the following semantics. The identifier for the light effect device is “arrayedlight1”. The color shall be represented with a 3-by-3 color matrix.

```
<InteractionInfo
xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004"
xsi:schemaLocation="urn:mpeg:mpeg-v:2014:01-IIDL-NS                MPEG-V-IIDL.xsd"
xmlns:iidl="urn:mpeg:mpeg-v:2014:01-IIDL-NS"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:mpegvct="urn:mpeg:mpeg-v:2014:01-CT-NS"
xmlns:dcv="urn:mpeg:mpeg-v:2014:01-DCV-NS"
xmlns:siv="urn:mpeg:mpeg-v:2014:01-SIV-NS"
```

```
xmlns="urn:mpeg:mpeg-v:2014:01-IIDL-NS">

  <iidl:DeviceCommandList>
    <iidl:DeviceCommand xsi:type="dcv:ArrayedLightType" id="arrayedlight1">
      <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="12:30:15"/>
      <dcv:LightSamplesIntensity mpeg7:dim="3 3">
      #000000 #FFFFFF #000000
      #000000 #FFFFFF #000000
      #000000 #FFFFFF #000000
      </dcv:LightSamplesIntensity>
    </iidl:DeviceCommand>
  </iidl:DeviceCommandList>
</InteractionInfo>
```


6 Sensed information vocabulary

6.1 General

This Clause describes syntax and semantics of the sensed information vocabulary to implement exchange of information acquired from individual sensors.

This Clause also describes the binary representation of each individual sensed information. There are two possible modes for the sensors requiring a high speed update rate and large data, such as motion sensor and intelligent camera, can utilize the update mode in addition to the normal mode. The sensed information with the update mode parses the elements, which values are different from their corresponding values in the previous sensed information.

6.2 Schema wrapper conventions

The syntax defined in this Clause assumes the following schema wrapper to form a valid XML schema document.

```
<schema xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004" xmlns:siv="urn:mpeg:mpeg-v:2017:01-SIV-
NS" xmlns:iidl="urn:mpeg:mpeg-v:2017:01-IIDL-NS" xmlns:mpegvct="urn:mpeg:mpeg-
v:2017:01-CT-NS" targetNamespace="urn:mpeg:mpeg-v:2017:01-SIV-NS"
elementFormDefault="qualified" attributeFormDefault="unqualified"
version="ISO/IEC 23005-5" id="MPEG-V-SIV.xsd">
   <import namespace="urn:mpeg:mpeg-v:2017:01-IIDL-NS"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-IIDL.xsd"/>
   <import namespace="urn:mpeg:mpeg-v:2017:01-CT-NS"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-CT.xsd"/>
   <import namespace="urn:mpeg:mpeg7:schema:2004"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
7_schema_files/mpeg7-v2.xsd"/>
```

Additionally, the following line should be appended to the resulting schema document in order to obtain a well-formed XML document.

```
</schema>
```

6.3 Light sensor type

6.3.1 General

This subclause specifies a sensor type which senses light intensity and color. The light sensor type does not specify any sensing methods such as photo resistor technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The properties of the sensor are specified in the light sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include multisensorial effect control, home securities, environmental monitoring and others.

6.3.2 Syntax

```
<!--##################################### -->
<!--Definition of light sensor type      -->
<!--##################################### -->
<complexType name="LightSensorType">
  <complexContent>
    <extension base="iidl:SensedInfoBaseType">
      <attribute name="value" type="siv:valueType" use="optional"/>
      <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      <attribute name="color" type="mpegvct:colorType" use="optional"/>
      <attribute name="colorValue" type="siv:colorValueType"
use="optional"/><attribute name="model" type="siv:colorSpaceType"
use="optional"/>
    </extension>
  </complexContent>
</complexType>

<simpleType name="valueType">
    <union memberTypes="float siv:colorWType"/>
  </simpleType>

  <simpleType name="colorWType">
    <restriction base="NMTOKEN">
      <whiteSpace value="collapse"/>
      <pattern value="#[0-9A-Fa-f]{2}"/>
    </restriction>
  </simpleType>

<simpleType name="colorValueType">
  <restriction base="mpeg7:doubleVector">
    <length value="3"/>
  </restriction>
</simpleType>

<simpleType name="colorSpaceType">
  <restriction base="NMTOKEN">
    <enumeration value="XYZ"/>
    <enumeration value="Yxy"/>
    <enumeration value="Lab"/>
    <enumeration value="Lch"/>
    <enumeration value="LUV"/>
    <enumeration value="HunterLab"/>
  </restriction>
</simpleType>
```

6.3.3 Binary representation syntax

LightSensorType{	*Number of bits*	*Mnemonic*
valueFlag	1	bslbf
unitFlag	1	bslbf
colorFlag	1	bslbf
colorValueFlag	1	bslbf
modelFlag	1	bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(valueFlag) {		
value		valueType
}		
if(unitFlag) {		
unit	8	bslbf
}		
if(colorFlag) {		
color		colorType
}		
if(colorValueFlag) {		
colorValue	32*3	fsbf
}		
if(modelFlag) {		
model	3	bslbf
}		
}		
valueType {		
whiteFlag	1	bslbf
if (whiteFlag) {		
white	8	bslbf

} else {		
lux	32	fsbf
}		
}		

6.3.4 Semantics

Semantics of the `LightSensorType`:

Name	*Definition*
`LightSensorType`	Tool for describing sensed information with respect to a light sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`value`	Describes the sensed intensity of the light with respect to lux. This attribute can be used to represent "white" when the light sensor senses "RGBW". EXAMPLE `#F0` would describe the white color value in XML syntax.
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the `UnitTypeCS` is also defined in ISO/IEC 23005-6:—, A.2.1.
`color`	Describes the list of colors which the light sensor can sense either as a reference to a classification scheme that shall be using the `meg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6 or as RGB value. A CS that may be used for this purpose is the `ColorCS` defined in ISO/IEC 23005-6:—, A.2.2. EXAMPLE 1 `urn:mpeg:mpeg-v:01-SI-ColorCS-NS:alice_blue` would describe the color Alice blue. EXAMPLE 2 The RGB representation of the color Alice blue is `#F0F8FF`.
`valueType`	Describes the light intensity with respect to lux or white.
`colorWType`	Tool for describing a color in 1 byte (256 level) value of white.
`colorValue`	Describes the sensed values of a color sensor with respect to color space models.
`model`	Describes the color model of the sensed values from a color sensor using colorSpaceType.

Name	*Definition*
`colorValueType`	Describes three values from a color sensor. The meaning of the three values is determined by the color space model. EXAMPLE The color model *CIEXYZ* would have three values of *X*, *Y*, and *Z* in order.
`colorSpaceType`	The color space models utilized by a color sensor are Yxy, CIEXYZ, CIELAB, CIELCH, CIELUV, and Hunter Lab. color space / value: Yxy: 000 CIEXYZ (XYZ): 001 CIELAB (Lab): 010 CIELCH (Lch): 011 CIELUV (LUV): 100 HunterLab: 101 reserved: 110-111
`valueFlag`	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.
`colorFlag`	This field, which is only present in the binary representation, signals the presence of color attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`colorValueFlag`	This field, which is only present in the binary representation, signals the presence of colorValue attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`modelFlag`	This field, which is only present in the binary representation, signals the presence of model attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.
`whiteFlag`	This field, which is only present in the binary representation, indicates a choice of the `value` descriptions. If it is 1 then the `value` is given by the white, otherwise the `value` is described by `lux`.

Name	*Definition*
white	This field, which is only present in the binary representation, describes value means “white” when the light sensor senses “RGBW”.
lux	This field, which is only present in the binary representation, describes value means “lux”.

6.3.5 Examples

This example shows the description of a light sensing with the following semantics. The sensor has an ID of “LS001” and references “LSID001”. The sensor shall be activated and the value shall be 200 (lux) with the color #FF0000. The sensor shall be sensed at system clock tick of 600000 where there are 1000 ticks per second.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:LightSensorType" id="LS001"
         sensorIdRef="LSID001" activate="true" value="200" color="#FF0000">
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="1000"
            pts="600000"/>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.4 Ambient noise sensor type

6.4.1 General

This subclause specifies a sensor type which senses ambinet noise and its duration. The ambient noise sensor type does not specify any sensing methods such as audio and microphone technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The properties of the sensor are specified in the ambient noise sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include multisensorial effect control, home securities, environmental monitoring and others.

6.4.2 Syntax

```
<!--##########################################-->
<!--Definition of ambient noise sensor type  -->
<!--##########################################-->
<complexType name="AmbientNoiseSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="lifespan" type="float" use="optional"/>
         <attribute name="value" type="float" use="optional"/>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.4.3 Binary representation syntax

AmbientNoiseSensorType{	*Number of bits*	*Mnemonic*
lifespanFlag	1	bslbf
valueFlag	1	bslbf
unitFlag	1	bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(lifespanFlag) {		
lifespan	32	fsbf
}		
if(valueFlag) {		
value	32	fsbf
}		
if(unitFlag) {		
unit	8	bslbf
}		
}		

6.4.4 Semantics

Semantics of the `AmbientNoiseSensorType`:

Name	*Definition*
`AmbientNoiseSensorType`	Tool for describing sensed information using an ambient noise sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`lifespan`	Describes the duration taken to measure the information based on the timestamp. The unit of lifespan is the internal clock count.
`value`	Describes the sensed value of the ambient noise with respect to decibel (dB).

unit	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the mpeg7:termReferenceType defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the UnitTypeCS defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the UnitTypeCS is also defined in ISO/IEC 23005-6:—, A.2.1.
lifespanFlag	This field, which is only present in the binary representation, signals the presence of the life span attribute. A value of “1” means the lifespan shall be used and “0” means the lifespan shall not be used.
valueFlag	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
unitFlag	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.
SensedInfoBaseType	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.

6.4.5 Examples

This example shows the description of an ambient noise sensing with the following semantics. The sensor has an ID of “ANS001” and references “ANSID001”. The sensor shall be activated and the value shall be 10 (dB). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second with a lifespan of 5 seconds.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:AmbientNoiseSensorType" id="ANS001"
         sensorIdRef="ANSID001" activate="true" value="10" lifespan="500">
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType"
               timeScale="100" pts="60000"/>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```


6.5 Temperature sensor type

6.5.1 General

This subclause specifies a sensor type which senses temperature. The temperature sensor type does not specify any sensing methods such as the thermally sensitive resistor technology. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The properties of the sensor are specified in the temperature sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include multisensorial effect control, home securities, environmental monitoring and others.

6.5.2 Syntax

```
<!--########################################## -->
<!--Definition of temperature sensor type  -->
<!--########################################## -->
<complexType name="TemperatureSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="value" type="float" use="optional"/>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.5.3 Binary representation syntax

TemperatureSensorType{	***Number of bits***	***Mnemonic***
valueFlag	1	bslbf
unitFlag	1	bslbf
if(valueFlag) {		
value	32	fsbf
}		
if(unitFlag) {		
unit	8	bslbf
}		
}		

6.5.4 Semantics

Semantics of the `TemperatureSensorType`:

Name	*Definition*
`TemperatureSensorType`	Tool for describing sensed information with respect to a temperature sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit (°C) is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the `UnitTypeCS` is also defined in

Name	*Definition*
	ISO/IEC 23005-6:—, A.2.1.
`Value`	Describes the sensed value of the temperature with respect to the Celsius scale.
`valueFlag`	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.

6.5.5 Examples

This example shows the description of a temperature sensing with the following semantics. The sensor has an ID of “TS001” and references “TSID001”. The sensor shall be activated and the value shall be 36.5 (°C). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:TemperatureSensorType" id="TS001"
         sensorIdRef="TSID001" activate="true" value="36.5">
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="60000"/>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.6 Humidity sensor type

6.6.1 General

This subclause specifies a sensor type which senses humidity. The humidity sensor type does not specify any sensing methods such as the capacitive, resisitive, and conductivity technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The properties of the sensor are specified in the humidity sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include multisensorial effect control, home securities, environmental monitoring and others.

6.6.2 Syntax

```
<!--##################################### -->
<!--Definition of humidity sensor type   -->
<!--##################################### -->
<complexType name="HumiditySensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="value" type="float" use="optional"/>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.6.3 Binary representation syntax

HumiditySensorType{	*Number of bits*	*Mnemonic*
valueFlag	1	bslbf
unitFlag	1	bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(valueFlag) {		
value	32	fsbf
}		
if(unitFlag) {		
unit	8	bslbf
}		
}		

6.6.4 Semantics

Semantics of the `HumiditySensorType`:

Name	*Definition*
`HumiditySensorType`	Tool for describing sensed information with respect to a humidity sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the `UnitTypeCS` is also defined in ISO/IEC 23005-6:—, A.2.1.
`value`	Describes the value sensed by the humidity sensor with respect to percentage (%).
`valueFlag`	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.

Name	*Definition*
SensedInfoBaseType	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.

6.6.5 Examples

This example shows the description of a humidity sensing with the following semantics. The sensor has an ID of “HS001” and references “HSID001”. The sensor shall be activated and the value shall be 60 (%). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:HumiditySensorType" id="HS001"
         sensorIdRef="HSID001" activate="true" value="60">
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="60000"/>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.7 Distance sensor type

6.7.1 General

This subclause specifies a sensor type which senses distance from the sensor-specified point to a sensed object. The distance sensor type does not specify any sensing methods such as ultrasonic, optical, and inductive technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The sensing properties of the sensor are specified in the distance sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include robotics, security systems, and others.

6.7.2 Syntax

```
<!--################################### -->
<!--Definition of distance sensor type   -->
<!--################################### -->
<complexType name="DistanceSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="value" type="float" use="optional"/>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.7.3 Binary representation syntax

DistanceSensorType{	***Number of bits***	***Mnemonic***
valueFlag	1	bslbf
unitFlag	1	bslbf

SensedInfoBaseType		SensedInfoBaseTypeType
if(valueFlag) {		
value	32	fsbf
}		
if(unitFlag) {		
unit	8	bslbf
}		
}		

6.7.4 Semantics

Semantics of the `DistanceSensorType`:

Name	*Definition*
`DistanceSensorType`	Tool for describing sensed information with respect to a length sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the `UnitTypeCS` is also defined in ISO/IEC 23005-6:—, A.2.1.
`value`	Describes the sensed value from the length sensor with respect to metre (m).
`valueFlag`	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of "1" means the attribute shall be used and "0" means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of "1" means the user-defined unit shall be used and "0" means the user-defined unit shall not be used.
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.

6.7.5 Examples

This example shows the description of a length sensing with the following semantics. The sensor has an ID of "LS001" and references "LSID001". The sensor shall be activated and the value shall be 5 (m). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:DistanceSensorType" id="DS001"
         sensorIdRef="DSID001" activate="true" value="5.0" >
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="60000"/>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.8 Atmospheric pressure sensor type

6.8.1 General

This subclause specifies a sensor type which senses atmospheric pressure. The atmospheric pressure sensor type does not specify any sensing methods such as the capacitive, resisitive, and conductivity technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The properties of the sensor are specified in the atmospheric pressure sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include multisensorial effect control, environmental monitoring and others.

6.8.2 Syntax

```
<!--################################################ -->
<!--Definition of atmospheric pressure Sensor type  -->
<!--################################################ -->
<complexType name="AtmosphericPressureSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="value" type="float" use="optional"/>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.8.3 Binary representation syntax

AtmosphericPressureSensorType{	***Number of bits***	***Mnemonic***
valueFlag	1	bslbf
unitFlag	1	bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(valueFlag) {		
value	32	fsbf

}		
if(unitFlag) {		
unit	8	bslbf
}		
}		

6.8.4 Semantics

Semantics of the `AtmosphericPressureSensorType`:

Name	*Definition*
`AtmosphericPressureSensorType`	Tool for describing sensed information with respect to an atmospheric pressure sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the `UnitTypeCS` is also defined in ISO/IEC 23005-6:—, A.2.1.
`value`	Describes the sensed value from the humidity sensor with respect to hectopascal (hPa).
`valueFlag`	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.

6.8.5 Examples

This example shows the description of an atmospheric pressure sensing with the following semantics. The sensor has an ID of “APS001” and references “APSID001”. The sensor shall be activated and the value shall be 1000 (hPa). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:AtmosphericPressureSensorType" id="APS001"
         sensorIdRef="APSID001" activate="true" value="1000.0" >
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="60000"/>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.9 Position sensor type

6.9.1 General

This subclause specifies a sensor type which senses position. The position sensor type does not specify any sensing methods such as ultrasonic, optical, and inductive technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The sensing properties of the sensor are specified in the position sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include robotics, natural user interface, security systems, and others.

6.9.2 Syntax

```
<!--##################################### -->
<!--Definition of position sensor type  -->
<!--##################################### -->
<complexType name="PositionSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <sequence>
            <element name="Position" type="mpegvct:Float3DVectorType"
               minOccurs="0"/>
         </sequence>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.9.3 Binary representation syntax

PositionSensorType {	***Number of bits***	***Mnemonic***
UpdateMode	1	bslbf
if(UpdateMode ==0){		
PositionSensorNormal		PositionSensorNormalType
}else{		
PositionSensorUpdate		PositionSensorUpdateType
}		
}		

PositionSensorNormalType{	***Number of bits***	***Mnemonic***
positionFlag	1	bslbf
unitFlag	1	bslbf

SensedInfoBaseType		SensedInfoBaseTypeType
if(positionFlag) {		
position		Float3DVectorType
}		
if(unitFlag) {		
unit	8	bslbf
}		
}		

PositionSensorUpdateType {	***Number of bits***	***Mnemonic***
TimeStampFlag	1	bslbf
IDFlag	1	bslbf
sensorIdRefFlag	1	bslbf
linkedlistFlag	1	bslbf
groupIDFlag	1	bslbf
priorityFlag	1	bslbf
activateFlag	1	bslbf
positionFlag	1	bslbf
unitFlag	1	bslbf
ListUpdate	TimeStampFlag + IDFlag + sensorIdRefFlag + linkedlistFlag + groupIDFlag + priorityFlag + activateFlag + positionFlag + unitFlag	bslbf
ListItemNum = 0		
if(TimeStampFlag){		
if(ListUpdate[ListItemNum]){		
TimeStamp		TimeStampType
}		
ListItemNum++		
}		

if(IDFlag){		
if(ListUpdate[ListItemNum]){		
ID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(sensorIdRefFlag){		
if(ListUpdate[ListItemNum]){		
sensorIdRef	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(linkedlistFlag){		
if(ListUpdate[ListItemNum]){		
linkedlist	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(groupIDFlag){		
if(ListUpdate[ListItemNum]){		
groupID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(priorityFlag){		
if(ListUpdate[ListItemNum]){		
priority	8	uimsbf
}		
ListItemNum++		

}		
if(activateFlag){		
if(ListUpdate[ListItemNum]){		
activate	1	bslbf
}		
ListItemNum++		
}		
if(positionFlag){		
if(ListUpdate[ListItemNum]){		
position		Float3DVectorType
}		
ListItemNum++		
}		
if(unitFlag){		
if(ListUpdate[ListItemNum]){		
unit	8	bslbf
}		
}		
}		

6.9.4 Semantics

Semantics of the `PositionSensorType`:

Name	*Definition*
`PositionSensorType`	Tool for describing sensed information with respect to a position sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`Position`	Describes the 3D value of the position sensor in the unit of metre (m). The origin of the coordinate shall be the position of the object sensed at the time of sensor activation. If a calibration has been performed on the position of the sensor, the origin shall be the position after the calibration. If this sensed information is used with the `PositionSensorCapability` information defined in ISO/IEC 23005-2, the origin of the coordinate shall be defined in the `PositionSensorCapability`.

Name	*Definition*
`Unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the `UnitTypeCS` is also defined in ISO/IEC 23005-6:—, A.2.1.
`UpdateMode`	This field, which is only present in the binary representation, signals whether the sensed information is on the normal mode or on the update mode. A value of “1” means the update mode shall be used and “0” means the normal mode shall be used.
`PositionSensorNormal`	This field is used to send the sensed information from a position sensor.
`PositionSensorUpdate`	This field is used to send the sensed information from a position sensor only for the updated elements.
`PositionSensorNormalType`	This field is used to send the sensed information from a position sensor on the normal mode.
`positionFlag`	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.
`PositionSensorUpdateType`	This field is used to send the sensed information from a position sensor on the update mode.
`ListUpdate`	Describes the updated list among all the active elements in the sensed information.
`priority`	Describes a priority for sensed information with respect to other sensed information sharing the same point in time when the sensed information becomes adapted. A value of one indicates the highest priority and larger values indicate lower priorities. The default value of the priority is one. If there are more than one sensed information with the same priority, the order of process can be determined by the Adaptation engine itself. NOTE The priority might be used to apply the sensed information on the virtual world object characteristics – defined within a group of sensors – according to the capabilities of the adaptation VR. EXAMPLE The adaptation RV processes the individual sensed information of a group of sensors according to their priority in descending order due to its limited capabilities. That is, the sensed information with the lower priority might get lost.

6.9.5 Examples

This example shows the description of a position sensing with the following semantics. The sensor has an ID of “PS001” and references “PSID001”. The sensor shall be activated and the value shall be Px="1.5" (m), Py="0.5" (m), and Pz="-2.1" (m). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:PositionSensorType" id="PS001"
         sensorIdRef="PSID001" activate="true" >
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="60000"/>
         <siv:Position>
            <mpegvct:X>1.5</mpegvct:X>
            <mpegvct:Y>0.5</mpegvct:Y>
            <mpegvct:Z>-2.1</mpegvct:Z>
         </siv:Position>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.10 Velocity sensor type

6.10.1 General

This subclause specifies a sensor type which senses velocity. The velocity sensor type does not specify any sensing methods such as ultrasonic, optical, inercial and inductive technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The sensing properties of the sensor are specified in the velocity sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include robotics, natural user interface, security systems, and others.

6.10.2 Syntax

```
<!--##################################### -->
<!--Definition of velocity sensor type   -->
<!--##################################### -->
<complexType name="VelocitySensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <sequence>
            <element name="Velocity" type="mpegvct:Float3DVectorType"
               minOccurs="0"/>
         </sequence>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.10.3 Binary representation syntax

VelocitySensorType {	***Number of bits***	***Mnemonic***
UpdateMode	1	bslbf

if(UpdateMode ==0){		
VelocitySensorNormal		VelocitySensorNormalType
}else{		
VelocitySensorUpdate		VelocitySensorUpdateType
}		
}		

VelocitySensorNormalType{	***Number of bits***	***Mnemonic***
velocityFlag	1	bslbf
unitFlag	1	bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(velocityFlag) {		
velocity		Float3DVectorType
}		
if(unitFlag) {		
unit	8	bslbf
}		
}		

VelocitySensorUpdateType{	***Number of bits***	***Mnemonic***
TimeStampFlag	1	bslbf
IDFlag	1	bslbf
sensorIdRefFlag	1	bslbf
linkedlistFlag	1	bslbf
groupIDFlag	1	bslbf
priorityFlag	1	bslbf
activateFlag	1	bslbf
velocityFlag	1	bslbf

unitFlag	1	bslbf
ListUpdate	TimeStampFlag + IDFlag + sensorIdRefFlag + linkedlistFlag + groupIDFlag + priorityFlag + activateFlag + velocityFlag + unitFlag	bslbf
ListItemNum = 0		
if(TimeStampFlag){		
if(ListUpdate[ListItemNum]){		
TimeStamp		TimeStampType
}		
ListItemNum++		
}		
if(IDFlag){		
if(ListUpdate[ListItemNum]){		
ID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(sensorIdRefFlag){		
if(ListUpdate[ListItemNum]){		
sensorIdRef	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(linkedlistFlag){		
if(ListUpdate[ListItemNum]){		
linkedlist	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		

if(groupIDFlag){		
if(ListUpdate[ListItemNum]){		
groupID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(priorityFlag){		
if(ListUpdate[ListItemNum]){		
priority	8	uimsbf
}		
ListItemNum++		
}		
if(activateFlag){		
if(ListUpdate[ListItemNum]){		
activate	1	bslbf
}		
ListItemNum++		
}		
if(velocityFlag){		
if(ListUpdate[ListItemNum]){		
velocity		Float3DVectorType
}		
ListItemNum++		
}		
if(unitFlag){		
if(ListUpdate[ListItemNum]){		
unit	8	bslbf
}		
}		
}		

6.10.4 Semantics

Semantics of the `VelocitySensorType`:

Name	*Definition*
`VelocitySensorType`	Tool for describing sensed information with respect to a velocity sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`Velocity`	Describes the sensed velocity by the sensor in a three dimensional vector with respect to metre per second (m/s).
`Unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the `UnitTypeCS` is also defined in ISO/IEC 23005-6:—, A.2.1.
`UpdateMode`	This field, which is only present in the binary representation, signals whether the sensed information is on the normal mode or on the update mode. A value of “1” means the update mode shall be used and “0” means the normal mode shall be used.
`VelocitySensorNormal`	This field, which is only present in the binary representation, is used to send the sensed information from a velocity sensor.
`VelocitySensorUpdate`	This field, which is only present in the binary representation, is used to send the sensed information from a velocity sensor only for the updated elements.
`VelocitySensorNormalType`	This field, which is only present in the binary representation, is used to send the sensed information from a velocity sensor on the normal mode.
`velocityFlag`	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.
`VelocitySensorUpdateType`	This field is used to send the sensed information from a velocity sensor on the update mode.
`ListUpdate`	Describes the updated list among all the active elements in the sensed information.
`Priority`	Describes a priority for sensed information with respect to other sensed information sharing the same point in time when the sensed information becomes adapted. A value of one indicates the highest

Name	*Definition*
	priority and larger values indicate lower priorities. The default value of the priority is one. If there are more than one sensed information with the same priority, the order of process can be determined by the Adaptation engine itself. NOTE The priority might be used to apply the sensed information on the virtual world object characteristics – defined within a group of sensors – according to the capabilities of the adaptation VR. EXAMPLE The adaptation RV processes the individual sensed information of a group of sensors according to their priority in descending order due to its limited capabilities. That is, the sensed information with the lower priority might get lost.

6.10.5 Examples

This example shows the description of a velocity sensing with the following semantics. The sensor has an ID of "VS001" and references "VSID001". The sensor shall be activated and the value shall be Vx="10.0" (m/s), Vy="5.0" (m/s), and Vz="0.1" (m/s). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:VelocitySensorType" id="VS001"
         sensorIdRef="VSID001" activate="true" >
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="60000"/>
         <siv:Velocity>
            <mpegvct:X>10.0</mpegvct:X>
            <mpegvct:Y>5.0</mpegvct:Y>
            <mpegvct:Z>0.1</mpegvct:Z>
         </siv:Velocity>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.11 Acceleration sensor type

6.11.1 General

This subclause specifies a sensor type which senses acceleration. The acceleration sensor type does not specify any sensing methods such as ultrasonic, optical, inercial and inductive technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The sensing properties of the sensor are specified in the acceleration sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include robotics, natural user interface, security systems, and others.

6.11.2 Syntax

```
<!--########################################## -->
<!--Definition of acceleration sensor type  -->
<!--########################################## -->
<complexType name="AccelerationSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <sequence>
            <element name="Acceleration" type="mpegvct:Float3DVectorType"
```

```
                minOccurs="0"/>
          </sequence>
          <attribute name="axis" type="mpeg7:unsigned2" use="optional"/>
          <attribute name="unit" type="mpegvct:unitType" use="optional"/>
       </extension>
    </complexContent>
</complexType>
```

6.11.3 Binary representation syntax

AccelerationSensorType {	***Number of bits***	***Mnemonic***
UpdateMode	1	bslbf
if(UpdateMode ==0){		
AccelerationSensorNormal		AccelerationSensorNormalType
}else{		
AccelerationSensorUpdate		AccelerationSensorUpdateType
}		
}		

AccelerationSensorType{	***Number of bits***	***Mnemonic***
accelerationFlag	1	bslbf
axisFlag	1	bslbf
unitFlag	1	bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(accelerationFlag) {		
acceleration		Float3DVectorType
}		
if(axisFlag) {		
axis	2	uimsbf
}		
if(unitFlag) {		
unit	8	bslbf
}		
}		

AccelerationSensorUpdateType {	***Number of bits***	***Mnemonic***
TimeStampFlag	1	bslbf
IDFlag	1	bslbf
sensorIdRefFlag	1	bslbf
linkedlistFlag	1	bslbf
groupIDFlag	1	bslbf
priorityFlag	1	bslbf
activateFlag	1	bslbf
accelerationFlag	1	bslbf
axisFlag	1	bslbf
unitFlag	1	bslbf
ListUpdate	TimeStampFlag + IDFlag + sensorIdRefFlag + linkedlistFlag + groupIDFlag + priorityFlag + activateFlag + accelerationFlag + axisFlag + unitFlag	bslbf
ListItemNum = 0		
if(TimeStampFlag){		
if(ListUpdate[ListItemNum]){		
TimeStamp		TimeStampType
}		
ListItemNum++		
}		
if(IDFlag){		
if(ListUpdate[ListItemNum]){		
ID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		

if(sensorIdRefFlag){		
if(ListUpdate[ListItemNum]){		
sensorIdRef	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(linkedlistFlag){		
if(ListUpdate[ListItemNum]){		
linkedlist	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(groupIDFlag){		
if(ListUpdate[ListItemNum]){		
groupID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(priorityFlag){		
if(ListUpdate[ListItemNum]){		
priority	8	uimsbf
}		
ListItemNum++		
}		
if(activateFlag){		
if(ListUpdate[ListItemNum]){		
activate	1	bslbf
}		
ListItemNum++		

}		
if(accelerationFlag){		
if(ListUpdate[ListItemNum]){		
acceleration		Float3DVectorType
}		
ListItemNum++		
}		
if(axisFlag) {		
if(ListUpdate[ListItemNum]){		
axis	2	uimsbf
}		
ListItemNum++		
}		
if(unitFlag){		
if(ListUpdate[ListItemNum]){		
unit	8	bslbf
}		
}		
}		

6.11.4 Semantics

Semantics of the `AccelerationSensorType`:

Name	*Definition*
`AccelerationSensorType`	Tool for describing sensed information with respect to an acceleration sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`Acceleration`	Describes the value of the acceleration sensor in a three dimensional vector with respect to m/s^2. When the *axis* is 1, only the first entry of the 3D vector is used. When the *axis* is 2, the first and the second entries of the 3D vector are used. When the *axis* is 3, all three entries of the 3D vector are used. By the nature of the sensor, the three entries of the 3D vectors are meant to be the values of three orthorgonal axis, but are irrelevant to the global coordinate system.

Name	*Definition*
`Acceleration`	Describes the value of the acceleration sensor in a three dimensional vector with respect to m/s^2. When the *axis* is 1, only *X* is used. When the *axis* is 2, *X* and *Y* are used. When the *axis* is 3, *X, Y*, and Z are used.
`axis`	The number of axis that the acceleration sensor can measure. The axis value shall be either 1, 2, or 3. The default axis is 3.
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the `UnitTypeCS` is also defined in ISO/IEC 23005-6:—, A.2.1.
`UpdateMode`	This field, which is only present in the binary representation, signals whether the sensed information is on the normal mode or on the update mode. A value of “1” means the update mode shall be used and “0” means the normal mode shall be used.
`AccelerationSensorNormal`	This field is used to send the sensed information from an acceleration sensor.
`AccelerationSensorUpdate`	This field is used to send the sensed information from an acceleration sensor only for the updated elements.
`AccelerationSensorNormalType`	This field is used to send the sensed information from an acceleration sensor on the normal mode.
`accelerationFlag`	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
axisFlag	This field, which is only present in the binary representation, signals the presence of axis attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.
`AccelerationSensorUpdateType`	This field is used to send the sensed information from an acceleration sensor on the update mode.
`ListUpdate`	Describes the updated list among all the active elements in the sensed information.
`priority`	Describes a priority for sensed information with respect to other sensed information sharing the same point in time when the sensed information becomes adapted. A value of one indicates the highest priority and larger values indicate lower priorities. The default value of the priority is one. If there are more than one sensed information

Name	*Definition*
	with the same priority, the order of process can be determined by the Adaptation engine itself. NOTE The priority might be used to apply the sensed information on the virtual world object characteristics – defined within a group of sensors – according to the capabilities of the adaptation VR. EXAMPLE The adaptation RV processes the individual sensed information of a group of sensors according to their priority in descending order due to its limited capabilities. That is, the sensed information with the lower priority might get lost.

6.11.5 Examples

This example shows the description of an acceleration sensing with the following semantics. The sensor has an ID of “AS001” and references “ASID001” and can mesure the values in three axis. The sensor shall be activated and the value shall be Ax="9.8" (m/s^2), Ay="4.9" (m/s^2), and Az="-4.9" (m/s^2). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
  <iidl:SensedInfoList>
    <iidl:SensedInfo       xsi:type="siv:AccelerationSensorType"       id="AS001"
sensorIdRef="ASID001" activate="true" axis="3">
    <iidl:TimeStamp      xsi:type="mpegvct:ClockTickTimeType"      timeScale="100"
pts="60000"/>
    <siv:Acceleration>
      <mpegvct:X>9.8</mpegvct:X>
      <mpegvct:Y>4.9</mpegvct:Y>
      <mpegvct:Z>-4.9</mpegvct:Z>
    </siv:Acceleration>
  </iidl:SensedInfo>
</iidl:SensedInfoList>
```

6.12 Orientation sensor type

6.12.1 General

This subclause specifies a sensor type which senses orientation. The orientation sensor type does not specify any sensing methods such as ultrasonic, optical, inercial and gyro technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The sensing properties of the sensor are specified in the orientation sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include robotics, natural user interface, security systems, and others.

6.12.2 Syntax

```
<!--########################################## -->
<!--Definition of orientation sensor type  -->
<!--########################################## -->
<complexType name="OrientationSensorType">
    <complexContent>
        <extension base="iidl:SensedInfoBaseType">
            <sequence>
                <element name="Orientation" type="mpegvct:Float3DVectorType"
                    minOccurs="0"/>
            </sequence>
            <attribute name="unit" type="mpegvct:unitType" use="optional"/>
        </extension>
    </complexContent>
</complexType>
```

6.12.3 Binary representation syntax

OrientationSensorType {	***Number of bits***	***Mnemonic***
UpdateMode	1	bslbf
if(UpdateMode ==0){		
OrientationSensorNormal		OrientationSensorNormalType
}else{		
OrientationSensorUpdate		OrientationSensorUpdateType
}		
}		

OrientationSensorNormalType{	***Number of bits***	***Mnemonic***
orientationFlag	1	bslbf
unitFlag	1	bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(orientationFlag) {		
orientation		Float3DVectorType
}		
if(unitFlag) {		

unit	8	bslbf
}		
}		

OrientationSensorUpdateModeType {	***Number of bits***	***Mnemonic***
TimeStampFlag	1	bslbf
IDFlag	1	bslbf
sensorIdRefFlag	1	bslbf
linkedlistFlag	1	bslbf
groupIDFlag	1	bslbf
priorityFlag	1	bslbf
activateFlag	1	bslbf
orientationFlag	1	bslbf
unitFlag	1	bslbf
ListUpdate	TimeStampFlag + IDFlag + sensorIdRefFlag + linkedlistFlag + groupIDFlag + priorityFlag + activateFlag + orientationFlag + unitFlag	bslbf
ListItemNum = 0		
if(TimeStampFlag){		
if(ListUpdate[ListItemNum]){		
TimeStamp		TimeStampType
}		
ListItemNum++		
}		
if(IDFlag){		
if(ListUpdate[ListItemNum]){		
ID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		

}		
if(sensorIdRefFlag){		
if(ListUpdate[ListItemNum]){		
sensorIdRef	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(linkedlistFlag){		
if(ListUpdate[ListItemNum]){		
linkedlist	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(groupIDFlag){		
if(ListUpdate[ListItemNum]){		
groupID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(priorityFlag){		
if(ListUpdate[ListItemNum]){		
priority	8	uimsbf
}		
ListItemNum++		
}		
if(activateFlag){		
if(ListUpdate[ListItemNum]){		
activate	1	bslbf
}		

ListItemNum++		
}		
if(orientationFlag){		
if(ListUpdate[ListItemNum]){		
orientation		Float3DVectorType
}		
ListItemNum++		
}		
if(unitFlag){		
if(ListUpdate[ListItemNum]){		
unit	8	bslbf
}		
}		
}		

6.12.4 Semantics

Semantics of the `OrientationSensorType`:

Name	*Definition*
`OrientationSensorType`	Tool for describing sensed information with respect to an orientation sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`Orientation`	Describes the sensed value by the orientation sensor in a three dimensional vector in the unit of degree. The orientation shall be measured as the inclined degree (orientation) with respect to the original pose. The original pose shall be the pose of the object sensed at the time of sensor activation. If a calibration has been performed on the orientation of the sensor after activation, the orientation after the calibration shall be considered as the original pose of the object.
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the `UnitTypeCS` is also defined in ISO/IEC 23005-6:—, A.2.1.

Name	*Definition*
`UpdateMode`	This field, which is only present in the binary representation, signals whether the sensed information is on the normal mode or on the update mode. A value of “1” means the update mode shall be used and “0” means the normal mode shall be used.
`OrientationSensorNormal`	This field is used to send the sensed information from an orientation sensor.
`OrientationSensorUpdate`	This field is used to send the sensed information from an orientation sensor only for the updated elements.
`OrientationSensorNormalType`	This field is used to send the sensed information from an orientation sensor on the normal mode.
`orientationFlag`	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.
`OrientationSensorUpdateType`	This field is used to send the sensed information from a orientation sensor on the update mode.
`ListUpdate`	Describes the updated list among all the active elements in the sensed information.
`priority`	Describes a priority for sensed information with respect to other sensed information sharing the same point in time when the sensed information becomes adapted. A value of one indicates the highest priority and larger values indicate lower priorities. The default value of the priority is one. If there are more than one sensed information with the same priority, the order of process can be determined by the Adaptation engine itself. NOTE The priority might be used to apply the sensed information on the virtual world object characteristics – defined within a group of sensors – according to the capabilities of the adaptation VR. EXAMPLE The adaptation RV processes the individual sensed information of a group of sensors according to their priority in descending order due to its limited capabilities. That is, the sensed information with the lower priority might get lost.

6.12.5 Examples

This example shows the description of an orientation sensing with the following semantics. The sensor has an ID of “OS001” and references “OSID001”. The sensor shall be activated and the value shall be Ox="6.0" (degrees), Oy="-3" (degrees), and Oz="3" (degrees). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:OrientationSensorType" id="OS001"
         sensorIdRef="OSID001" activate="true" unit="urn:mpeg:mpeg-v:01-CI-
UnitTypeCS-NS:radian">
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="60000"/>
         <siv:Orientation>
            <mpegvct:X>6.0</mpegvct:X>
            <mpegvct:Y>-3</mpegvct:Y>
            <mpegvct:Z>3</mpegvct:Z>
         </siv:Orientation>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.13 Angular velocity sensor type

6.13.1 General

This subclause specifies a sensor type which senses angular velocity. The angular velocity sensor type does not specify any sensing methods such as ultrasonic, optical, inertial and gyro technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The sensing properties of the sensor are specified in the angular velocity sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include robotics, natural user interface, security systems, and others.

6.13.2 Syntax

```
<!--############################################ -->
<!--Definition of angular velocity sensor type  -->
<!--############################################ -->
<complexType name="AngularVelocitySensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <sequence>
            <element name="AngularVelocity" type="mpegvct:Float3DVectorType"
               minOccurs="0"/>
         </sequence>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```


6.13.3 Binary representation syntax

AngularVelocitySensorType {	***Number of bits***	***Mnemonic***
UpdateMode	1	bslbf
if(UpdateMode ==0){		
AngularVelocitySensorNormal		AngularVelocitySensorNormalType
}else{		

AngularVelocitySensorUpdate		AngularVelocitySensorUpdateType
}		
}		

AngularVelocitySensorNormalType {	***Number of bits***	***Mnemonic***
angularvelocityFlag	1	Bslbf
unitFlag	1	Bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(angularvelocityFlag) {		
angularvelocity		Float3DVectorType
}		
if(unitFlag) {		
unit	8	Bslbf
}		
}		

AngularVelocitySensorUpdateType {	***Number of bits***	***Mnemonic***
TimeStampFlag	1	Bslbf
IDFlag	1	bslbf
sensorIdRefFlag	1	bslbf
linkedlistFlag	1	bslbf
groupIDFlag	1	bslbf
priorityFlag	1	bslbf
activateFlag	1	bslbf
angularvelocityFlag	1	bslbf
unitFlag	1	bslbf
ListUpdate	TimeStampFlag + IDFlag + sensorIdRefFlag + linkedlistFlag + groupIDFlag + priorityFlag +	bslbf

	activateFlag + angularvelocityFlag + unitFlag	
ListItemNum = 0		
if(TimeStampFlag){		
if(ListUpdate[ListItemNum]){		
TimeStamp		TimeStampType
}		
ListItemNum++		
}		
if(IDFlag){		
if(ListUpdate[ListItemNum]){		
ID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(sensorIdRefFlag){		
if(ListUpdate[ListItemNum]){		
sensorIdRef	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(linkedlistFlag){		
if(ListUpdate[ListItemNum]){		
linkedlist	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(groupIDFlag){		
if(ListUpdate[ListItemNum]){		
groupID	See ISO/IEC 10646	UTF-8

}		
ListItemNum++		
}		
if(priorityFlag){		
if(ListUpdate[ListItemNum]){		
priority	8	uimsbf
}		
ListItemNum++		
}		
if(activateFlag){		
if(ListUpdate[ListItemNum]){		
activate	1	bslbf
}		
ListItemNum++		
}		
if(angularvelocityFlag){		
if(ListUpdate[ListItemNum]){		
angularvelocity		Float3DVectorType
}		
ListItemNum++		
}		
if(unitFlag){		
if(ListUpdate[ListItemNum]){		
unit	8	bslbf
}		
}		
}		

6.13.4 Semantics

Semantics of the `AngularVelocitySensorType`:

Name	*Definition*
AngularVelocitySensorType	Tool for describing sensed information with respect to with respect to an angular velocity sensor.
TimeStamp	Describes the time that the information is acquired (sensed).
AngularVelocity	Describes the sensed value by the Angular Velocity sensor in a three dimensional vector with respect to degree per second.
unit	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the mpeg7:termReferenceType defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the UnitTypeCS defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the UnitTypeCS is also defined in ISO/IEC 23005-6:—, A.2.1.
UpdateMode	This field, which is only present in the binary representation, signals whether the sensed information is on the normal mode or on the update mode. A value of “1” means the update mode shall be used and “0” means the normal mode shall be used.
AngularVelocitySensorNormal	This field is used to send the sensed information from an angular velocity sensor.
AngularVelocitySensorUpdate	This field is used to send the sensed information from an angular velocity sensor only for the updated elements.
AngularVelocitySensorNormalType	This field is used to send the sensed information from an angular velocity sensor on the normal mode.
angularvelocityFlag	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
unitFlag	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.
SensedInfoBaseType	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.
AngularVelocitySensorUpdateType	This field is used to send the sensed information from an angular velocity sensor on the update mode.
ListUpdate	Describes the updated list among all the active elements in the sensed information.
priority	Describes a priority for sensed information with respect to other sensed information sharing the same point in time when the sensed information becomes adapted. A value of one indicates the highest priority and larger values indicate lower priorities. The default value of the priority is one. If there are more than one sensed information with the same priority, the order of process can be determined by the Adaptation engine itself. NOTE The priority might be used to apply the sensed information on the virtual world object characteristics – defined within a group of sensors – according to the capabilities of the adaptation VR. EXAMPLE The adaptation RV processes the individual sensed information of a group of sensors according to their priority in descending order due to its limited capabilities. That is, the sensed information with the lower priority might get lost.

6.13.5 Examples

This example shows the description of an angular velocity sensing with the following semantics. The sensor has an ID of “AVS001” and references “AVSID001”. The sensor shall be activated and the value shall be AVx="6" (degree/s), AVy="-4" (degree/s), and AVz="15" (degrees/s). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:AngularVelocitySensorType" id="AVS001"
         sensorIdRef="AVSID001" activate="true" unit="urn:mpeg:mpeg-v:01-CI-
UnitTypeCS-NS:radpersec">
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="60000"/>
         <siv:AngularVelocity>
            <mpegvct:X>6.0</mpegvct:X>
            <mpegvct:Y>-4.0</mpegvct:Y>
            <mpegvct:Z>15.0</mpegvct:Z>
         </siv:AngularVelocity>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.14 Angular acceleration sensor type

6.14.1 General

This subclause specifies a sensor type which senses angular acceleration. The angular acceleration sensor type does not specify any sensing methods such as ultrasonic, optical, inercial and gyro technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The sensing properties of the sensor are specified in the angular acceleration sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include robotics, natural user interface, security systems, and others.

6.14.2 Syntax

```
<!--################################################ -->
<!--Definition of angular acceleration sensor type  -->
<!--################################################ -->
<complexType name="AngularAccelerationSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <sequence>
            <element name="AngularAcceleration"
               type="mpegvct:Float3DVectorType" minOccurs="0"/>
         </sequence>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.14.3 Binary representation syntax

AngularAccelerationSensorType {	***Number of bits***	***Mnemonic***
UpdateMode	1	bslbf
if(UpdateMode ==0){		
AngularAccelerationSensorNormal		AngularAccelerationSensorNormalType
}else{		
AngularAccelerationSensorUpdate		AngularAccelerationSensorUpdateType
}		
}		

AngularAccelerationSensorType{	***Number of bits***	***Mnemonic***
angularaccelerationFlag	1	bslbf
unitFlag	1	bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(angularaccelerationFlag) {		
angularacceleration		Float3DVectorType
}		
if(unitFlag) {		
unit	8	bslbf
}		
}		

AngularAccelerationSensorUpdateType {	***Number of bits***	***Mnemonic***
TimeStampFlag	1	bslbf
IDFlag	1	bslbf
sensorIdRefFlag	1	bslbf
linkedlistFlag	1	bslbf

groupIDFlag	1	bslbf
priorityFlag	1	bslbf
activateFlag	1	bslbf
angularaccelerationFlag	1	bslbf
unitFlag	1	bslbf
ListUpdate	TimeStampFlag + IDFlag + sensorIdRefFlag + linkedlistFlag + groupIDFlag + priorityFlag + activateFlag + angularacceleration Flag + unitFlag	bslbf
ListItemNum = 0		
if(TimeStampFlag){		
if(ListUpdate[ListItemNum]){		
TimeStamp		TimeStampType
}		
ListItemNum++		
}		
if(IDFlag){		
if(ListUpdate[ListItemNum]){		
ID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(sensorIdRefFlag){		
if(ListUpdate[ListItemNum]){		
sensorIdRef	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		

if(linkedlistFlag){		
if(ListUpdate[ListItemNum]){		
linkedlist	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(groupIDFlag){		
if(ListUpdate[ListItemNum]){		
groupID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(priorityFlag){		
if(ListUpdate[ListItemNum]){		
priority	8	uimsbf
}		
ListItemNum++		
}		
if(activateFlag){		
if(ListUpdate[ListItemNum]){		
activate	1	bslbf
}		
ListItemNum++		
}		
if(angularaccelerationFlag){		
if(ListUpdate[ListItemNum]){		
angularacceleration		Float3DVectorType
}		
ListItemNum++		

}		
if(unitFlag){		
if(ListUpdate[ListItemNum]){		
unit	8	bslbf
}		
}		
}		

6.14.4 Semantics

Semantics of the `AngularAccelerationSensorType`:

Name	*Definition*
`AngularAcceleration SensorType`	Tool for describing sensed information with respect to an angular acceleration sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`AngularAcceleration`	Describes the sensed value by the Angular Acceleration sensor in a three dimensional vector with respect to degree per second squared.
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the `UnitTypeCS` is also defined in ISO/IEC 23005-6:—, A.2.1.
`UpdateMode`	This field, which is only present in the binary representation, signals whether the sensed information is on the normal mode or on the update mode. A value of “1” means the update mode shall be used and “0” means the normal mode shall be used.
`AngularAccelerationSensor Normal`	This field is used to send the sensed information from an angular acceleration sensor.
`AngularAccelerationSensor Update`	This field is used to send the sensed information from an angular acceleration sensor only for the updated elements.
`AngularAccelerationSensor NormalType`	This field is used to send the sensed information from an angular acceleration sensor on the normal mode.
`angularaccelerationFlag`	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.

Name	*Definition*
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.
`AngularAccelerationSensorUpdateType`	This field is used to send the sensed information from an angular acceleration sensor on the update mode.
`ListUpdate`	Describes the updated list among all the active elements in the sensed information.
`priority`	Describes a priority for sensed information with respect to other sensed information sharing the same point in time when the sensed information becomes adapted. A value of one indicates the highest priority and larger values indicate lower priorities. The default value of the priority is one. If there are more than one sensed information with the same priority, the order of process can be determined by the adaptation engine itself. NOTE The priority might be used to apply the sensed information on the virtual world object characteristics – defined within a group of sensors – according to the capabilities of the adaptation VR. EXAMPLE The adaptation RV processes the individual sensed information of a group of sensors according to their priority in descending order due to its limited capabilities. That is, the sensed information with the lower priority might get lost.

6.14.5 Examples

This example shows the description of an angular acceleration sensing with the following semantics. The description has identifier of “aas001” and the sensor references an actual sensor with ID of “aas0001”. The sensor shall be activated and the value shall be AVx="10.0" (rad/s^2), AVy="1.0" (rad/s^2), and AVz="20" (rad/s^2). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:AngularAccelerationSensorType" id="aas001"
         sensorIdRef="aas0001" activate="true" unit="urn:mpeg:mpeg-v:01-CI-
UnitTypeCS-NS:radpersecsquare">
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="60000"/>
         <siv:AngularAcceleration>
            <mpegvct:X>10.0</mpegvct:X>
            <mpegvct:Y>10.0</mpegvct:Y>
            <mpegvct:Z>20.0</mpegvct:Z>
         </siv:AngularAcceleration>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.15 Force sensor type

6.15.1 General

This subclause specifies a sensor type which senses force. The force sensor type does not specify any sensing methods such as ultrasonic, optical, and inductive technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The sensing properties of the sensor are specified in the force sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include robotics, user interface, haptics, and telepresence.

6.15.2 Syntax

```
<!--##################################### -->
<!--Definition of force sensor type      -->
<!--##################################### -->
<complexType name="ForceSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <sequence>
            <element name="Force" type="mpegvct:Float3DVectorType"
               minOccurs="0"/>
         </sequence>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.15.3 Binary representation syntax

ForceSensorType{	***Number of bits***	***Mnemonic***
forceFlag	1	Bslbf
unitFlag	1	Bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(forceFlag) {		
force		Float3DVectorType
}		
if(unitFlag) {		
unit	8	bslbf
}		
}		

6.15.4 Semantics

Semantics of the `ForceSensorType`:

Name	*Definition*
`ForceSensorType`	Tool for describing sensed information with respect to a force sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`Force`	Describes the sensed value by the force sensor in a three dimensional vector with respect to N (Newton).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the `UnitTypeCS` is also defined in ISO/IEC 23005-6:—, A.2.1.
`forceFlag`	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.

6.15.5 Examples

This example shows the description of a force sensing with the following semantics. The description has identifier of “fst01” and the sensor references an actual sensor with ID of “fst001”. The device shall be activated and the value shall be x="10.0", y="1.0", and z="20" (Newton). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:ForceSensorType" id="fst01"
         sensorIdRef="fst001" activate="true">
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="60000"/>
         <siv:Force>
            <mpegvct:X>10.0</mpegvct:X>
            <mpegvct:Y>1.0</mpegvct:Y>
            <mpegvct:Z>20.0</mpegvct:Z>
         </siv:Force>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.16 Torque sensor type

6.16.1 General

This subclause specifies a sensor type which senses torque. The torque sensor type does not specify any sensing methods such as ultrasonic, optical, and inductive technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The sensing properties of the sensor are specified in the torque sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include robotics, user interface, haptics, and telepresence.

6.16.2 Syntax

```
<!--##################################### -->
<!--Definition of torque sensor type     -->
<!--##################################### -->
<complexType name="TorqueSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <sequence>
            <element name="Torque" type="mpegvct:Float3DVectorType"
               minOccurs="0"/>
         </sequence>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.16.3 Binary representation syntax

TorqueSensorType{	***Number of bits***	***Mnemonic***
TorqueFlag	1	bslbf
unitFlag	1	bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(TorqueFlag) {		
Torque		Float3DVectorType
}		
if(unitFlag) {		
unit	8	bslbf
}		
}		

6.16.4 Semantics

Semantics of the `TorqueSensorType`:

Name	*Definition*
`TorqueSensorType`	Tool for describing sensed information with respect to a torque sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`Torque`	Describes the sensed value by the torque sensor in a three dimensional vector with respect to N-mm (Newton millimetre).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the `UnitTypeCS` is also defined in ISO/IEC 23005-6:—, A.2.1.
`TorqueFlag`	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.

6.16.5 Examples

This example shows the description of a torque sensing with the following semantics. The description has identifier of “`mytorq01`” and the sensor references an actual sensor with ID of “`fttorque`”. The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second. The value shall be x="10.0", y="15.0", and z="14" (Newton millimetre).

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:TorqueSensorType" id="mytorq01"
         sensorIdRef="fttorque">
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="60000"/>
         <siv:Torque>
            <mpegvct:X>10.0</mpegvct:X>
            <mpegvct:Y>15.0</mpegvct:Y>
            <mpegvct:Z>14.0</mpegvct:Z>
         </siv:Torque>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.17 Pressure sensor type

6.17.1 General

This Subclause specifies a sensor type which senses pressure. The pressure sensor type does not specify any sensing methods such as as the capacitive, resisitive, and conductivity technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The sensing properties of the sensor are specified in the pressure sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include multisensorial effect control, environmental monitoring, and others.

6.17.2 Syntax

```
<!--################################### -->
<!--Definition of pressure sensor type  -->
<!--################################### -->
<complexType name="PressureSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="value" type="float" use="optional"/>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.17.3 Binary representation syntax

PressureSensorType{	***Number of bits***	***Mnemonic***
valueFlag	1	bslbf
unitFlag	1	bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(valueFlag) {		
value	32	fsbf
}		
if(unitFlag) {		
unit	8	bslbf
}		
}		

6.17.4 Semantics

Semantics of the `PressureSensorType`:

Name	*Definition*
`PressureSensorType`	Tool for describing sensed information with respect to a pressure sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. If the unit is not defined here, the default unit is N/mm^2 (Newton/millimetre squared). The binary representation of the `UnitTypeCS` is also defined in ISO/IEC 23005-6:—, A.2.1.
`value`	Describes the sensed pressure value by the pressure with respect to the default unit or the unit defined in the unit attribute.
`valueFlag`	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.

6.17.5 Examples

This example shows the description of a pressure sensing with the following semantics. The description has identifier of “press01” and the sensor references an actual sensor with ID of “fff”. The sensed pressure is 0.1 N/mm^2. The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:PressureSensorType" activate="true"
         id="press01" sensorIdRef="fff" value="0.1">
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="60000"/>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.18 Motion sensor type

6.18.1 General

This Subclause specifies an aggregated sensor type which contains sensed information such as position, velocity, acceleration, orientation, angular velocity, and angular acceleration. The aggregated sensor type may contain just a subset of the sensed information. Moreover, the motion sensor type does not specify any sensing methods such as ultrasonic, MEMS sensor-based and camera-based technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The sensing properties of the sensor are specified in the motion sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include motion-based computer games, and others.

6.18.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of motion sensor type                -->
<!-- ################################################ -->
<complexType name="MotionSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <sequence>
            <element name="Position" type="siv:PositionSensorType"
               minOccurs="0"/>
            <element name="Orientation" type="siv:OrientationSensorType"
               minOccurs="0"/>
            <element name="Velocity" type="siv:VelocitySensorType"
               minOccurs="0"/>
            <element name="AngularVelocity"
               type="siv:AngularVelocitySensorType" minOccurs="0"/>
            <element name="Acceleration" type="siv:AccelerationSensorType"
               minOccurs="0"/>
            <element name="AngularAcceleration"
               type="siv:AngularAccelerationSensorType" minOccurs="0"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>
```

6.18.3 Binary representation syntax

MotionSensorType {	***Number of bits***	***Mnemonic***
UpdateMode	1	bslbf
if(UpdateMode ==0){		
MotionSensorNormal		MotionSensorNormalType
}else{		
MotionSensorUpdate		MotionSensorUpdateType
}		
}		

MotionSensorNormalType{	***Number of bits***	***Mnemonic***
positionFlag	1	bslbf
orientationFlag	1	bslbf
velocityFlag	1	bslbf
angularVelocityFlag	1	bslbf
accelerationFlag	1	bslbf
angularAccelerationFlag	1	bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(positionFlag) {		
Position		PositionSensorType
}		
if(orientationFlag) {		
Orientation		OrientationSensorType
}		
if(velocityFlag) {		
Velocity		VelocitySensorType
}		
if(angularVelocityFlag) {		
AngularVelocity		AngularVelocitySensorType
}		
if(accelerationFlag) {		
Acceleration		AccelerationSensorType
}		
if(angularAccelerationFlag) {		
AngularAcceleration		AngularAccelerationSensorType
}		

MotionSensorUpdateType {	***Number of bits***	***Mnemonic***
TimeStampFlag	1	bslbf
IDFlag	1	bslbf
sensorIdRefFlag	1	bslbf
linkedlistFlag	1	bslbf
groupIDFlag	1	bslbf
priorityFlag	1	bslbf
activateFlag	1	bslbf
positionFlag	1	bslbf
orientationFlag	1	bslbf
velocityFlag	1	bslbf
angularVelocityFlag	1	bslbf
accelerationFlag	1	bslbf
angularAccelerationFlag	1	bslbf
ListUpdate	TimeStampFlag + IDFlag + sensorIdRefFlag + linkedlistFlag + groupIDFlag + priorityFlag + activateFlag + positionFlag + orientationFlag + velocityFlag + angularVelocityFlag + accelerationFlag + angularAccelerationFlag	bslbf
ListItemNum = 0		
if(TimeStampFlag){		
if(ListUpdate[ListItemNum]){		
TimeStamp		TimeStampType
}		
ListItemNum++		
}		
if(IDFlag){		
if(ListUpdate[ListItemNum]){		

ID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(sensorIdRefFlag){		
if(ListUpdate[ListItemNum]){		
sensorIdRef	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(linkedlistFlag){		
if(ListUpdate[ListItemNum]){		
linkedlist	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(groupIDFlag){		
if(ListUpdate[ListItemNum]){		
groupID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(priorityFlag){		
if(ListUpdate[ListItemNum]){		
priority	8	uimsbf
}		
ListItemNum++		
}		
if(activateFlag){		

if(ListUpdate[ListItemNum]){		
activate	1	bslbf
}		
ListItemNum++		
}		
if(positionFlag){		
if(ListUpdate[ListItemNum]){		
Position		PositionSensorUpdateType
}		
ListItemNum++		
}		
if(orientationFlag){		
if(ListUpdate[ListItemNum]){		
Orientation		OrientationSensorUpdateType
}		
ListItemNum++		
}		
if(velocityFlag){		
if(ListUpdate[ListItemNum]){		
Velocity		VelocitySensorUpdateType
}		
ListItemNum++		
}		
if(angularVelocityFlag){		
if(ListUpdate[ListItemNum]){		
AngularVelocity		AngularVelocitySensorUpdateType
}		

ListItemNum++		
}		
if(accelerationFlag){		
if(ListUpdate[ListItemNum]){		
Acceleration		AccelerationSensorUpdateType
}		
ListItemNum++		
}		
if(angularAccelerationFlag){		
if(ListUpdate[ListItemNum]){		
AngularAcceleration		AngularAccelerationSensorUpdateType
}		
}		
}		

6.18.4 Semantics

Semantics of the `MotionSensorType`:

Name	*Definition*
`MotionSensorType`	Tool for describing a motion sensed information.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`Position`	Describes the sensed vector value of the position.
`Orientation`	Describes the sensed vector value of the orientation.
`Velocity`	Describes the sensed vector value of the velocity.
`AngularVelocity`	Describes the sensed vector value of the angular velocity.
`Acceleration`	Describes the sensed vector value of the acceleration.

Name	*Definition*
AngularAcceleration	Describes the sensed vector value of the angular acceleration.
UpdateMode	This field, which is only present in the binary representation, signals whether the sensed information is on the normal mode or on the update mode. A value of “1” means the update mode shall be used and “0” means the normal mode shall be used.
MotionSensorNormal	This field is used to send the sensed information from a motion sensor.
MotionSensorUpdate	This field is used to send the sensed information from a motion sensor only for the updated elements.

Semantics of the MotionSensorNormalType:

Name	*Definition*
MotionSensorNormalType	This field is used to send the sensed information from a motion sensor on the normal mode.
positionFlag	This field, which is only present in the binary representation, signals the presence of position value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
orientationFlag	This field, which is only present in the binary representation, signals the presence of orientation value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
velocityFlag	This field, which is only present in the binary representation, signals the presence of velocity value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
angularvelocityFlag	This field, which is only present in the binary representation, signals the presence of angular velocity value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
accelerationFlag	This field, which is only present in the binary representation, signals the presence of acceleration value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
angularaccelerationFlag	This field, which is only present in the binary representation, signals the presence of angular acceleration value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
SensedInfoBaseType	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.

Semantics of the `MotionSensorUpdateType`:

Name	*Definition*
`MotionSensorUpdateType`	This field is used to send the sensed information from a motion sensor on the update mode.
`TimeStampFlag`	This field, which is only present in the binary representation, signals the presence of TimeStamp element in the update mode. A value of “1” means that the element shall included in the updated information and “0” means the element is not updated and shall not be used.
`IDFlag`	This field, which is only present in the binary representation, signals the presence of ID attribute in the update mode. A value of “1” means that the attribute shall included in the updated information and “0” means the element is not updated and shall not be used.
`sensorIdRefFlag`	This field, which is only present in the binary representation, signals the presence of `sensorIdRef` attribute in the update mode. A value of “1” means that the attribute shall included in the updated information and “0” means the element is not updated and shall not be used.
`linkedlistFlag`	This field, which is only present in the binary representation, signals the presence of `linkedlist` attribute in the update mode. A value of “1” means that the attribute shall included in the updated information and “0” means the element is not updated and shall not be used.
`groupIDFlag`	This field, which is only present in the binary representation, signals the presence of `groupID` attribute in the update mode. A value of “1” means that the attribute shall included in the updated information and “0” means the element is not updated and shall not be used.
`priorityFlag`	This field, which is only present in the binary representation, signals the presence of `priority` attribute in the update mode. A value of “1” means that the attribute shall included in the updated information and “0” means the element is not updated and shall not be used.
`activateFlag`	This field, which is only present in the binary representation, signals the presence of `activate` attribute in the update mode. A value of “1” means that the attribute shall included in the updated information and “0” means the element is not updated and shall not be used.
`positionFlag`	This field, which is only present in the binary representation, signals the presence of `position` attribute in the update mode. A value of “1” means that the attribute shall included in the updated information and “0” means the element is not updated and shall not be used.
`orientationFlag`	This field, which is only present in the binary representation, signals the presence of `orientation` attribute in the update mode. A value of “1” means that the attribute shall included in the updated information and “0” means the element is not updated and shall not be used.

velocityFlag	This field, which is only present in the binary representation, signals the presence of velocity attribute in the update mode. A value of “1” means that the attribute shall included in the updated information and “0” means the element is not updated and shall not be used.
angularVelocityFlag	This field, which is only present in the binary representation, signals the presence of angularVelocity attribute in the update mode. A value of “1” means that the attribute shall included in the updated information and “0” means the element is not updated and shall not be used.
accelerationFlag	This field, which is only present in the binary representation, signals the presence of acceleration attribute in the update mode. A value of “1” means that the attribute shall included in the updated information and “0” means the element is not updated and shall not be used.
angularAccelerationFlag	This field, which is only present in the binary representation, signals the presence of angularAcceleration attribute in the update mode. A value of “1” means that the attribute shall included in the updated information and “0” means the element is not updated and shall not be used.
ListUpdate	Describes the updated list among all the active elements in the sensed information.
activate	Describes whether this device is to be activated or not.
Position	Describes the sensed vector value of the position.
Orientation	Describes the sensed vector value of the orientation.
Velocity	Describes the sensed vector value of the velocity.
AngularVelocity	Describes the sensed vector value of the angular velocity.
Acceleration	Describes the sensed vector value of the acceleration.
AngularAcceleration	Describes the sensed vector value of the angular acceleration.

6.18.5 Examples

This example shows the description of a motion sensing with the following semantics. The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:MotionSensorType" id="MS001"
         sensorIdRef="MSID001" activate="true">
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="60000"/>
         <siv:Position xsi:type="siv:PositionSensorType" >
            <siv:Position>
               <mpegvct:X>1.5</mpegvct:X>
               <mpegvct:Y>0.5</mpegvct:Y>
               <mpegvct:Z>-2.1</mpegvct:Z>
            </siv:Position>
         </siv:Position>
```

```
          <siv:Orientation xsi:type="siv:OrientationSensorType"
unit="urn:mpeg:mpeg-v:01-CI-UnitTypeCS-NS:radian">
              <siv:Orientation>
                  <mpegvct:X>2.0</mpegvct:X>
                  <mpegvct:Y>-0.5</mpegvct:Y>
                  <mpegvct:Z>1.0</mpegvct:Z>
              </siv:Orientation>
          </siv:Orientation>
          <siv:Velocity xsi:type="siv:VelocitySensorType" >
              <siv:Velocity>
                  <mpegvct:X>10.0</mpegvct:X>
                  <mpegvct:Y>5.0</mpegvct:Y>
                  <mpegvct:Z>0.1</mpegvct:Z>
              </siv:Velocity>
          </siv:Velocity>
          <siv:AngularVelocity xsi:type="siv:AngularVelocitySensorType"
              unit="urn:mpeg:mpeg-v:01-CI-UnitTypeCS-NS:radpersec">
              <siv:AngularVelocity>
                  <mpegvct:X>2.0</mpegvct:X>
                  <mpegvct:Y>-0.5</mpegvct:Y>
                  <mpegvct:Z>1.0</mpegvct:Z>
              </siv:AngularVelocity>
          </siv:AngularVelocity>
          <siv:Acceleration xsi:type="siv:AccelerationSensorType">
              <siv:Acceleration>
                  <mpegvct:X>9.8</mpegvct:X>
                  <mpegvct:Y>4.9</mpegvct:Y>
                  <mpegvct:Z>-4.9</mpegvct:Z>
              </siv:Acceleration>
          </siv:Acceleration>
          <siv:AngularAcceleration xsi:type="siv:AngularAccelerationSensorType"
              unit="urn:mpeg:mpeg-v:01-CI-UnitTypeCS-NS:radpersecsquare">
              <siv:AngularAcceleration>
                  <mpegvct:X>150.0</mpegvct:X>
                  <mpegvct:Y>-100.0</mpegvct:Y>
                  <mpegvct:Z>50.0</mpegvct:Z>
              </siv:AngularAcceleration>
          </siv:AngularAcceleration>
      </iidl:SensedInfo>
  </iidl:SensedInfoList>
</iidl:InteractionInfo>
```


6.19 Intelligent camera type

6.19.1 General

This subclause specifies a camera sensor type which is capable of sensing information such as facial expressions, body gestures, facial and body feature points. The sensor type may contain just a subset of the sensed information. The sensor type does not specify if the camera uses marker or not to detect such information. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The properties of the sensor are specified in the intelligent camera capability in ISO/IEC 23005-2. The applications of the sensor type may include robotics, natural user interface, and others.

6.19.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of intelligent camera type           -->
<!-- ################################################ -->
<complexType name="IntelligentCameraType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <sequence>
            <element name="FacialAnimationID" type="anyURI" minOccurs="0"/>
            <element name="BodyAnimationID" type="anyURI" minOccurs="0"/>
            <element name="FaceFeature" type="mpegvct:Float3DVectorType"
               minOccurs="0" maxOccurs="255"/>
            <element name="BodyFeature" type="mpegvct:Float3DVectorType"
               minOccurs="0" maxOccurs="255"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>
```

6.19.3 Binary representation syntax

IntelligentCameraType{	***Number of bits***	***Mnemonic***
UpdateMode	1	bslbf
if(UpdateMode ==0){		
IntelligentCameraNormal		IntelligentCameraNormalType
}else{		
IntelligentCameraUpdate		IntelligentCameraUpdateType
}		
}		

IntelligentCameraNormalType{	***Number of bits***	***Mnemonic***
FacialIDFlag	1	bslbf
BodyIDFlag	1	bslbf
FaceFeatureFlag	1	bslbf
BodyFeatureFlag	1	bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(FacialIDFlag) {		

FacialAnimationID	See ISO/IEC 10646	UTF-8
}		
if(BodyIDFlag) {		
BodyAnimationID	See ISO/IEC 10646	UTF-8
}		
if(FaceFeatureFlag) {		
NumOfFaceFeature	8	uimsbf
for(k=0; k<NumOfFaceFeature; k++) {		
FaceFeature[k]		Float3DVectorType
}		
}		
if(BodyFeatureFlag) {		
NumOfBodyFeature	8	uimsbf
for(k=0; k<NumOfBodyFeature; k++) {		
BodyFeature[k]		Float3DVectorType
}		
}		
}		

IntelligentCameraUpdateType {	***Number of bits***	***Mnemonic***
TimeStampFlag	1	bslbf
IDFlag	1	bslbf
sensorIdRefFlag	1	bslbf
linkedlistFlag	1	bslbf
groupIDFlag	1	bslbf
priorityFlag	1	bslbf

activateFlag	1	bslbf
FacialIDFlag	1	bslbf
BodyIDFlag	1	bslbf
FaceFeatureFlag	1	bslbf
BodyFeatureFlag	1	bslbf
ListUpdate	TimeStampFlag + IDFlag + sensorIdRefFlag + linkedlistFlag + groupIDFlag + priorityFlag + activateFlag + FacialIDFlag + BodyIDFlag + FaceFeatureFlag + BodyFeatureFlag	bslbf
ListItemNum = 0		
if(TimeStampFlag){		
if(ListUpdate[ListItemNum]){		
TimeStamp		TimeStampType
}		
ListItemNum++		
}		
if(IDFlag){		
if(ListUpdate[ListItemNum]){		
ID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(sensorIdRefFlag){		
if(ListUpdate[ListItemNum]){		
sensorIdRef	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(linkedlistFlag){		

if(ListUpdate[ListItemNum]){		
linkedlist	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(groupIDFlag){		
if(ListUpdate[ListItemNum]){		
groupID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(priorityFlag){		
if(ListUpdate[ListItemNum]){		
priority	8	uimsbf
}		
ListItemNum++		
}		
if(activateFlag){		
if(ListUpdate[ListItemNum]){		
activate	1	bslbf
}		
ListItemNum++		
}		
if(FacialIDFlag){		
if(ListUpdate[ListItemNum]){		
FacialAnimationID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		

if(BodyIDFlag){		
if(ListUpdate[ListItemNum]){		
BodyAnimationID	See ISO/IEC 10646	UTF-8
}		
ListItemNum++		
}		
if(FaceFeatureFlag)		
if(ListUpdate[ListItemNum]){		
NumOfFaceFeature	8	uimsbf
UpdateMaskFace	NumOfFaceFeature	bslbf
for(k=0;k< NumOfFaceFeature; k++){		
if(UpdateMaskFace[k]){		
FaceFeature[k]		Float3DVectorType
}		
}		
}		
if(BodyFeatureFlag)		
if(ListUpdate[ListItemNum]){		
NumOfBodyFeature	8	uimsbf
UpdateMaskBody	NumOfBodyFeature	bslbf
for(k=0;k< NumOfBodyFeature; k++){		
if(UpdateMaskBody[k]){		
BodyFeature[k]		Float3DVectorType
}		
}		
}		
}		
}		

6.19.4 Semantics

Semantics of the `IntelligentCameraType`:

Name	*Definition*
`IntelligentCameraSensorType`	Tool for describing an intelligent camera.
`FacialAnimationID`	Describes the ID referencing the facial expression defined in the `FacialExpressionAnimationCS` in ISO/IEC 23005-4:2018, A.2.3.12.
`BodyAnimationID`	Describes the ID referencing the body expression defined in the `BodyExpressionAnimationCS` in ISO/IEC 23005-4:2018, A.2.3.13.
`FaceFeature`	Describes the 3D position of each of the face feature points detected by the camera. NOTE The order of the elements corresponds to the order of the face feature points defined at the `featureControl` for face in ISO/IEC 23005-4:2018, 5.2.8.7.
`BodyFeature`	Describes the 3D position of each of the body feature points detected by the camera. NOTE The order of the elements corresponds to the order of the body feature points defined at the `featureControl` for body in ISO/IEC 23005-4:2018, 5.2.8.6.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`UpdateMode`	This field, which is only present in the binary representation, signals whether the sensed information is on the normal mode or on the update mode. A value of “1” means the update mode shall be used and “0” means the normal mode shall be used.
`IntelligentCameraNormal`	This field is used to send the sensed information from an intelligent camera.
`IntelligentCameraUpdate`	This field is used to send the sensed information from an intelligent camera only for the updated elements.
`IntelligentCameraNormalType`	This field is used to send the sensed information from an intelligent camera on the normal mode.
`FacialIDFlag`	This field, which is only present in the binary representation, signals the presence of the facial animation ID. A value of “1” means the facial animation ID mode shall be used and “0” means the facial animation ID mode shall not be used.
`BodyIDFlag`	This field, which is only present in the binary representation, signals the presence of the body animation ID. A value of “1” means the body animation ID mode shall be used and “0” means the body animation ID mode shall not be used.

Name	*Definition*
`FaceFeatureFlag`	This field, which is only present in the binary representation, signals the presence of the face features. A value of “1” means the face feature tracking mode shall be used and “0” means the face feature tracking mode shall not be used.
`BodyFeatureFlag`	This field, which is only present in the binary representation, signals the presence of the body features. A value of “1” means the body feature tracking mode shall be used and “0” means the body feature tracking mode shall not be used.
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.
`FacialAnimationIDLength`	This field, which is only present in the binry representation, specifies the length of the `FacialAnimationIDLength` field encoded in binary representation in bytes.
`BodyAnimationIDLength`	This field, which is only present in the binry representation, specifies the length of the `BodyAnimationIDLength` field encoded in binary representation in bytes.
`NumOfFaceFeature`	This field, which is only present in the binary mode, describes the number of FaceFeature points.
`NumOfBodyFeature`	This field, which is only present in the binary mode, describes the number of BodyFeature points.
`IntelligentCameraUpdateType`	This field is used to send the sensed information from an intelligent camera on the update mode.
`ListUpdate`	Describes the updated list among all the active elements in the sensed information.
`UpdateMaskBody`	This field, which is only present in the binary syntax, specifies a bit-field that indicates whether the updated value is assigned to the corresponding partition.
`UpdateMaskFace`	This field, which is only present in the binary syntax, specifies a bit-field that indicates whether the updated value is assigned to the corresponding partition.

6.19.5 Examples

This example shows the description of an intelligent camera sensing with the following semantics. The information from the intelligent camera with the ID of ICS002 shall be sensed at timestamp="60000" where there are 100 clock ticks per second. The 1st Point of the head outline is located in 3D at (0.0, 0.0, 0.0). The 2nd Point of the head outline is located in 3D at (0.01, 0.0, 0.01). … the 4th point of the mouse lips is located in 3D at (0.05, 0.1, 0.01). The 3D point of the head skull is located at (0.0, 0.5, -0.05). The 3D point of the left clavicle is located at (0.0, 0.4, -0.04). … The 3D point of the left foot is located at (-0.3, 1.2, -0.04).

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:IntelligentCameraType" id="ICS002"
         activate="true">
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="60000"/>
         <siv:FaceFeature>
            <mpegvct:X>0.0</mpegvct:X>
            <mpegvct:Y>0.0</mpegvct:Y>
            <mpegvct:Z>0.0</mpegvct:Z>
         </siv:FaceFeature>
         <siv:FaceFeature>
            <mpegvct:X>0.01</mpegvct:X>
            <mpegvct:Y>0.0</mpegvct:Y>
            <mpegvct:Z>0.01</mpegvct:Z>
         </siv:FaceFeature>
         <siv:FaceFeature>
            <mpegvct:X>0.05</mpegvct:X>
            <mpegvct:Y>0.1</mpegvct:Y>
            <mpegvct:Z>0.01</mpegvct:Z>
         </siv:FaceFeature>
         <siv:FaceFeature>
            <mpegvct:X>0.0</mpegvct:X>
            <mpegvct:Y>0.5</mpegvct:Y>
            <mpegvct:Z>-0.05</mpegvct:Z>
         </siv:FaceFeature>
         <siv:BodyFeature>
            <mpegvct:X>0.0</mpegvct:X>
            <mpegvct:Y>0.5</mpegvct:Y>
            <mpegvct:Z>-0.05</mpegvct:Z>
         </siv:BodyFeature>
         <siv:BodyFeature>
            <mpegvct:X>0.0</mpegvct:X>
            <mpegvct:Y>0.4</mpegvct:Y>
            <mpegvct:Z>-0.04</mpegvct:Z>
         </siv:BodyFeature>
         <siv:BodyFeature>
            <mpegvct:X>-0.3</mpegvct:X>
            <mpegvct:Y>1.2</mpegvct:Y>
            <mpegvct:Z>-0.04</mpegvct:Z>
         </siv:BodyFeature>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```


6.20 Multi interaction point sensor type

6.20.1 General

This subclause specifies a sensor type which senses a set of states for multiple interaction points which receivce user's selecting inputs. For instance, the current touch-based devices provide lots of multi-touch applications while some of such applications may include a set of buttons, which are supposed to receive user's touch. In this case, this type of sensed information represents a set of states whether the buttons are pressed or not at the moment. Meanwhile the semantic meaning of the interaction points is out of scope of this part, the other parts such as scenes or adaptation engines which request those interaction points may determine tha semantic meaning of the interaction points. This sensor type does not specify any sensing methods such as touch-pad, gesture-based sensor, motion sensor, and even intelligent vision intechnologies. The sensing properties of the sensor are specified as the multi interaction point sensor capability in ISO/IEC 23005-2.

6.20.2 Syntax

```
<!-- ################################################## -->
<!--  Definition of multi interaction point sensor type -->
<!-- ################################################## -->
<complexType name="MultiInteractionPointSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <sequence>
            <element name="InteractionPoint" type="siv:InteractionPointType"
maxOccurs="unbounded"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>
<complexType name="InteractionPointType">
   <attribute name="interactionPointId" type="ID" use="required"/>
   <attribute name="interactionPointStatus" type="boolean" default="false"/>
</complexType>
```

6.20.3 Binary representation syntax

MultiInteractionPointSensorType{	***Number of bits***	***Mnemonic***
SensedInfoBaseType		SensedInfoBaseType
NumOfInteractionPoint	16	uimsbf
for(k=0; k< NumOfInteractionPoint; k++) {		
InteractionPoint [k]		InteractionPointType
}		
}		
InteractionPointType {		
interactionPointId	See ISO/IEC 10646	UTF-8
interactionPointStatus	1	bslbf
}		

6.20.4 Semantics

Semantics of the `MultiInteractionPointSensorType`:

Name	*Definition*
`MultiInteractionPointSensorType`	Tool for describing sensed information captured by a multi interaction point sensor. The connection between each interaction point and its semantic on the adaptation engine is already known to both of them.

Name	*Definition*
	EXAMPLE Multi-button devices such as multi-touch pad, multi-finger detecting device, etc.
`SensedInfoBaseType`	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.
`InteractionPoint`	Describes the identification and the status of an interaction point which is included in a multi interaction point sensor.
`InteractionPointType`	Describes the referring identification of an interaction point and the status of an interaction point.
`interactionPointIdLength`	This field, which is only present in the binry representation, specifies the length of the `interactionPointIdLength` field encoded in binary representation in bytes.
`interactionPointId`	Describes the identifier of associated interaction point.
`interactionPointStatus`	Indicates the status of an interaction point which is included in a multi interaction point sensor. A value of "true" means that the interaction point receives user's input and "false" means that the interaction point does not.

Binary value (1 bits)	status of the interaction point
0	The interaction point does not receive user's input
1	The interaction point receives user's input

6.20.5 Examples

This example shows the description of a set of interaction points sensing with the following semantics. A multi-touch game, drum-kit, has a base drum, a tom-tom drum and a cymbal. To receive user's input, the game creates an interface via a multi interaction point sensor. The multi interaction point sensor of id "MIPSID001" includes three interaction points for a drum kit. At timestamp = "50000" where there are 1000 clock ticks per second, the region of the cymbal image is not pressed while the images of the base drum and the tom-tom drum are pressed.

```
<iidl:InteractionInfo xmlns:siv="urn:mpeg:mpeg-v:2017:01-SIV-NS"
xmlns:mpegvct="urn:mpeg:mpeg-v:2017:01-CT-NS" xmlns:iidl="urn:mpeg:mpeg-
v:2017:01-IIDL-NS" xsi:schemaLocation="urn:mpeg:mpeg-v:2017:01-SIV-NS
http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-SIV.xsd">
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:MultiInteractionPointSensorType"
id="MPS001" sensorIdRef="MPSID001" activate="true" >
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="1000"
pts="50000"/>
         <siv:InteractionPoint interactionPointId="IPT001"
interactionPointStatus="false"/>
         <siv:InteractionPoint interactionPointId="IPT002"
interactionPointStatus="true"/>
         <siv:InteractionPoint interactionPointId="IPT003"
interactionPointStatus="true"/>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

After 1 second, timestamp = “51000” the user is trying to release the drum kit. Therefore all interaction points are not pressed.

```
<iidl:InteractionInfo xmlns:siv=”urn:mpeg:mpeg-v:2017:01-SIV-NS”
xmlns:mpegvct=”urn:mpeg:mpeg-v:2017:01-CT-NS” xmlns:iidl=”urn:mpeg:mpeg-
v:2017:01-IIDL-NS” xmlns:xsi=”http://www.w3.org/2001/XMLSchema-instance”
xsi:schemaLocation=”urn:mpeg:mpeg-v:2017:01-SIV-NS
http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-SIV.xsd”>
    <iidl:SensedInfoList>
       <iidl:SensedInfo xsi:type=”siv:MultiInteractionPointSensorType”
id=”MPS001” sensorIdRef=”MPSID001” activate=”true” >
          <iidl:TimeStamp xsi:type=”mpegvct:ClockTickTimeType” timescale=”1000”
pts=”51000”/>
          <siv:InteractionPoint interactionPointId=”IPT001”
interactionPointStatus=”false”/>
          <siv:InteractionPoint interactionPointId=”IPT002”
interactionPointStatus=”false”/>
          <siv:InteractionPoint interactionPointId=”IPT003”
interactionPointStatus=”false”/>
       </iidl:SensedInfo>
    </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.21 Gaze tracking sensor type

6.21.1 General

This subclause specifies a sensor type which senses a position and a blinking status of user’s eye along with the orientation of user’s gaze. The gaze tracking sensor type does not specify any sensing methods such as infrared, vision, and inductive technologies. The sensing properties of the sensor are specified in a gaze tracking sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include user interactions, robotics, security systems, and others.

6.21.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of gaze tracking sensor type         -->
<!-- ################################################ -->
<complexType nae="GazeTrackingSensorType">
    <complexContent>
       <extension base="iidl:SensedInfoBaseType">
          <sequence>
             <element name="Gaze" type="siv:GazeType" maxOccurs="2"/>
          </sequence>
          <attribute name="personIdx" type="anyURI" use="optional"/>
       </extension>
    </complexContent>
</complexType>

<complexType name="GazeType">
    <sequence>
       <element name="Position" type="siv:PositionSensorType" minOccurs="0"/>
       <element name="Orientation" type="siv:OrientationSensorType"
minOccurs="0"/>
    </sequence>
    <attribute name="gazeIdx" type="anyURI" use="optional"/>
    <attribute name="blinkStatus" type="boolean" use="optional" default="false"/>
</complexType>
```

6.21.3 Binary representation syntax

GazeTrackingSensorType{	***Number of bits***	***Mnemonic***
SensedInfoBaseType		SensedInfoBaseType
personIdxRefFlag	1	bslbf
if(personIdxRefFlag) {		
personIxdRef	16	uimsbf
}		
NumOfGazes	16	uimsbf
for(k=0; k< NumOfGazes; k++) {		
Gaze [k]		GazeType
}		
}		
GazeType{		
PositionFlag	1	Bslbf
OrientationFlag	1	Bslbf
gazeIdxFlag	1	bslbf
blinkStatusFlag	1	bslbf
if(PositionFlag) {		
Position		PositionSensorType
}		
if(OrientatioinFlag) {		
Orientation		OrientationSensorType
}		
if(gazeIdxFlag) {		
gazeIdx	16	uimsbf
}		
if(blinkStatusFlag) {		
blinkStatus	1	uimsbf

}		
}		

6.21.4 Semantics

Semantics of the GazeTrackingSensorType:

Name	*Definition*
GazeTrackingSensorType	Tool for describing sensed information captured by one or more gaze tracking sensor. EXAMPLE Gaze tracking sensor, etc.
TimeStamp	Describes the time that the information is sensed.
personIdx	Describes an index of the person who is being sensed.
Gaze	Describes a set of gazes from a person.
GazeType	Describes the referring identification of a set of gazes.
Position	Describes the position information of an eye which is defined as PositionSensorType.
Orientation	Describes the direction of a gaze which is defined as OrientationSensorType.
gazeIdx	Describes an index of a gaze which is sensed from the same eye.
blinkStatus	Describes the eye's status in terms of blinking. "false" means the eye is not blinking and "true" means the eye is blinking. Default value of this attribute is "false".

6.21.5 Examples

This example shows the description of a gaze sensing with the following semantics. The gaze tracking sensor of id "GTSID001" was sensing two gazes from a person. According to the attributes, the sensor was tracking a person who is considered as the personIdx, "pSID001". One gaze was sensed at the position, (1.5, 0.5, -2.1) and the orientation of that gaze is (1.0, 1.0, 0.0) with no blink during a period. The other gaze was sensed at the position, (1.7, 0.5, -2.1) with same orientation with a blinking during a period. The sensor shall be sensed at timestamp="50000" where there are 1000 clock ticks per second.

```
<iidl:InteractionInfo xmlns:siv="urn:mpeg:mpeg-v:2017:01-SIV-NS"
xmlns:mpegvct="urn:mpeg:mpeg-v:2017:01-CT-NS" xmlns:iidl="urn:mpeg:mpeg-
v:2017:01-IIDL-NS" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:mpeg:mpeg-v:2017:01-SIV-NS
http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-SIV.xsd">
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:GazeTrackingSensorType" id="GTS001"
sensorIdRef="GTSID001" activate="true" personIdx="pSID001" >
```

```
          <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="1000"
pts="50000"/>
          <siv:Gaze gazeIdx="gz001" blinkStatus="false" >
             <siv:Position id="PS001" sensorIdRef="PSID001">
                <siv:Position>
                   <mpegvct:X>1.5</mpegvct:X>
                   <mpegvct:Y>0.5</mpegvct:Y>
                   <mpegvct:Z>-2.1</mpegvct:Z>
                </siv:Position>
             </siv:Position>
             <siv:Orientation id="OS001" sensorIdRef="OSID001">
                <siv:Orientation>
                   <mpegvct:X>1.0</mpegvct:X>
                   <mpegvct:Y>1.0</mpegvct:Y>
                   <mpegvct:Z>0.0</mpegvct:Z>
                </siv:Orientation>
             </siv:Orientation>
          </siv:Gaze>
          <siv:Gaze gazeIdx="gz002" blinkStatus="true" >
             <siv:Position id="PS002" sensorIdRef="PSID002">
                <siv:Position>
                   <mpegvct:X>1.7</mpegvct:X>
                   <mpegvct:Y>0.5</mpegvct:Y>
                   <mpegvct:Z>-2.1</mpegvct:Z>
                </siv:Position>
             </siv:Position>
             <siv:Orientation id="OS002" sensorIdRef="OSID002">
                <siv:Orientation>
                   <mpegvct:X>1.0</mpegvct:X>
                   <mpegvct:Y>1.0</mpegvct:Y>
                   <mpegvct:Z>0.0</mpegvct:Z>
                </siv:Orientation>
             </siv:Orientation>
          </siv:Gaze>
       </iidl:SensedInfo>
    </iidl:SensedInfoList>
</iidl:InteractionInfo>
```


6.22 Wind sensor type

6.22.1 General

This subclause specifies a sensor type also known as “anemometer”, which measures a velocity of wind at a certain position. The wind sensor type does not specify any sensing methods such as ultrasonic, laser-dopper, windmill, and inductive technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The applications of the sensor type may include weather forecast, media room control, and others.

6.22.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of wind sensor type                  -->
<!-- ################################################ -->
<complexType name="WindSensorType">
   <complexContent>
      <extension base="siv:VelocitySensorType"/>
   </complexContent>
</complexType>
```

6.22.3 Binary representation syntax

WindSensorType{	***Number of bits***	***Mnemonic***
Velocity		VelocityType
}		

6.22.4 Semantics

Semantics of the `WindSensorType`:

Name	*Definition*
`WindSensorType`	Tool for describing sensed information captured by none or more wind sensor. EXAMPLE Wind sensor, etc.
`Velocity`	Describes the speed and direction of a wind flow.

6.22.5 Examples

This example shows the description of a wind sensing with the following semantics. The wind sensor of id "WSID001" includes a velocity sensor. The wind vector was with a velocity, (1.0, 1.0, 0.0). The sensor shall be sensed at timestamp="50000" where there are 1000 clock ticks per second.

```
<iidl:SensedInfo xsi:type="siv:WindSensorType" id="WS001" sensorIdRef="WSID001"
activate="true" >
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="1000"
pts="50000"/>
   <siv:Velocity>
      <mpegvct:X>1.0</mpegvct:X>
      <mpegvct:Y>1.0</mpegvct:Y>
      <mpegvct:Z>0.0</mpegvct:Z>
   </siv:Velocity>
</iidl:SensedInfo>
```


6.23 Global position sensor type

6.23.1 General

This subclause specifies XML syntax, binary representation syntax, and semantics of the GlobalPositionSensorType with an example instantiation of the sensed information. This complex type is defined to specify the syntax for the interchange of information sensed by a global position sensor. Note that most of the global position sensors also have a capability of sensing the altitude, but a vocabulary for handling sensed altitude information is defined in a separate subclause.

6.23.2 Syntax

```
<!--############################################ -->
<!--Definition of global position sensor type  -->
<!--############################################ -->
<complexType name="GlobalPositionSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="crs" type="anyURI"
default="urn:ogc:def:crs:EPSG::4326"/>
         <attribute name="longitude" use="required">
            <simpleType>
               <restriction base="double">
                  <minInclusive value="-180.0"/>
                  <maxInclusive value="180.0"/>
               </restriction>
            </simpleType>
         </attribute>
         <attribute name="latitude" use="required">
            <simpleType>
               <restriction base="double">
                  <minInclusive value="-90.0"/>
                  <maxInclusive value="90.0"/>
               </restriction>
            </simpleType>
         </attribute>
      </extension>
   </complexContent>
</complexType>
```

6.23.3 Binary representation syntax

GlobalPositionSensorType{	***Number of bits***	***Mnemonic***
SensedInfoBaseType	See above	SensedInfoBaseType
crsflag	1	bslbf
if(crsflag){		
crs	See ISO/IEC 10646	UTF-8
}		
latitude	32	fsfb
longitude	32	fsfb
}		

6.23.4 Semantics

Semantics of the `GlobalPositionSensorType`:

Name	*Definition*
`GlobalPositionSensorType`	Tool for describing sensed information through global positioning system (gps) sensor with respect to a global position.
`TimeStamp`	Describes the time that the information is acquired mi(sensed).
`crsLength`	This field, which is only present in the binry representation, specifies the length of the `crsLength` field encoded in binary representation in bytes.
`crs`	Specifies the URI of the coordinate reference system based on which the values of longitude, latitude and altitude are given. The default is urn:ogc:def:crs:EPSG::4326 specifying the Coordinate Reference System (CRS) with code 4326 specified in the EPSG database available at http://www.epsg.org/.
`longitude`	Describes the position of the sensor in terms of degrees of longitude. Positive values represent eastern longitude and negative values represent western longitude. Ex: -132.236 represents 132.236 degrees West.
`Latitude`	Describes the position of the sensor in terms of degrees of latitude. Positive value represents northern latitude and negative value represents southern latitude. Ex: 37.103 represents 37.103 degrees North.
`crsflag`	This field, which is only present in the binary representation, signals the presence of the `crs` attribute. A value of “1” means the value of crs attribute shall be present and “0” means the value of `crs` attribute assumes the default value of “`urn:ogc:def:crs:EPSG::4326`” as specified in the semantics of `crs` without specifying it.

6.23.5 Examples

This example shows the description of a global position sensing with the following semantics. The latitude is 37.23 N and the longitude is 131.23 E. The position is sensed at system clock tick of 600 000 where there are 1000 ticks per second. The id of this sensed information is GPS001 and the id of the sensor is GPSID001.

```
<iidl:SensedInfo xsi:type="siv:GlobalPositionSensorType" id="GPS001"
sensorIdRef="GPSID001" activate="true" longitude="131.23" latitude="37.23">
<iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
</iidl:SensedInfo>
```

6.24 Altitude sensor type

6.24.1 General

This subclause specifies XML syntax, binary representation syntax, and semantics of the AltitudeSensorType with an example instantiation of the sensed information. This complex type is defined to specify the syntax for the interchange of information sensed by an altitude sensor. Note that no matter what kind of technology is

used to sense the altitude, the sensor type specified in this subclause only specifies the altitude above the geoid in metres.

6.24.2 Syntax

```
<!--##################################### -->
<!--Definition of altitude sensor type   -->
<!--##################################### -->
<complexType name="AltitudeSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="crs" type="anyURI"
default="urn:ogc:def:crs:EPSG::4326"/>
         <attribute name="altitude" type="double" use="required"/>
      </extension>
   </complexContent>
</complexType>
```

6.24.3 Binary representation syntax

AltitudeSensorType{	***Number of bits***	***Mnemonic***
SensedInfoBaseType	See above	SensedInfoBaseType
crsflag	1	bslbf
if(crsflag){		
crs	See ISO/IEC 10646	UTF-8
}		
Altitude	32	fsfb
}		

6.24.4 Semantics

Semantics of the `AltitudeSensorType`:

Name	*Definition*
`AltitudeSensorType`	Tool for describing sensed information through altimeter. The altitude defined in this type is following the WGS-84 coordinate reference system.
`crsLength`	This field, which is only present in the binary representation, specifies the length of the `crsLength` field encoded in binary representation in bytes.
`crs`	Specifies the URI of the coordinate reference system based on which the values of longitude, latitude and altitude are given. The default is urn:ogc:def:crs:EPSG::4326 specifying the Coordinate Reference System (CRS) with code 4326 specified in the EPSG

Name	*Definition*
	database available at http://www.epsg.org/.
`altitude`	Describes the altitude in the unit of metres above the geoid.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit (metre) is used, as a reference to a classification scheme term provided by `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1.
`crsflag`	This field, which is only present in the binary representation, signals the presence of the `crs` attribute. A value of "1" means the value of crs attribute shall be present and "0" means the value of `crs` attribute assumes the default value of "`urn:ogc:def:crs:EPSG::4326`" as specified in the semantics of `crs` without specifying it.

6.24.5 Examples

This example shows the description of an altitude sensing with the following semantics. The altitude is 123.21 metres above the geoid. The altitude is measured at the system clock tick of 600 000 where there are 1000 ticks per second. The id of this sensed information is AL001 and the id of the sensor is AltiID001.

```
<iidl:SensedInfo xsi:type="siv:AltitudeSensorType" id="AL001"
sensorIdRef="AltiID001" activate="true" altitude="123.21">
<iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
</iidl:SensedInfo>
```

6.25 Bend sensor type

6.25.1 General

This subclause specifies a bend sensor type which senses bending angles at each sensing point. The sensor type may sense a single angle or multiple angles of multiple axes depending upon the capability of the particular sensor. The bend sensor type does not specify any sensing methods such as resistive and fibre-optic technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The properties of the sensor are specified in the bend sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include robotics, exoskeleton controls, and others.

6.25.2 Syntax

```
<!--##################################### -->
<!--Definition of bend sensor type       -->
<!--##################################### -->
<complexType name="BendSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <sequence>
            <element name="ArrayBendValue" type="mpeg7:FloatMatrixType"
minOccurs="1" maxOccurs="unbounded"/>
         </sequence>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.25.3 Binary representation syntax

BendSensorType{	***Number of bits***	***Mnemonic***
unitFlag	1	bslbf
SensedInfoBaseType		SensedInfoBaseType
numOfChannels	16	uimsbf
numOfAxes	2	uimsbf
numOfLocations	16	uimsbf
for(i=0 ;i<numOfChannels ;i ++){		
for(j = 0; j< numOfAxes*numOfLocations; j++){		
ArrayBendValue[i, j]	32	fsbf
}		
}		
}		
If (unitFlag == 1){		
Unit	8	bslbf
}		
}		

6.25.4 Sematics

Semantics of the BendSensorType:

Name	*Definition*
BendSensorType	Tool for describing sensed information with respect to a bend sensor.
TimeStamp	Describes the time that the information is acquired (sensed).
unit	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by UnitTypeCS defined in ISO/IEC 23005-6:—, A.2.1. If the unit is not defined here, the default unit is degree.
ArrayBendValue	Describes the set of sensed values by the bend sensor with respect to the default unit or the unit defined in the unit attribute on each joint. NOTE 1 The ArrayBendValue is defined by the number of axes and the number of locations. Three by ten matrix indicates that the arrayValue has 10 sensing locations, each of which has 3-axis bend angles. The order of values in each row of the matrix could be started from the fingertip to the palm side. NOTE 2 In the binary representation, the number of the ArrayBendValue matrix is defined by the number of channels. Each matrix is defined by the number of axes and the number of locations similar to NOTE 1.
unitFlag	This field, which is only present in the binary representation, indicates the type of unit used in this sensed information.
numOfChannels	This field, which is only present in the binary representation, indicates the number of channels of the bend sensor
numOfAxes	This field, which is only present in the binary representation, indicates the dimension of the data at the sensing locations in each channel.
numOfLocations	This field, which is only present in the binary representation, indicates the number of sensing locations in each channel.

6.25.5 Examples

This example shows the description of a bend sensing with the following semantics. The bend sensor used for this sensed information has 2 channels, each of which has 2 sensing locations with 3 axes. The sensed values at the first channel are (0.0, 90.0, 0.0) and (10.0, 50.0, 40.0) for the two sensing locations. The sensed values at the second channel are (90.0, 0.0, 0.0) and (40.0, 10.0, 50.0) for the two sensing locations. The sensed information is measured at the system clock tick of 6000 where there are 100 ticks per second. The id of this sensed information is bs01 and the id of the sensor is BS_01.

```
<iidl:SensedInfo xsi:type="siv:BendSensorType" id="bs01" sensorIdRef="BS_01">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
   <siv:ArrayBendValue mpeg7:dim="3 3">
      0.0  90.0   0.0
      10.0 50.0 40.0
   </siv:ArrayBendValue>
   <siv:ArrayBendValue mpeg7:dim="3 3">
      90.0   0.0  0.0
      40.0 10.0 50.0
   </siv:ArrayBendValue>
</iidl:SensedInfo>
```

6.26 Gas sensor type

6.26.1 General

This subclause specifies a gas sensor type which senses a gas type and its gas concentration value. The sensor type may sense a single gas type or multiple types of gas depending upon the capability of the particular sensor. The gas sensor type does not specify any sensing methods such as chemical and bio-chemical technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The properties of the sensor are specified in the gas sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include home securities, environmental monitoring and others.

6.26.2 Syntax

```
<!--################################### -->
<!--Definition of gas sensor type      -->
<!--################################### -->
<complexType name="GasSensorType">
  <complexContent>
    <extension base="iidl:SensedInfoBaseType">
      <sequence>
        <element name="GasType" type="mpeg7:termReferenceType" minOccurs="0"/>
      </sequence>
      <attribute name="value" type="float" use="optional"/>
      <attribute name="unit" type="mpegvct:unitType" use="optional"/>
    </extension>
  </complexContent>
</complexType>
```


6.26.3 Binary representation syntax

GasSensorType{	***Number of bits***	***Mnemonic***
gasTypeFlag	1	bslbf
valueFlag	1	bslbf
unitFlag	1	bslbf
SensedInfoBaseType	See above	SensedInfoBaseType

If (gasTypeFlag == 1){		
GasTypeSelect	16	uimsbf
}		
If (valueFlag == 1){		
value	32	fsfb
}		
If (unitFlag == 1){		
unit	8	bslbf
}		
}		

6.26.4 Semantics

Semantics of the `GasSensorType`:

Name	*Definition*
`GasSensorType`	Tool for describing sensed information with respect to a gas sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`GasType`	Describes the sensed type by the gas sensor. Tool for describing a gas type as a reference to a classification scheme term provided by `GasTypeCS` defined in ISO/IEC 23005-6:—, A.2.16. The details of the structure and use of classification scheme and `termReferencetype` description is defined in ISO/IEC 15938-5. EXAMPLE `urn:mpeg:mpeg-v:01-CI-GasCS-NS:oxygen` would describe the unit for gas concentrations in ppm (parts per million).
`value`	Describes the sensed gas concentration value by the gas sensor with respect to the default unit or the unit defined in the unit attribute.
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The default unit for the GasSensorType is ppm. EXAMPLE `urn:mpeg:mpeg-v:01-CI-UnitTypeCS-NS:ppm` would describe the unit for gas concentrations in ppm (parts per million). `urn:mpeg:mpeg-v:01-CI-UnitTypeCS-NS:pcpl` would describe the unit for gas concentrations in pCi/l (picocuries per litre).

Name	*Definition*
gasTypeFlag	This field, which is only present in the binary representation, signals the presence of GasTypeSelect. A value of “1” means the GasTypeSelect shall be used and “0” means that the GasTypeSelect shall not be used.
valueFlag	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of “1” means that the attribute shall be used and “0” means that the attribute shall not be used.
unitFlag	This field, which is only present in the binary representation, indicates the type of unit used in this sensed information.
GasTypeSelect	This field, which is only present in the binary representation, signals what value type is used. The binary reprepentation of GasTypeSelect is give in ISO/IEC 23005-6:—, A.2.16, as GasTypeCS.

6.26.5 Examples

This example shows the description of a gas sensing with the following semantics. The description has identifier of “gas01” and the sensor references an actual sensor with ID of “GSID_01”. The sensor shall be activated and the value shall be 100 with the unit of ppm. The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:SensedInfo xsi:type="siv:GasSensorType" id="gas01" activate="true"
sensorIdRef="GSID_01" value="100" unit="urn:mpeg:mpeg-v:01-CI-UnitTypeCS-NS:ppm">
  <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
  <siv:GasType>urn:mpeg:mpeg-v:01-CI-GasCS-NS:oxygen</siv:GasType>
</iidl:SensedInfo>
```

6.27 Dust sensor type

6.27.1 General

This subclause specifies a gas sensor type which senses dust concentration value without identifying the types of dust. The dust sensor type does not specify any sensing methods such as optical and tribo-electric technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The properties of the sensor are specified in the dust sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include home securities, environmental monitoring and others.

6.27.2 Syntax

```
<!--##################################### -->
<!--Definition of dust sensor type       -->
<!--##################################### -->
<complexType name="DustSensorType">
  <complexContent>
    <extension base="iidl:SensedInfoBaseType">
      <attribute name="value" type="float" use="optional"/>
      <attribute name="unit" type="mpegvct:unitType" use="optional"/>
    </extension>
  </complexContent>
</complexType>
```

6.27.3 Binary representation syntax

DustSensorType{	***Number of bits***	***Mnemonic***
valueFlag	1	bslbf
unitFlag	1	bslbf
SensedInfoBaseType	See above	SensedInfoBaseType
If (valueFlag == 1){		
value	32	fsfb
}		
If (unitFlag == 1){		
unit	8	bslbf
}		
}		

6.27.4 Semantics

Semantics of the `DustSensorType`:

Name	*Definition*
`DustSensorType`	Tool for describing sensed information with respect to a dust sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`value`	Describes the sensed dust concentration value by the dust check with respect to the default unit or the unit defined in the unit attribute.
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The default unit is $\mu g/m^3$.
`valueFlag`	This field, which is only present in the binary representation, signals the presence of sensor value attribute. A value of ”1” means that the attribute shall be used and ”0” means that the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals if a unit other than default unit is used. A value of ”1” indicates that the unit specified in the unit attribute shall be used and ”0” indicates that the default unit shall be used.

6.27.5 Examples

This example shows the description of a dust sensing with the following semantics. The description has identifier of “dust01” and the sensor references an actual sensor with ID of “DTID_01”. The sensor shall be

activated and the value shall be 100 with the unit of $\mu g/m^3$. The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:SensedInfo xsi:type="siv:DustSensorType" id="dust01" activate="true"
sensorIdRef="DTID_01" value="100" unit="urn:mpeg:mpeg-v:01-CI-UnitTypeCS-
NS:microgpcm">
  <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
</iidl:SensedInfo>
```

6.28 Body height sensor type

6.28.1 General

This subclause specifies a sensor type which senses body height. The body height sensor type does not specify any sensing methods such as ultrasonic, optical, and inductive technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The applications of the sensor type may include physical interactive game, health monitoring, and others.

6.28.2 Syntax

```
<!--########################################## -->
<!--Definition of body height sensor type  -->
<!--########################################## -->
<complexType name="BodyHeightSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="value" type="float" use="required"/>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.28.3 Binary representation syntax

BodyHeightSensorType{	***Number of bits***	***Mnemonic***
unitFlag	1	bslbf
SensedInfoBaseType	See above	SensedInfoBaseType
value	32	fsfb
If (unitFlag == 1){		
unit	8	bslbf
}		
}		

6.28.4 Semantics

Semantics of the `BodyHeightSensorType`:

Name	*Definition*
`BodyHeightSensorType`	Tool for describing sensed information with respect to a body height sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by UnitTypeCS defined in A ISO/IEC 23005-2:2018, A.2.1.
`value`	Describes the sensed value of the body height with respect to the centimetre (cm) scale.
`unitFlag`	This field, which is only present in the binary representation, signals if a unit other than default unit is used. A value of "1" indicates that the unit specified in the unit attribute shall be used and "0" indicates that the default unit shall be used.

6.28.5 Examples

This example shows the description of a body height sensing with the following semantics. The sensor has an ID of "BHS001" and references "BHSID001". The sensor shall be activated and the value shall be 170.5 (cm). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:SensedInfo xsi:type="siv:BodyHeightSensorType" id="BHS001"
sensorIdRef="BHSID001" activate="true" value="170.5">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
</iidl:SensedInfo>
```

6.29 Body weight sensor type

6.29.1 General

This subclause specifies a sensor type which senses body weight. The body weight sensor type does not specify any sensing methods such as strain gauge and gravity technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The applications of the sensor type may include physical interactive game, health monitoring, and others.

6.29.2 Syntax

```
<!--######################################## -->
<!--Definition of body weight sensor type  -->
<!--######################################## -->
<complexType name="BodyWeightSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="value" type="float" use="required"/>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.29.3 Binary representation syntax

BodyWeightSensorType{	***Number of bits***	***Mnemonic***
unitFlag	1	bslbf
SensedInfoBaseType	See above	SensedInfoBaseType
value	32	fsfb
If (unitFlag == 1){		
unit	8	bslbf
}		
}		

6.29.4 Semantics

Semantics of the `BodyWeightSensorType`:

Name	*Definition*
`BodyWeightSensorType`	Tool for describing sensed information with respect to a body weight sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by UnitTypeCS defined in ISO/IEC 23005-2:2018, A.2.1.
`value`	Describes the sensed value of the body weight with respect to the kilogram (kg) scale.
`unitFlag`	This field, which is only present in the binary representation, signals if a unit other than default unit is used. A value of “1” indicates that the unit specified in the unit attribute shall be used and “0” indicates that the default unit shall be used.

6.29.5 Examples

This example shows the description of a body weight sensing with the following semantics. The sensor has an ID of “BWS001” and references “BWSID001”. The sensor shall be activated and the value shall be 65.4 (kg). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:SensedInfo xsi:type="siv:BodyWeightSensorType" id="BWS001"
sensorIdRef="BWSID001" activate="true" value="65.4">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
</iidl:SensedInfo>
```

6.30 Body temperature sensor type

6.30.1 General

This Subclause specifies a sensor type which senses body temperature. The body temperature sensor type does not specify any sensing methods such as the thermally sensitive resistor technology. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The applications of the sensor type may include physical interactive game, health monitoring, and others.

6.30.2 Syntax

```
<!--################################################ -->
<!--Definition of body temperature sensor type  -->
<!--################################################ -->
<complexType name="BodyTemperatureSensorType">
   <complexContent>
      <extension base="siv:TemperatureSensorType">
         <attribute name="location" type="nonNegativeInteger" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.30.3 Binary representation syntax

BodyTemperatureSensorType{	***Number of bits***	***Mnemonic***
locationFlag	1	bslbf
TemperatureSensorType	See above	TemperatureSensorType
if (locationFlag == 1){		
location	4	uimsbf
}		
}		

6.30.4 Semantics

Semantics of the `BodyTemperatureSensorType`:

Name	*Definition*
`BodyTemperatureSensorType`	Tool for describing sensed information with respect to a body temperature sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`value`	Describes the sensed value of the body weight with respect to the Celsius (°C) scale.

<table>
<tr><th>Name</th><th>Definition</th></tr>
<tr><td>location</td><td>Describes the position information where the sensor is sensed. The default value of location is 1.

1. General body temperature 2. Axillary (armpit) 3. Ear (usually earlobe) 4. Finger 5. Gastro-intestinal tract 6. Mouth 7. Rectum 8. Toe 9. Tympanum (ear drum).

The following table shall be used for binary representation.

<table>
<tr><th>Binary representation (4 bits)</th><th>Position information</th></tr>
<tr><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>General body temperature</td></tr>
<tr><td>2</td><td>Axillary (armpit)</td></tr>
<tr><td>3</td><td>Ear (usually earlobe)</td></tr>
<tr><td>4</td><td>Finger</td></tr>
<tr><td>5</td><td>Gastro-intestinal tract</td></tr>
<tr><td>6</td><td>Mouth</td></tr>
<tr><td>7</td><td>Rectum</td></tr>
<tr><td>8</td><td>Toe</td></tr>
<tr><td>9</td><td>Tympanum (ear drum)</td></tr>
<tr><td>10-15</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>locationFlag</td><td>This field, which is only present in the binary representation, signals if the body location type is used. A value of “1” indicates that the type shall be used and “0” indicates that the default location shall be used.</td></tr>
</table>

6.30.5 Examples

This example shows the description of a body temperature sensing with the following semantics. The sensor has an ID of “BTS001” and references “BTSID001”. The sensor shall be activated and the value shall be 36.5 (°C). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second and be located in the mouth.

```
<iidl:SensedInfo xsi:type="siv:BodyTemperatureSensorType" id="BTS001"
sensorIdRef="BTSID001" activate="true" value="36.5" location="6">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
</iidl:SensedInfo>
```

6.31 Body fat sensor type

6.31.1 General

This subclause specifies a sensor type which senses body fat. The body fat sensor type does not specify any sensing methods such as chemical and bio-chemical technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The applications of the sensor type may include physical interactive game, health monitoring, and others.

6.31.2 Syntax

```
<!--################################### -->
<!--Definition of body fat sensor type   -->
<!--################################### -->
<complexType name="BodyFatSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="value" type="float" use="required"/>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.31.3 Binary representation syntax

BodyFatSensorType{	***Number of bits***	***Mnemonic***
unitFlag	1	bslbf
SensedInfoBaseType	See above	SensedInfoBaseType
value	32	fsfb
If (unitFlag == 1){		
unit	8	bslbf
}		
}		

6.31.4 Semantics

Semantics of the `BodyFatSensorType`:

Name	*Definition*
`BodyFatSensorType`	Tool for describing sensed information with respect to a body fat sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided

Name	*Definition*
	by UnitTypeCS defined in ISO/IEC 23005-2:2018, A.2.1.
`value`	Describes the sensed value of the body fat with respect to the percentage (%).
`unitFlag`	This field, which is only present in the binary representation, signals if a unit other than default unit is used. A value of “1” indicates that the unit specified in the unit attribute shall be used and “0” indicates that the default unit shall be used.

6.31.5 Examples

This example shows the description of a body fat sensing with the following semantics. The sensor has an ID of “BFS001” and references “BFSID001”. The sensor shall be activated and the value shall be 75 (%). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:SensedInfo xsi:type="siv:BodyFatSensorType" id="BFS001"
sensorIdRef="BFSID001" activate="true" value="75">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
</iidl:SensedInfo>
```

6.32 Blood type sensor type

6.32.1 General

This subclause specifies a sensor type which senses blood type. The blood type sensor type does not specify any sensing methods such as chemical and bio-chemical technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The applications of the sensor type may include physical interactive game, health monitoring, and others.

6.32.2 Syntax

```
<!--##################################### -->
<!--Definition of Blood type sensor type -->
<!--##################################### -->
<complexType name="BloodTypeSensorType">
  <complexContent>
    <extension base="iidl:SensedInfoBaseType">
      <sequence>
        <element name="ABOType">
          <simpleType>
            <restriction base="string">
              <enumeration value="A"/>
              <enumeration value="B"/>
              <enumeration value="AB"/>
              <enumeration value="O"/>
            </restriction>
          </simpleType>
        </element>
        <element name="RhType">
          <simpleType>
```

```
          <restriction base="string">
            <enumeration value="+"/>
            <enumeration value="-"/>
          </restriction>
        </simpleType>
      </element>
    </sequence>
  </extension>
 </complexContent>
</complexType>
```

6.32.3 Binary representation syntax

BloodTypeSensorType	***Number of bits***	***Mnemonic***
SensedInfoBaseType	See above	SensedInfoBaseType
ABOType	2	bslbf
RhType	1	bslbf
}		

6.32.4 Semantics

Semantics of the `BloodTypeSensorType`:

Name	*Definition*
`BloodTypeSensorType`	Tool for describing sensed information with respect to a blood type sensor.
`ABOType`	Describes the sensed value of the ABO blood types: A, B, AB, and O. The following table shall be used for binary representation.
`RhType`	Describes the sensed value of the Rh blood types: Rh positive (+) and Rh negative (-). The following table shall be used for binary representation.

Binary representation (2 bits)	ABO type
00	A
01	B
10	AB
11	O

Name	*Definition*	
	Binary representation (1 bit)	Rh type
	0	Rh positive (+)
	1	Rh negative (-)

6.32.5 Examples

This example shows the description of a blood type sensing with the following semantics. The sensor has an ID of “BTYS001” and references “BTYSID001”. The sensor shall be activated. The ABO blood type shall be A and the Rh blood type shall be Rh + (Rh positive). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:SensedInfo xsi:type="siv:BloodTypeSensorType" id="BTYS001"
sensorIdRef="BTYSID001" activate="true">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
   <siv:ABOType>A</siv:ABOType>
   <siv:RhType>+</siv:RhType>
</iidl:SensedInfo>
```

6.33 Blood pressure sensor type

6.33.1 General

This subclause specifies a sensor type which senses blood pressure. The blood pressure sensor type does not specify any sensing methods such as a sphygmomanometer technology. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The applications of the sensor type may include physical interactive game, health monitoring, and others.

6.33.2 Syntax

```
<!--########################################### -->
<!--Definition of blood pressure sensor type  -->
<!--########################################### -->
<complexType name="BloodPressureSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="systolicBP" type="float" use="optional"/>
         <attribute name="diastolicBP" type="float" use="optional"/>
         <attribute name="MAP" type="float" use="optional"/>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.33.3 Binary representation syntax

BloodPressureSensorType{	*Number of bits*	*Mnemonic*
unitFlag	1	Bslbf
systolicBPFlag	1	Bslbf
diastolicBPFlag	1	Bslbf
MAPFlag	1	Bslbf
SensedInfoBaseType	See above	SensedInfoBaseType
if (systolicBPFlag == 1) {		
systolicBP	32	Fsfb
}		
if (diastolicBPFlag == 1) {		
diastolicBP	32	Fsfb
}		
if (MAPFlag == 1) {		
MAP	32	Fsfb
}		
if (unitFlag == 1){		
unit	8	bslbf
}		
}		

6.33.4 Semantics

Semantics of the `BloodPressureSensorType`:

Name	*Definition*
`BloodPressureSensorType`	Tool for describing sensed information with respect to a blood pressure sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by UnitTypeCS defined in ISO/IEC 23005-2:2018, A.2.1.

Name	*Definition*
systolicBP	Describes the sensed value of the systolic blood pressure with respect to the millimetres of mercury (mmHg).
diastolicBP	Describes the sensed value of the diastolic blood pressure with respect to the millimetres of mercury (mmHg).
MAP	Describes the sensed value of the mean arterial pressure with respect to the millimetres of mercury (mmHg).
unitFlag	This field, which is only present in the binary representation, signals if a unit other than default unit is used. A value of “1” indicates that the unit specified in the unit attribute shall be used and “0” indicates that the default unit shall be used.
systolicBPFlag	This field, which is only present in the binary representation, signals if the value of systolicBP is present. A value of “1” indicates that the value of systolicBP shall be present and “0” indicates that systolicBP shall not be present.
diastolicBPFlag	This field, which is only present in the binary representation, signals if the value of diastolicBP is present. A value of “1” indicates that the value of diastolicBP shall be present and “0” indicates that diastolicBP shall not be present.
MAPFlag	This field, which is only present in the binary representation, signals if the value of MAP is present. A value of “1” indicates that the value of MAP shall be present and “0” indicates that MAP shall not be present.

6.33.5 Examples

This example shows the description of a blood pressure sensing with the following semantics. The sensor has an ID of “BPS001” and references “BPSID001”. The sensor shall be activated. The systolic blood pressure shall be 121(mmHg), the diastolic blood pressure shall be 83(mmHg) and the mean arterial pressure shall be 100(mmHg). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:SensedInfo xsi:type="siv:BloodPressureSensorType" id="BPS001"
sensorIdRef="BPSID001" activate="true" systolicBP="121" diastolicBP="83"
MAP="100">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
</iidl:SensedInfo>
```

6.34 Blood sugar sensor type

6.34.1 General

This subclause specifies a sensor type which senses blood sugar. The blood sugar sensor type does not specify any sensing methods such as chemical and bio-chemical technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The applications of the sensor type may include physical interactive game, health monitoring, and others.

6.34.2 Syntax

```
<!--########################################## -->
<!--Definition of blood sugar sensor type  -->
<!--########################################## -->
<complexType name="BloodSugarSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="value" type="float" use="required"/>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.34.3 Binary representation syntax

BloodSugarSensorType{	***Number of bits***	***Mnemonic***
unitFlag	1	bslbf
SensedInfoBaseType	See above	SensedInfoBaseType
value	32	fsfb
If (unitFlag == 1){		
unit	8	bslbf
}		
}		

6.34.4 Semantics

Semantics of the `BloodSugarSensorType`:

Name	*Definition*
`BloodSugarSensorType`	Tool for describing sensed information with respect to a blood sugar sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by UnitTypeCS defined in ISO/IEC 23005-2:2018, A.2.1.
`value`	Describes the sensed value of the blood sugar with respect to the milligrams per deciliter (mg/dL).
`unitFlag`	This field, which is only present in the binary representation, signals if a unit other than default unit is used. A value of “1” indicates that the unit specified in the unit attribute shall be used and “0” indicates

Name	*Definition*
	that the default unit shall be used.

6.34.5 Examples

This example shows the description of a blood sugar sensing with the following semantics. The sensor has an ID of “BSS001” and references “BSSID001”. The sensor shall be activated and the value shall be 115 (mg/dL). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:SensedInfo xsi:type="siv:BloodSugarSensorType" id="BSS001"
sensorIdRef="BSSID001" activate="true" value="115">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
</iidl:SensedInfo>
```

6.35 Blood oxygen sensor type

6.35.1 General

This subclause specifies a sensor type which senses blood oxygen. The blood oxygen sensor type does not specify any sensing methods such as chemical and bio-chemical technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The applications of the sensor type may include physical interactive game, health monitoring, and others.

6.35.2 Syntax

```
<!--########################################## -->
<!--Definition of blood oxygen sensor type  -->
<!--########################################## -->
<complexType name="BloodOxygenSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="value" type="float" use="required"/>
         <attribute  name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```


6.35.3 Binary representation syntax

BloodOxygenSensorType{	***Number of bits***	***Mnemonic***
unitFlag	1	bslbf
SensedInfoBaseType	See above	SensedInfoBaseType
value	32	fsfb
If (unitFlag == 1){		

unit	8	bslbf
}		
}		

6.35.4 Semantics

Semantics of the `BloodOxygenSensorType`:

Name	*Definition*
`BloodOxygenSensorType`	Tool for describing sensed information with respect to a blood oxygen sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by UnitTypeCS defined in ISO/IEC 23005-2:2018, A.2.1.
`value`	Describes the sensed value of the blood oxygen saturation with respect to the percentage (%).
`unitFlag`	This field, which is only present in the binary representation, signals if a unit other than default unit is used. A value of “1” indicates that the unit specified in the unit attribute shall be used and “0” indicates that the default unit shall be used.

6.35.5 Examples

This example shows the description of a blood oxygen sensing with the following semantics. The sensor has an ID of “BOS001” and references “BOSID001”. The sensor shall be activated and the value shall be 96.0 (%). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:SensedInfo xsi:type="siv:BloodOxygenSensorType" id="BOS001"
sensorIdRef="BOSID001" activate="true" value="96.0">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
</iidl:SensedInfo>
```

6.36 Heart rate sensor type

6.36.1 General

This subclause specifies a sensor type which senses heart rate. The heart rate sensor type does not specify any sensing methods such as a bio-mechanical technology. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The applications of the sensor type may include physical interactive game, health monitoring, and others.

6.36.2 Syntax

```
<!--##################################### -->
<!--Definition of heart rate sensor type -->
<!--##################################### -->
<complexType name="HeartRateSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="value" type="float" use="required"/>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.36.3 Binary representation syntax

HearRateSensorType{	***Number of bits***	***Mnemonic***
unitFlag	1	bslbf
SensedInfoBaseType	See above	SensedInfoBaseType
value	32	fsfb
If (unitFlag == 1){		
unit	8	bslbf
}		
}		

6.36.4 Semantics

Semantics of the `HeartRateSensorType`:

Name	*Definition*
`HeartRateSensorType`	Tool for describing sensed information with respect to a heart rate sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by UnitTypeCS defined in ISO/IEC 23005-2:2018, A.2.1.
`value`	Describes the sensed value of the heart rate with respect to the beats per minute (BPM).
`unitFlag`	This field, which is only present in the binary representation, signals if a unit other than default unit is used. A value of “1” indicates that the unit specified in the unit attribute shall be used and “0” indicates

Name	*Definition*
	that the default unit shall be used.

6.36.5 Examples

This example shows the description of a heart rate sensing with the following semantics. The sensor has an ID of “HRS001” and references “HRSID001”. The sensor shall be activated and the value shall be 65 (BPM). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:SensedInfo xsi:type="siv:HeartRateSensorType" id="HRS001"
sensorIdRef="HRSID001" activate="true" value="65">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
</iidl:SensedInfo>
```

6.37 Electrograph sensor type

6.37.1 General

This subclause specifies an electrograph sensor type which produces any electrical graphs. The electrograph sensor type is a base type of electrograph-related bio-signals such as electroencephalogram (EEG), electrocardiogram (ECG), electromyogram (EMG), and electro-oculogram (EOG). The properties of the sensor are specified in the electrograph sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include medical use, health monitoring, brain computer interface and others.

6.37.2 Syntax

```
<!--##################################### -->
<!--Definition of electrograph sensor type -->
<!--##################################### -->
<complexType name="ElectrographSensorType">
   <complexContent>
   <extension base="iidl:SensedInfoBaseType">
      <sequence>
         <element name="waveValue" type="mpeg7:FloatMatrixType" minOccurs="0"/>
      </sequence>
      <attribute name="unit" type="mpegvct:unitType"
 use="optional"/>
      <attribute name="waveformLabel" type="mpeg7:termReferenceType"
use="optional"/>
      <attribute name="maxAmplitude" type="float" use="optional"/>
   </extension>
 </complexContent>
</complexType>
```

6.37.3 Binary representation syntax

ElectrographSensorType{	***Number of bits***	***Mnemonic***
waveValueFlag	1	bslbf
unitFlag	1	bslbf

waveformLabelFlag	1	bslbf
maxAmplitudeFlag	1	bslbf
SensedInfoBaseType	See above	SensedInfoBaseType
numOfChannels	16	uimsbf
numOfSamples	16	uimsbf
If (unitFlag == 1){		
unit	8	bslbf
}		
If (waveformLabelFlag == 1){		
waveformLabel	8	bslbf
}		
If (maxAmplitudeFlag == 1){		
maxAmplitude	32	fsbf
}		
If (waveValueFlag == 1){		
for(k = 0; k< numOfSamples; k++){		
for(j=0;j< numOfChannels;j++){		
WaveValue[(k * numOfChannels + j)]	32	fsbf
}		
}		
}		
}		

6.37.4 Semantics

Semantics of the ElectrographSensorType:

Name	*Definition*
`ElectrographSensorType`	Tool for describing sensed information with respect to an electrograph sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).

Name	*Definition*
`WaveValue`	Describes the time series sensed value of the electrograph sensor with respect to the microvolt (μV). The `dim` attribute of the matrix specifies in the order of the number of samples per channel and the number of channels. (i.e., the first dimension is number samples, and the second dimension is the number of channels).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by UnitTypeCS defined in ISO/IEC 23005-6:—, A.2.1.
`waveformLabel`	Describes the label of the waveform based as a reference to a classification scheme term provided by waveformLabelCS of EEG, ECG, EMG and EOG defined in B.1, B.2, B.3, and B.4, respectively.
`maxAmplitude`	Describes the maximum amplitude among the acquired time series value of the electrode sensor with respect to the microvolt (μV).
`waveValueFlag`	This field, which is only present in the binary representation, signals the presence of WaveValue. A value of “1” means the WaveValue shall be used and “0” means that the WaveValue shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals if a unit other than default unit is used. A value of “1” indicates that the unit specified in the unit attribute shall be used and “0” indicates that the default unit shall be used.
`waveformLabelFlag`	This field, which is only present in the binary representation, signals if the label of waveform is present. A value of “1” indicates that the label of waveform shall be present and “0” indicates that the waveform label shall not be present.
`maxAmplitudeFlag`	This field, which is only present in the binary representation, signals if the value of maximum amplitude is present. A value of “1” indicates that the value of maximum amplitude shall be present and “0” indicates that the value of maximum amplitude shall not be present.
`numOfChannels`	This field, which is only present in the binary representation, represents the number of channels.
`numOfSamples`	This field, which is only present in the binary representation, represents the number of samples in the sensed information for each channel.

6.37.5 Examples

This example shows the description of an electrograph sensing with the following semantics. The segmented sensed electrograph signal stream is composed of ten values of “0.5, 1.1, 2.2, 1.7, 1.1, 2.4, 5.7, 1.3, 0.5, 1.1”. The unit of the signal is millivolt. The electrograph signal stream is obtained at the system clock tick of 60000 where there are 100 ticks per second. The id of this sensed information is egs01 and the id of the sensor is EGSID_01.

```
<iidl:SensedInfo xsi:type="siv:ElectrographSensorType" id="egs01"
sensorIdRef="EGS_01">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
   <siv:WaveValue mpeg7:dim="10 1">
      0.5 1.1 2.2 1.7 1.1 2.4 5.7 1.3 0.5 1.1
   </siv:WaveValue>
</iidl:SensedInfo>
```

6.38 EEG sensor type

6.38.1 General

This subclause specifies an electroencephalogram sensor type which detects a set of brain waves among the electrodes attached to the scalp that act as transducers. The applications of the sensor type may include medical use, health monitoring, brain computer interface and others.

6.38.2 Syntax

```
<!--##################################### -->
<!--Definition of EEG sensor type        -->
<!--##################################### -->
<complexType name="EEGSensorType">
   <complexContent>
   <extension base="siv:ElectrographSensorType">
   <attribute name="wavePattern" type="mpeg7:termReferenceType" use="optional"/>
   </extension>
   </complexContent>
</complexType>
```

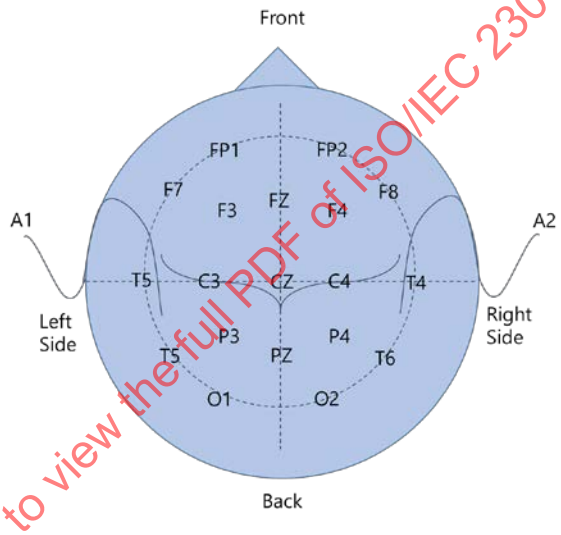
6.38.3 Binary representation syntax

EEGSensorType{	***Number of bits***	***Mnemonic***
wavePatternFlag	1	bslbf
electrographSensorType	See above	electrographSensorType
if (wavePatternFlag == 1){		
wavePattern	4	bslbf
}		
}		

6.38.4 Semantics

Semantics of the `EEGSensorType`:

Name	*Definition*
`EEGSensorType`	Tool for describing sensed information with respect to an electroencephalogram (EEG) sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`WaveValue`	Describes the time series sensed value of the EEG sensor with respect to the microvolt (μV).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by UnitTypeCS defined in ISO/IEC 23005-6:—, A.2.1.
`waveformLabel`	Describes the label of the waveform based as a reference to a classification scheme term provided by EEG_waveformLabelCS defined in B.1.



< Electrode locations >

FP1.2	Frontal pole
F3.4	Frontal
C3.4	Central
P3.4	Pariental
O1.2	Occipital
F7.8	Anterior temporal
T3.4	Middle temporal
T5.6	Posterior temporal
FZ	Midline-frontal
CZ	Midline-central
PZ	Midline-pariental
A1.2	Auricular

< Symbols and their corresponding meaning >

[10-20 electrode EEG system]

Name	*Definition*

WaveformLabel	Description
EEG_FP1_F7	Describes the waveform between FP1 and F7
EEG_F7_T3	Describes the waveform between F7 and T3
EEG_T3_T5	Describes the waveform between T3 and T5
EEG_T5_O1	Describes the waveform between T5 and O1
EEG_FP2_F8	Describes the waveform between FP2 and F8
EEG_F8_T4	Describes the waveform between F8 and T4
EEG_T4_T6	Describes the waveform between T4 and T6
EEG_T6_O2	Describes the waveform between T6 and O2
EEG_FP1_F3	Describes the waveform between FP1 and F3
EEG_F3_C3	Describes the waveform between F3 and C3
EEG_C3_P3	Describes the waveform between C3 and P3
EEG_P3_O1	Describes the waveform between P3 and O1
EEG_FP2_F4	Describes the waveform between FP2 and F4
EEG_F4_C4	Describes the waveform between F4 and C4
EEG_C4_P4	Describes the waveform between C4 and P4
EEG_P4_O2	Describes the waveform between P4 and O2
EEG_FZ_CZ	Describes the waveform between FZ and CZ
EEG_CZ_PZ	Describes the waveform between CZ and PZ

[EEG waveform label]

The following table shall be used for binary representation.

Binary representation (5 bits)	**EEG waveform label**
0	EEG_FP1_F7
1	EEG_F7_T3
2	EEG_T3_T5

Name	Definition
	3 EEG_T5_O1 4 EEG_FP2_F8 5 EEG_F8_T4 6 EEG_T4_T6 7 EEG_T6_O2 8 EEG_FP1_F3 9 EEG_F3_C3 10 EEG_C3_P3 11 EEG_P3_O1 12 EEG_FP2_F4 13 EEG_F4_C4 14 EEG_C4_P4 15 EEG_P4_O2 16 EEG_FZ_CZ 17 EEG_CZ_PZ 18-31 reserved
`maxAmplitude`	Describes the maximum amplitude among the acquired time series value of the EEG sensor with respect to the microvolt (μV).
`wavePattern`	Specifies the pattern of the sensed wave as a reference to a classification scheme term provided by WavePatternCS defined in B.5. The following table shall be used for binary representation. Binary representation (4 bits) / Wave pattern type 0 reserved 1 EEG Delta 2 EEG Theta 3 EEG Alpha 4 EEG Beta 5 EEG Gamma 6-15 reserved

Name	*Definition*
wavePatternFlag	This field, which is only present in the binary representation, signals if wavePattern attribute is specified or not. A value of ”1” indicates that the attribute is used and ”0” shall indicate that the attribute is not used.

6.38.5 Examples

This example shows the description of an EEG sensing with the following semantics. The segmented sensed EEG signal wave, measured between FP1 and F7, is composed of ten values of “0.5, 1.1, 2.2, 1.7, 1.1, 2.4, 5.7, 1.3, 0.5, 1.1”. The unit of the signal is millivolt. The EEG signal stream is obtained at the system clock tick of 60000 where there are 100 ticks per second. The id of this sensed information is eeg01 and the id of the sensor is EEGID_01.

```
<iidl:SensedInfo xsi:type="siv:EEGSensorType" id="eeg01" activate="true"
sensorIdRef="EEGID_01" unit="urn:mpeg:mpeg-v:01-CI-UnitTypeCS-NS:millivolt"
waveformLabel="urn:mpeg:mpeg-v:01-CI-EEG_WaveformLabelCS-NS:EEG_FP1_F7"
wavePattern="urn:mpeg:mpeg-v:01-CI-EEG_WavePatternCS-NS:EEG_Delta">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
   <siv:WaveValue mpeg7:dim="1 10">
      0.5 1.1 2.2 1.7 1.1 2.4 5.7 1.3 0.5 1.1
   </siv:WaveValue>
</iidl:SensedInfo>
```

6.39 ECG sensor type

6.39.1 General

This subclause specifies an electrocodiogram sensor type which detects a set of heart waves among the electrodes attached to the skin that are caused when the heart muscle depolarizes during each heartbeat. The applications of the sensor type may include medical use, health monitoring, physical interactive game and others.

6.39.2 Syntax

```
<!--################################ -->
<!--Definition of ECG sensor type  -->
<!--################################ -->
<complexType name="ECGSensorType">
   <complexContent>
      <extension base="siv:ElectrographSensorType">
      </extension>
   </complexContent>
</complexType>
```

6.39.3 Binary representation syntax

ECGSensorType{	***Number of bits***	***Mnemonic***
electrographSensorType	See above	electrographSensorType
}		

6.39.4 Semantics

Semantics of the `ECGSensorType`:

Name	*Definition*
`ECGSensorType`	Tool for describing sensed information with respect to an electrocardiogram (ECG) sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`WaveValue`	Describes the time series sensed value of the ECG sensor with respect to the microvolt (μV).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by UnitTypeCS defined in ISO/IEC 23005-6:—, A.2.1.
`waveformLabel`	Describes the label of the waveform based as a reference to a classification scheme term provided by ECG_wveformLabelCS defined in B.2. For ECG sensors, the name of each 12 leads is used as a waveform label.

Electrode label	**Electrode placement**
RA	On the right arm, avoiding bony prominences.
LA	In the same location that RA was placed, but on the left arm this time.
RL	On the right leg, avoiding bony prominences.
LL	In the same location that RL was placed, but on the left leg this time.
V1	In the *fourth* intercostal space (between ribs 4 and 5) just to the *right* of the sternum (breastbone).
V2	In the *fourth* intercostal space (between ribs 4 and 5) just to the *left* of the sternum.
V3	Between leads V2 and V4.
V4	In the fifth intercostal space (between ribs 5 and 6) in the mid-clavicular line (the imaginary line that extends down from the midpoint of the clavicle (collarbone).

Name	*Definition*

V5	Horizontally even with V4, but in the anterior axillary line. (The anterior axillary line is the imaginary line that runs down from the point midway between the middle of the clavicle and the lateral end of the clavicle; the lateral end of the collarbone is the end closer to the arm.)
V6	Horizontally even with V4 and V5 in the midaxillary line. (The midaxillary line is the imaginary line that extends down from the middle of the patient's armpit.)

[ECG electrode labels and their corresponding meaning]

WaveformLabel	**Description**
ECG_V1	The label of the waveform acquired from the electrode V1.
ECG_V2	The label of the waveform acquired from the electrode V2.
ECG_V3	The label of the waveform acquired from the electrode V3.
ECG_V4	The label of the waveform acquired from the electrode V4.
ECG_V5	The label of the waveform acquired from the electrode V5.
ECG_V6	The label of the waveform acquired from the electrode V6.
ECG_I	Lead I is the voltage between the (positive) left arm (LA) electrode and right arm (RA) electrode. (I=LA-RA)
ECG_II	Lead II is the voltage between the (positive) left leg (LL) electrode and right arm (RA) electrode. (II=LL-RA)
ECG_III	Lead III is the voltage between the (positive) left leg (LL) electrode and left arm (LA) electrode. (III=LL-LA)
ECG_aVR	Lead **augmented vector right (aVR)** has the positive electrode on the right arm. The negative electrode is a combination of the left arm electrode and the left leg electrode, which "augments" the signal strength of the positive electrode on the right arm. (aVR=RA-0.5(LA+LL))

Name	*Definition*

ECG_aVL	Lead **augmented vector left (aVL)** has the positive electrode on the left arm. The negative electrode is a combination of the right arm electrode and the left leg electrode, which "augments" the signal strength of the positive electrode on the left arm. (aVL=LA-0.5(RA+LL))
ECG_aVF	Lead **augmented vector foot (aVF)** has the positive electrode on the left leg. The negative electrode is a combination of the right arm electrode and the left arm electrode, which "augments" the signal of the positive electrode on the left leg. (aVF=LL-0.5(RA+LA))

[ECG waveform labels and their corresponding meaning]

The following table shall be used for binary representation.

Binary representation (5 bits)	**WaveformLabel**
0	ECG_V1
1	ECG_V2
2	ECG_V3
3	ECG_V4
4	ECG_V5
5	ECG_V6
6	ECG_I
7	ECG_II
8	ECG_III
9	ECG_aVR
10	ECG_aVL
11	ECG_aVF
12-31	reserved

`maxAmplitude`	Describes the maximum amplitude among the acquired time series value of the ECG sensor with respect to the microvolt (μV).

6.39.5 Examples

This example shows the description of an ECG sensing with the following semantics. The segmented sensed ECG signal wave acquired at the electrode V1 is composed of ten values of “0.5, 1.1, 2.2, 1.7, 1.1, 2.4, 5.7, 1.3, 0.5, 1.1”. The unit of the signal is millivolt. The ECG signal stream is obtained at the system clock tick of 60000 where there are 100 ticks per second. The id of this sensed information is ecg01 and the id of the sensor is ECGID_01.

```
<iidl:SensedInfo xsi:type="siv:ECGSensorType" id="ecg01" activate="true"
sensorIdRef="ECGID_01" unit="urn:mpeg:mpeg-v:01-CI-UnitTypeCS-NS:millivolt"
waveformLabel="urn:mpeg:mpeg-v:01-CI-ECG_WaveformLabelCS-NS:ECG_V1">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
   <siv:WaveValue mpeg7:dim="1 10">
      0.5 1.1 2.2 1.7 1.1 2.4 5.7 1.3 0.5 1.1
   </siv:WaveValue>
</iidl:SensedInfo>
```

6.40 EMG sensor type

6.40.1 General

This subclause specifies an electromyogram sensor type which detects a set of muscle tissure waves among the electrodes attached to skeletal muscles. The applications of the sensor type may include medical use, health monitoring, physical interactive game and others.

6.40.2 Syntax

```
<!--################################ -->
<!--Definition of EMG sensor type  -->
<!--################################ -->
<complexType name="EMGSensorType">
   <complexContent>
      <extension base="siv:ElectrographSensorType">
      </extension>
   </complexContent>
</complexType>
```

6.40.3 Binary representation syntax

EMGSensorType{	***Number of bits***	***Mnemonic***
electrographSensorType	See above	electrographSensorType
}		

6.40.4 Semantics

Semantics of the `EMGSensorType`:

Name	*Definition*
`EMGSensorType`	Tool for describing sensed information with respect to an electromyogram (EMG) sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`WaveValue`	Describes the time series sensed value of the EMG sensor with respect to the microvolt (μV).

Name	*Definition*
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by UnitTypeCS defined in ISO/IEC 23005-6:—, A.2.1.
`waveformLabel`	Describes the label of the waveform based as a reference to a classification scheme term provided by EMG_wveformLabelCS defined in B.3. NOTE each waveform label is defined by the name of the corresponding muscle for both fine wire sites and surface sites.

WaveformLabel	**Description**
EMG_SmallerFaceMuscles	Describes the waveform on the smaller face muscles
EMG_SmallerNeckMuscles	Describes the waveform on the smaller neck muscles
EMG_PectoralisMinor	Describes the waveform on the pectoralis minor, which is a thin, triangular muscle, situated at the upper part of the chest, beneath the pectoralis major
EMG_Diaphragma	Describes the waveform on the diaphragma
EMG_SmallerForearmMuscles	Describes the waveform on the smaller forearm muscles, which are the structure and distal region of the upper limb, between the elbow and the wrist
EMG_TransversusAbdominis	Describes the waveform on the transverses abdominis, which is a muscle layer of the anterior and lateral abdominal wall which is deep to (layered below) the internal oblique muscle
EMG_Iliacus	Describes the waveform on the iliacus, which is a flat, triangular muscle which fills the iliac fossa
EMG_PsoasMajor	Describes the waveform on the psoas major, which is a long fusiform muscle placed on the side of the thoracic region of the vertebral column and brim of the lesser pelvis
EMG_Adductors	Describes the waveform on the adductors, which are muscles of the thigh
EMG_VastusIntermedius	Describes the waveform on the vastus intermedius, which arises from the front and lateral surfaces of the body of the femur in its upper two-thirds, sitting under Rectus Femoris and from the lower part of the lateral intermuscular septum
EMG_ThinDeepShankMuscles	Describes the waveform on the thin deep shank muscles
EMG_SmallerFootMuscles	Describes the waveform on the smaller foot muscles
EMG_DeepNeckMuscles	Describes the waveform on the deep neck muscles
EMG_Supraspinatus	Describes the waveform on the Supraspinatus, which is a relatively small muscle of the upper limb

Name	Definition	
		that runs from the supraspinatous fossa superior of th scapula (shoulderblade) to the spine of the scapula
	EMG_Subscapularis	Describes the waveform on the subscapularis, which is a large triangular muscle which fills the subscapuplar fossa and inserts into the lesser tubercle of the humerus and the front of the capsule of the shoulder-joint
	EMG_Rhomboideus	Describes the waveform on the rhomboideus, which are rhombus-shaped muscles associated with the scapula and are chiefly responsible for its retraction
	EMG_TeresMajorMinor	Describes the waveform on the teres major, which is a muscle of the upper limb and one of six scapulohumeral muscles
	EMG_ThoracicErectorSpinae	Describes the waveform on the thoracic erector spinae
	EMG_TricepsBrachiiCMed	Describes the waveform on the triceps brachii c med
	EMG_DeepSegmentalErectorSpinae	Describes the waveform on the deep segmental erector spinae
	EMG_QuadratusLumborum	Describes the waveform on the quadrates lumborum, which is irregular and irregular and quadrilateral in shape, and broader below than above
	EMG_SmallerForearmExtensors	Describes the waveform on the smaller forearm extensors
	EMG_DeepMultifii	Describes the waveform on the deep multifii
	EMG_DeepHipMuscles	Describes the waveform on the deep hip muscles
	EMG_Frontalis	Describes the waveform on the frontalis, which is thin, of a quadrilateral form, and intimately adherent to the superficial fascia
	EMG_Masseter	Describes the waveform on the masseter, which is a thick, somewhat quadrilateral muscle, consisting of two parts, superficial and deep
	EMG_Sternocleidomastoideus	Describes the waveform on the sternocleidomastoideus, which is a paired muscle in the superficial layers of the anterior portion of the neck
	EMG_DeltoideusPAcromialis	Describes the waveform on the deltoideus p. acromialis
	EMG_DeltoideusPClavicularis	Describes the waveform on the deltoideus p. clavicularis
	EMG_PectoralisMajor	Describes the waveform on the pectoralis major, which is a thick, fan-shaped muscle, situated at the chest (anterior) of the body
	EMG_BicepsBrachii	Describes the waveform on the biceps brachii, which is a muscle located on the upper arm

Name	*Definition*
EMG_SerratusAnterior	Describes the waveform on the serratus anterior, which is a muscle that originates on the surface of the upper eight or nine ribs at the side of the chest and inserts along the entire anterior length of the medial border of the scapula
EMG_RectusAbdominis	Describes the waveform on the rectus abdominis, which is a paired muscle running vertically on each side of the anterior wall of the human abdomen
EMG_Brachioradialis	Describes the waveform on the brachioradialis, which is a muscle of the forearm that acts to flex the forearm at the elbow
EMG_FlexorCarpum Radialis	Describes the waveform on the flexor carpum radialis, which is a muscle of the human forearm that acts to flex and abduct the hand
EMG_FlexorCarpum Ulnaris	Describes the waveform on the flexor carpum ulnaris, which is a muscle of the human forearm that acts to flex and adduct the hand
EMG_ObliquusExternusAbdominis	Describes the waveform on the obliquus externus abdominis, which is the largest and superficial (outermost) of the three flat muscles of the lateral anterior abdomen
EMG_InternusTransversusAbdominis	Describes the waveform on the internus/transverses abdominis, which is a muscle layer of the anterior and lateral abdominal wall which is deep to (layered below) the internal oblique muscle
EMG_TensorFasciaLatae	Describes the waveform on the tensor fascia latae, which arises from the posterior part of the outer lip of the iliac crest; from the outer surface of the anterior superior iliac spine, and part of the outer border of the notch below it, between the gluteus medius and Sartorius; and from the deep surface of the fascia lata
EMG_Interosseus	Describes the waveform on the interosseus
EMG_Adductores	Describes the waveform on the adductores
EMG_RectusFemoris	Describes the waveform on the rectus femoris, which is one of the four qualdriceps muscles of the human body
EMG_VastusLateralis	Describes the waveform on the vastus lateralis, which is the largest part of the Quadriceps femoris
EMG_VastusMedialis	Describes the waveform on the vastus medialis, which is a medially located muscle of the quadriceps
EMG_PeroneusLongus	Describes the waveform on the peroneus longus, which is a superficial muscle in the lateral compartment of the leg, and acts to evert and plantar flex the ankle
EMG_TibialisAnterior	Describes the waveform on the tibialis anterior, which is a muscle that originates in the upper two-

Name	Definition	
		thirds of the lateral surface of the tibia and inserts into the medial cuneiform and first metatarsal bones of the foot
	EMG_NeckExtensors	Describes the waveform on the neck extensors
	EMG_TrapeziusPDescendenz	Describes the waveform on the trapezius p. descendenz
	EMG_TrapeziusPTransversus	Describes the waveform on the trapezius p. transversus
	EMG_DeltoideusPScapularis	Describes the waveform on the deltoideus p. scapularis
	EMG_Infraspinatus	Describes the waveform on the infraspinatus, which is a thick triangular muscle. It occupies the chief part of the infraspinatous fossa
	EMG_TrapeziusPAscendenz	Describes the waveform on the trapezius p. ascendenz
	EMG_TricepsBrachii	Describes the waveform on the triceps brachii, which is the large muscle on the back of the upper limb of many vertebrates
	EMG_LatissimusDorsi	Describes the waveform on the latissimus dorsi, which is the larger, flat, dorso-lateral muscle on the trunk, posterior to the arm, and partly covered by the trapezius on its median dorsal region
	EMG_ErectorSpinaeThoracicRegion	Describes the waveform on the erector spinae thoracic region
	EMG_ErectorSpinaeLumbarRegion	Describes the waveform on the erector spinae lumbar region
	EMG_SmallerForcearmExtensors	Describes the waveform on the smaller forcearm extensors
	EMG_MultifidusLumbarRegion	Describes the waveform on the multifidus lumbar region
	EMG_GlutaeusMedius	Describes the waveform on the glutaeus medius, which is a broad, thick, radiating muscle, situated on the outer surface of the pelvis
	EMG_GlutaeusMaximus	Describes the waveform on the glutaeus maximus, which is the largest and most superficial of the three gluteal muscles
	EMG_BicepsFemoris	Describes the waveform on the biceps femoris, which is a muscle of the posterior (the back) thigh
	EMG_Semitendinosus	Describes the waveform on the semitendinosus, which is a muscle in the back of the thigh
	EMG_GastrocnemiusLat	Describes the waveform on the gastrocnemius lat
	EMG_GastrocemiusMed	Describes the waveform on the gastrocemius med
	EMG_Soleus	Describes the waveform on the soleus, which is a powerful muscle in the back part of the lower leg (the calf)

[EMG waveform labels and their corresponding meaning]

Name	*Definition*

The following table shall be used for binary representation.

Binary representation (7 bits)	**WaveformLabel**
0	EMG_SmallerFaceMuscles
1	EMG_SmallerNeckMuscles
2	EMG_PectoralisMinor
3	EMG_Diaphragma
4	EMG_SmallerForearmMuscles
5	EMG_TransversusAbdominis
6	EMG_Iliacus
7	EMG_PsoasMajor
8	EMG_Adductors
9	EMG_VastusIntermedius
10	EMG_ThinDeepShankMuscles
11	EMG_SmallerFootMuscles
12	EMG_DeepNeckMuscles
13	EMG_Supraspinatus
14	EMG_Subscapularis
15	EMG_Rhomboideus
16	EMG_TeresMajorMinor
17	EMG_ThoracicErectorSpinae
18	EMG_TricepsBrachiiCMed
19	EMG_DeepSegmentalErectorSpinae
20	EMG_QuadratusLumborum
21	EMG_SmallerForearmExtensors
22	EMG_DeepMultifii
23	EMG_DeepHipMuscles
24	EMG_Frontalis
25	EMG_Masseter
26	EMG_Sternocleidomastoideus
27	EMG_DeltoideusPAcromialis
28	EMG_DeltoideusPClavicularis
29	EMG_PectoralisMajor
30	EMG_BicepsBrachii
31	EMG_SerratusAnterior
32	EMG_RectusAbdominis

Name	Definition	
	33	EMG_Brachioradialis
	34	EMG_FlexorCarpumRadialis
	35	EMG_FlexorCarpumUlnaris
	36	EMG_ObliquusExternusAbdominis
	37	EMG_InternusTransversusAbdominis
	38	EMG_TensorFasciaLatae
	39	EMG_Interosseus
	40	EMG_Adductores
	41	EMG_RectusFemoris
	42	EMG_VastusLateralis
	43	EMG_VastusMedialis
	44	EMG_PeroneusLongus
	45	EMG_TibialisAnterior
	46	EMG_NeckExtensors
	47	EMG_TrapeziusPDescendenz
	48	EMG_TrapeziusPTransversus
	49	EMG_DeltoideusPScapularis
	50	EMG_Infraspinatus
	51	EMG_TrapeziusPAscendenz
	52	EMG_TricepsBrachii
	53	EMG_LatissimusDorsi
	54	EMG_ErectorSpinaeThoracicRegion
	55	EMG_ErectorSpinaeLumbarRegion
	56	EMG_SmallerForcearmExtensors
	57	EMG_MultifidusLumbarRegion
	58	EMG_GlutaeusMedius
	59	EMG_GlutaeusMaximus
	60	EMG_BicepsFemoris
	61	EMG_Semitendinosus
	62	EMG_GastrocnemiusLat
	63	EMG_GastrocemiusMed
	64	EMG_Soleus
	65-128	reserved
`maxAmplitude`	Describes the maximum amplitude among the acquired time series value of	

Name	*Definition*
	the EMG sensor with respect to the microvolt (μV).

6.40.5 Examples

This example shows the description of an EMG sensing with the following semantics. The segmented sensed EMG signal wave acquired on the smaller face muscles is composed of ten values of “0.5, 1.1, 2.2, 1.7, 1.1, 2.4, 5.7, 1.3, 0.5, 1.1”. The unit of the signal is millivolt. The EMG signal stream is obtained at the system clock tick of 60000 where there are 100 ticks per second. The id of this sensed information is emg01 and the id of the sensor is EMGID_01.

```
<iidl:SensedInfo xsi:type="siv:EMGSensorType" id="emg01" activate="true"
sensorIdRef="EMGID_01" unit="urn:mpeg:mpeg-v:01-CI-UnitTypeCS-NS:millivolt"
waveformLabel="urn:mpeg:mpeg-v:01-CI-EMG_WaveformLabelCS-
NS:EMG_SmallerFaceMuscles">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
   <siv:WaveValue mpeg7:dim="1 10">
      0.5 1.1 2.2 1.7 1.1 2.4 5.7 1.3 0.5 1.1
   </siv:WaveValue>
</iidl:SensedInfo>
```

6.41 EOG sensor type

6.41.1 General

This subclause specifies an electrooculogram sensor type which detects a set of the resting potential waves among the electrodes attached to positions around eyes. The applications of the sensor type may include medical use, health monitoring, physical interactive game and others.

6.41.2 Syntax

```
<!--################################ -->
<!--Definition of EOG sensor type  -->
<!--################################ -->
<complexType name="EOGSensorType">
   <complexContent>
   <extension base="siv:ElectrographSensorType">
   </extension>
   </complexContent>
</complexType>
```

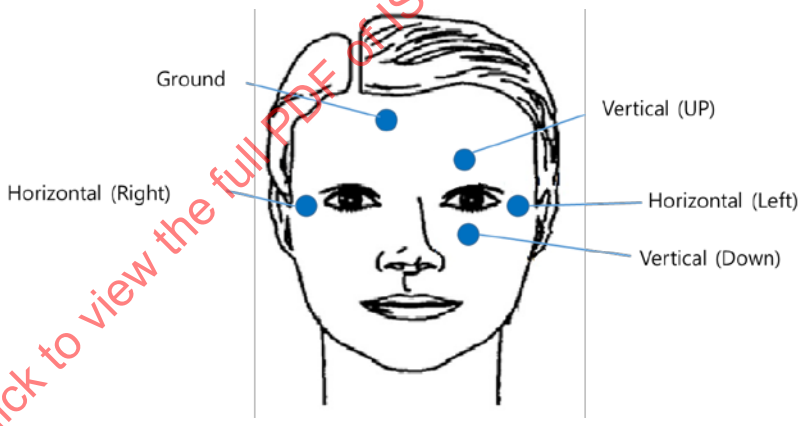
6.41.3 Binary representation syntax

EEGSensorType{	***Number of bits***	***Mnemonic***
electrographSensorType	See above	electrographSensorType
}		

6.41.4 Semantics

Semantics of the `EOGSensorType`:

Name	*Definition*
`EOGSensorType`	Tool for describing sensed information with respect to an electro-oculogram (EOG) sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`WaveValue`	Describes the time series sensed value of the EOG sensor with respect to the microvolt (μV).
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by UnitTypeCS defined in ISO/IEC 23005-6:—, A.2.1.
`waveformLabel`	Describes the label of the waveform based as a reference to a classification scheme term provided by EOG_wveformLabelCS defined in B.4.



[Electrode locations of EOG]

WaveformLabel	Description
EOG_VerticalUp	Describes the waveform between Ground and Vertical (Up)
EOG_VerticalDown	Describes the waveform between Ground and Vertical (Down)
EOG_HorizontalRight	Describes the waveform between Ground and Horizontal (Right)
EOG_HorizontalLeft	Describes the waveform between Ground and Horizontal (Left)
EOG_VerticalUD	Describes the waveform between Vertical (Up) and Vertical (Down)

Name	*Definition*
	EOG_HorizontalRL: Describes the waveform between Horizontal (Right) and Horizontal (Left)

[EOG waveform labels and their corresponding meaning]

The following table shall be used for binary representation.

Binary representation (4 bits)	**WaveformLabel**
0	EOG_VerticalUp
1	EOG_VerticalDown
2	EOG_HorizontalRight
3	EOG_HorizontalLeft
4	EOG_VerticalUD
5	EOG_HorizontalRL
6-15	reserved

`maxAmplitude`	Describes the maximum amplitude among the acquired time series value of the EOG sensor with respect to the microvolt (μV).

6.41.5 Examples

This example shows the description of an EOG sensing with the following semantics. The segmented sensed EOG signal wave acquired between ground and vertical electrodes is composed of ten values of “0.5, 1.1, 2.2, 1.7, 1.1, 2.4, 5.7, 1.3, 0.5, 1.1”. The unit of the signal is millivolt. The EOG signal stream is obtained at the system clock tick of 60000 where there are 100 ticks per second. The id of this sensed information is emg01 and the id of the sensor is EOGID_01.

```
<iidl:SensedInfo xsi:type="siv:EOGSensorType" id="eog01" activate="true"
sensorIdRef="EOGID_01" unit="urn:mpeg:mpeg-v:01-CI-UnitTypeCS-NS:millivolt"
waveformLabel="urn:mpeg:mpeg-v:01-CI-EOG_WaveformLabelCS-NS:EOG_VerticalUp">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000" />
   <siv:WaveValue mpeg7:dim="1 10">
         0.5 1.1 2.2 1.7 1.1 2.4 5.7 1.3 0.5 1.1
   </siv:WaveValue>
</iidl:SensedInfo>
```

6.42 GSR sensor type

6.42.1 General

This subclause specifies a galvanic skin response sensor type which detects a set of the electrical conductance waves of skin, which varies with its moisture level. The applications of the sensor type may include medical use, health monitoring, physical interactive game and others.

6.42.2 Syntax

```
<!--##################################### -->
<!--Definition of GSR sensor type        -->
<!--##################################### -->
<complexType name="GSRSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <sequence>
            <element name="Array_Value" type="mpeg7:FloatMatrixType"/>
         </sequence>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.42.3 Binary representation syntax

GSRSensorType{	***Number of bits***	***Mnemonic***
unitFlag	1	bslbf
SensedInfoBaseType	See above	SensedInfoBaseType
numOfChannels	16	uimsbf
numOfSamples	16	uimsbf
for(k = 0; k< numOfSamples; k++){		
for(j=0;j< numOfChannels;j++){		
array_value [(k * numOfChannels + j]	32	fsbf
}		
}		
If (unitFlag == 1){		
unit	8	bslbf
}		
}		

6.42.4 Semantics

Semantics of the `GSRSensorType`:

Name	*Definition*
`GSRSensorType`	Tool for describing sensed information with respect to a galvanic skin response (GSR) sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).

Name	*Definition*
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by UnitTypeCS defined in ISO/IEC 23005-6:—, A.2.1.
`Array_Value`	Describes the sensed value of the GSR with respect to the micromho (μ℧). Mho is unit of electrical conductance and the reciprocal of an ohm (Ω).
`unitFlag`	This field, which is only present in the binary representation, signals if a unit other than default unit is used. A value of "1" indicates that the unit specified in the unit attribute shall be used and "0" indicates that the default unit shall be used.
`numOfChannels`	This field, which is only present in the binary representation, represents the number of channels.
`numOfSamples`	This field, which is only present in the binary representation, represents the sample number of the sensed information for each channel.

6.42.5 Examples

This example shows the description of a GSR sensing with the following semantics. The sensor has an ID of "GSRS001" and references "GSRSID001". The sensor shall be activated and the array value with 2 channels shall be [0.3 0.5] (μ℧). The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:SensedInfo xsi:type="siv:GSRSensorType" id="GSRS001"
sensorIdRef="GSRSID001" activate="true">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
   <siv:Array_Value mpeg7:dim="2">0.3 0.5</siv:Array_Value>
</iidl:SensedInfo>
```

6.43 Bio sensor type

6.43.1 General

This Subclause specifies an aggregated sensor type which contains sensed information such as body height, body weight, body temperature, body fat, blood type, blood pressure, blood sugar, blood oxygen, heart rate, EEG, ECG, EMG, EOG, and GSR. The aggregated sensor type may contain just a subset of the sensed information. Moreover, the bio sensor type does not specify any sensing methods such as chemical and bio-chemical technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The applications of the sensor type may include medical use, health monitoring, physical interactive game and others.

6.43.2 Syntax

```
<!-- ############################################ -->
<!--  Definition of bio sensor type               -->
<!-- ############################################ -->
<complexType name="BioSensorType">
  <complexContent>
```

```
    <extension base="iidl:SensedInfoBaseType">
      <sequence>
        <element name="BodyHeight" type="siv:BodyHeightSensorType" minOccurs="0"/>
        <element name="BodyWeight" type="siv:BodyWeightSensorType" minOccurs="0"/>
        <element name="BodyTemperature" type="siv:BodyTemperatureSensorType"
minOccurs="0"/>
        <element name="BodyFat" type="siv:BodyFatSensorType" minOccurs="0"/>
        <element name="BloodType" type="siv:BloodTypeSensorType" minOccurs="0"/>
        <element name="BloodPressure" type="siv:BloodPressureSensorType"
minOccurs="0"/>
        <element name="BloodSugar" type="siv:BloodSugarSensorType" minOccurs="0"/>
        <element name="BloodOxygen" type="siv:BloodOxygenSensorType"
minOccurs="0"/>
        <element name="HeartRate" type="siv:HeartRateSensorType" minOccurs="0"/>
        <element name="EEG" type="siv:EEGSensorType" minOccurs="0"/>
        <element name="ECG" type="siv:ECGSensorType" minOccurs="0"/>
        <element name="EMG" type="siv:EMGSensorType" minOccurs="0"/>
        <element name="EOG" type="siv:EOGSensorType" minOccurs="0"/>
        <element name="GSR" type="siv:GSRSensorType" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

6.43.3 Binary representation syntax

BioSensorType {	*Number of bits*	*Mnemonic*
BodyHeightFlag	1	bslbf
BodyWeightFlag	1	bslbf
BodyTemperatureFlag	1	bslbf
BodyFatFlag	1	bslbf
BloodTypeFlag	1	bslbf
BloodPressureFlag	1	bslbf
BloodSugarFlag	1	bslbf
BloodOxygenFlag	1	bslbf
HeartRateFlag	1	bslbf
EEGFlag	1	bslbf
ECGFlag	1	bslbf
EMGFlag	1	bslbf
EOGFlag	1	bslbf
GSRFlag	1	bslbf

SensedInfoBaseType		SensedInfoBaseTypeType
if(BodyHeightFlag) {		
BodyHeight		BodyHeightSensorType
}		
if(BodyWeightFlag) {		
BodyWeight		BodyWeightSensorType
}		
if(BodyTemperatureFlag) {		
BodyTemperature		BodyTemperatureSensorType
}		
if(BodyFatFlag) {		
BodyFat		BodyFatSensorType
}		
if(BloodTypeFlag) {		
BloodType		BloodTypeSensorType
}		
if(BloodPressureFlag) {		
BloodPressure		BloodPressureSensorType
}		
if(BloodSugarFlag) {		
BloodSugar		BloodSugarSensorType
}		
if(BloodOxygenFlag) {		
BloodOxygen		BloodOxygenSensorType
}		
if(HeartRateFlag) {		
HeartRate		HeartRateSensorType
}		
if(EEGFlag) {		

EEG		EEGSensorType
}		
if(ECGFlag) {		
ECG		ECGSensorType
}		
if(EMGFlag) {		
EMG		EMGSensorType
}		
if(EOGFlag) {		
EOG		EOGSensorType
}		
if(GSRFlag) {		
GSR		GSRSensorType
}		
}		

6.43.4 Semantics

Semantics of the `BioSensorType`:

Name	*Definition*
`BioSensorType`	Tool for describing sensed information with respect to a bio sensor.
`BodyHeight`	Describes sensed information with respect to a body height sensor.
`BodyWeight`	Describes sensed information with respect to a body weight sensor.
`BodyTemperature`	Describes sensed information with respect to a body temperature sensor.
`BodyFat`	Describes sensed information with respect to a body fat sensor.
`BloodType`	Describes sensed information with respect to a blood type sensor.
`BloodPressure`	Describes sensed information with respect to a blood pressure sensor.
`BloodSugar`	Describes sensed information with respect to a blood sugar sensor.

Name	*Definition*
`BloodOxygen`	Describes sensed information with respect to a blood oxygen sensor.
`HeartRate`	Describes sensed information with respect to a heart rate sensor.
`EEG`	Describes sensed information with respect to an EEG sensor.
`ECG`	Describes sensed information with respect to an ECG sensor.
`EMG`	Describes sensed information with respect to an EMG sensor.
`EOG`	Describes sensed information with respect to an EOG sensor.
`GSR`	Describes sensed information with respect to a GSR sensor.
`BodyHeightFlag`	This field, which is only present in the binary representation, signals if body height sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.
`BodyWeightFlag`	This field, which is only present in the binary representation, signals if body weight sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.
`BodyTemperatureFlag`	This field, which is only present in the binary representation, signals if body temperature sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.
`BodyFatFlag`	This field, which is only present in the binary representation, signals if body fat sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.
`BloodTypeFlag`	This field, which is only present in the binary representation, signals if blood type sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.
`BloodPressureFlag`	This field, which is only present in the binary representation, signals if blood pressure sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.
`BloodSugarFlag`	This field, which is only present in the binary representation, signals if blood sugar sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.
`BloodOxygenFlag`	This field, which is only present in the binary representation, signals if blood oxygen sensed information is available. A value of “1” indicates that the sensed information shall be included and “0”

Name	*Definition*
	indicates that the sensed information shall not be included.
HeartRateFlag	This field, which is only present in the binary representation, signals if heart rate sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.
EEGFlag	This field, which is only present in the binary representation, signals if EEG sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.
ECGFlag	This field, which is only present in the binary representation, signals if ECG sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.
EMGFlag	This field, which is only present in the binary representation, signals if EMG sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.
EOGFlag	This field, which is only present in the binary representation, signals if EOG sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.
GSRFlag	This field, which is only present in the binary representation, signals if GSR sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.

6.43.5 Examples

This example shows the description of aggregated sensed information acquired from a bio-sensor with the following semantics. The sensed information is obtained at the system clock tick of 60000 where there are 100 ticks per second. The id of this sensed information is bio01 and the id of the sensor is BIOID_01. The sensed height is 180 cm, the sensed body fat is 13.5 %, and the sensed heart rate is 60 Hz.

```
<iidl:SensedInfo xsi:type="siv:BioSensorType" sensorIdRef="BIO_01" id="bio01">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
   <siv:BodyHeight xsi:type="siv:BodyHeightSensorType" value="180"
unit="urn:mpeg:mpeg-v:01-CI-UnitTypeCS-NS:cm"/>
   <siv:BodyFat xsi:type="siv:BodyFatSensorType" value="13.5" />
   <siv:HeartRate xsi:type="siv:HeartRateSensorType" value="60" />
</iidl:SensedInfo>
```

6.44 Weather sensor type

6.44.1 General

This subclause specifies XML syntax, binary representation syntax, and semantics of the WeatherSensorType with an example instantiation of the sensed information. Unlike most of other sensed information vocabularies, the weather sensor may be considered as a virtual sensor composed of various unit sensors such as temperature, humidity, snow, and precipitation sensors. In practice, the sensed information from this type of sensor may be either acquired by a number of actual sensors or fed by a weather related service.

6.44.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of seather sensor type               -->
<!-- ################################################ -->
<complexType name="WeatherSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <sequence>
            <element name="WeatherDescription" type="mpeg7:termReferenceType"
maxOccurs="unbounded"/>
            <element name="Temperature" type="siv:TemperatureSensorType"
minOccurs="0"/>
            <element name="Precipitation" minOccurs="0" maxOccurs="unbounded">
               <complexType>
                  <attribute name="value" type="float"/>
                  <attribute name="valueUnit" type="mpegvct:unitType"
use="optional"/>
                  <attribute name="duration" type="integer"/>
                  <attribute name="durationUnit" type="mpegvct:unitType"
use="optional"/>
               </complexType>
            </element>
            <element name="Snow" minOccurs="0" maxOccurs="unbounded">
               <complexType>
                  <attribute name="value" type="float"/>
                  <attribute name="valueUnit" type="mpegvct:unitType"
use="optional"/>
                  <attribute name="duration" type="integer"/>
                  <attribute name="durationUnit" type="mpegvct:unitType"
use="optional"/>
               </complexType>
            </element>
            <element name="Wind" minOccurs="0">
               <complexType>
                  <attribute name="velocity" type="float"/>
                  <attribute name="unit" type="mpegvct:unitType"
use="optional"/>
                  <attribute name="direction" type="mpeg7:termReferenceType"/>
               </complexType>
            </element>
            <element name="Humidity" type="siv:HumiditySensorType"
minOccurs="0"/>
            <element name="Place" type="mpeg7:PlaceType" minOccurs="0"/>
         </sequence>
         <attribute name="time" type="siv:timePointType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

```
<!-- Definition of timePoint datatype which is the same as the one in MPEG-7-->
<simpleType name="timePointType">
<restriction base="string">
<pattern value="(\-?\d+(\-\d{2}(\-
\d{2})?)?)?(T\d{2}(:\d{2}(:\d{2}(:\d+)?)?)?)?(F\d+)?((\-|\+)\d{2}:\d{2})?"/>
</restriction>
</simpleType>
```

6.44.3 Binary representation syntax

WeatherSensorType{	***Number of bits***	***Mnemonic***
SensedInfoBase		SensedInfoBaseType
TemperatureFlag	1	bslbf
PrecipitationFlag	1	bslbf
SnowFlag	1	bslbf
WindFlag	1	bslbf
HumidityFlag	1	bslbf
PlaceFlag	1	bslbf
TimeFlag	1	bslbf
WeatherDescriptionCount	32	uimsbf
PrecipitationCount	32	uimsbf
SnowCount	32	uimsbf
For (i=1;i<WeatherDescriptionCount;i++) {		
WeatherDescription	4	WeatherCS
}		
If (TemperatureFlag) {		
TemperatureSensor		TemperatureSensorType
}		
If (PrecipitationFlag) {		
Value	32	Flbf
valueUnitFlag	1	bslbf
If (valueUnitFlag) {		

valueUnit	8	bslbf
}		
duration	32	Simsbf
durationUnitFlag	1	bslbf
if (durationUnitFlag) {		
durationUnit	8	bslbf
}		
}		
If (SnowFlag) {		
value	32	flbf
valueUnitFlag	1	bslbf
If (valueUnitFlag) {		
valueUnit	8	bslbf
}		
duration	32	simsbf
durationUnitFlag	1	bslbf
If (durationUnitFlag) {		
durationUnit	8	bslbf
}		
}		
If (WindFlag) {		
velocity	32	flbf
velocityUnitFlag	1	bslbf
If (velocityUnitFlag) {		
velocityUnit	8	bslbf
}		
direction	5	WindDirectionTypeCS
}		
If (HumidityFlag) {		

Humidity		HumiditySensorType
}		
If (PlaceFlag) {		
Place	See ISO/IEC 10646	UTF-8
}		
If (timeFlag) {		
Time	See ISO/IEC 10646	UTF-8
}		
}		

6.44.4 Semantics

Semantics of the `WeatherSensorType`:

Names	Description
`WeatherType`	Tool for describing the observed weather.
`WeatherDescription`	Describes the observed weather as a reference to a classification scheme term provided by `WeatherCS` defined in B.7 using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. Binary representation of WeatherCS is as follows: Weather: Term ID of weather 0001: Sunny 0010: Cloudy 0011: Rainy 0100: Snow 0101: Windy 0110: Partly cloudy 0111~1111: Reserved
`Temperature`	Describes the temperature using the structure defined by `TemperatureSensorType`.
`Precipitation`	Describes the precipitation during the specified period of time as defined by the `duration` attribute in the default unit of milimetre or

	in the unit specified by the `valueUnit` attribute.
`value`	Specifies the precipitation in the default unit of milimetre or in the unit specified by the `valueUnit` attribute.
`valueUnit`	Specifies the unit of the precipitation value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1 using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6.
`duration`	Specifies the time period up to the time of measuring the precipitation in the default unit of hour or in the unit specified by `durationUnit` attribute.
`durationUnit`	Specifies the unit of the duration, if a unit other than the default unit is used, as a reference to a classification scheme term provided by `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1 using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6.
`Snow`	Describes the amount of snowfall during the specified period of time as defined by the `duration` attribute in the default unit of centimetre or in the unit specified by the `valueUnit` attribute.
`value`	Specifies the amount of snowfall in the default unit of centimetre or in the unit specified by the `valueUnit` attribute.
`valueUnit`	Specifies the unit of the snowfall value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1 using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6.
`duration`	Specifies the time period up to the time of measuring the amount of snowfall in the default unit of hour or in the unit specified by `durationUnit` attribute.
`durationUnit`	Specifies the unit of the duration, if a unit other than the default unit is used, as a reference to a classification scheme term provided by `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1 using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6.
`Wind`	Describes the strength and the direction of the wind.
`velocity`	Specifies the strength of the wind in metre per second by default, or in the unit specified by the `unit` attribute.
`unit`	Specifies the unit of the wind strength, if a unit other than the default unit is used, as a reference to a classification scheme term provided by `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1 using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6.
`Direction`	Specifies the direction of the wind coming from, as a reference to a classification scheme term provided by `WindDirectionTypeCS` defined in B.8 using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. Binary representation of WindDirectionTypeCS is as follows:

	<table><tr><th>WindDirectionType</th><th>Term ID of WindDirectionType</th></tr><tr><td>00001</td><td>N</td></tr><tr><td>00010</td><td>W</td></tr><tr><td>00011</td><td>S</td></tr><tr><td>00100</td><td>E</td></tr><tr><td>00101</td><td>NW</td></tr><tr><td>00110</td><td>NE</td></tr><tr><td>00111</td><td>SW</td></tr><tr><td>01000</td><td>SE</td></tr><tr><td>01001</td><td>NNW</td></tr><tr><td>01010</td><td>WNW</td></tr><tr><td>01011</td><td>NNE</td></tr><tr><td>01100</td><td>ENE</td></tr><tr><td>01101</td><td>SSW</td></tr><tr><td>01110</td><td>WSW</td></tr><tr><td>01111</td><td>SSE</td></tr><tr><td>10000</td><td>ESE</td></tr><tr><td>10001~11111</td><td>Reserved</td></tr></table>
`Humidity`	Describes the humidity using the structure defined by `HumiditySensorType`.
`PlaceLength`	This field, which is only present in the binry representation, specifies the length of the `PlaceLength` field encoded in binary representation in bytes.
`Place`	Describes the place where the given weather information is acquired.
`TimeLength`	This field, which is only present in the binry representation, specifies the length of the `TimeLength` field encoded in binary representation in bytes.
`Time`	Describes the time when the given weather information is acquired.

6.44.5 Examples

This example shows the description of Weather description. In this example, there is a virtual weather sensor with id "WeatherSensor01." This weather sensor gives weather information of a geographic position with latitude of 37 degrees north and longitude of 127 degrees east. The weather sensed at the media time of 600000 when 1000 represents one second, is snow with 100 centimetre for a period of one hour.

```
<iidl:SensedInfo xsi:type="siv:WeatherSensorType" id="Weather01"
sensorIdRef="WeatherSensor01" activate="true">
   <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="1000"
pts="600000"/>
   <siv:WeatherDescription>urn:mpeg:mpeg-v:01-CI-WeatherCS-
NS:snow</siv:WeatherDescription>
   <siv:Snow value="100" valueUnit="urn:mpeg:mpeg-v:01-CI-UnitTypeCS-NS:cm"
duration="1"/>
   <siv:Place>
      <mpeg7:GeographicPosition>
         <mpeg7:Point latitude="37" longitude="127"/>
      </mpeg7:GeographicPosition>
   </siv:Place>
</iidl:SensedInfo>
```

6.45 Facial expression sensor type

6.45.1 General

This subclause specifies a facial expression sensor type which senses relative displacement data of facial expression using a facial expression retargeting, a method to create facial animation for the different facial configurations of between source data and a target model. The applications of the sensor type may include avatar facial control and others.

6.45.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of facial expression sensor type     -->
<!-- ################################################ -->
<complexType name="FacialExpressionSensorType">
   <complexContent>
      <extension base="siv:IntelligentCameraType">
         <sequence>
            <element name="FacialExpressionBasis"
type="siv:FacialExpressionBasisType" minOccurs="0" maxOccurs="66"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>

<complexType name="FacialExpressionBasisType">
   <attribute name="facialExpressionBasisID" type="mpeg7:termReferenceType"
use="optional"/>
   <attribute name="facialExpressionBasisValue" type="float" use="optional"/>
   <attribute name="facialExpressionBasisUnit" type="mpegvct:unitType"
use="optional"/>
</complexType>
```

6.45.3 Binary representation syntax

FacialExpressionSensorType{	***Number of bits***	***Mnemonic***
FacialExpressionBasisFlag		
IntelligentCamera		IntelligentCameraType
if(FacialExpressionBasisFlag) {		
NumOfFacialExpressionBasis	7	uimsbf
for(k=0; k<NumOfFacialExpressionBasis; k++) {		
FacialExpressionBasis[k]		FacialExpressionBasisType
}		
}		
}		
FacialExpressionBasisType {		
facialExpressionBasisIDFlag	1	bslbf
facialExpressionBasisValueFlag	1	bslbf
facialExpressionBasisUnitFlag	1	bslbf
if(facialExpressionBasisIDFlag) {		
facialExpressionBasisID	8	bslbf
}		
if(facialExpressionBasisValueFlag) {		
facialExpressionBasisValue	32	fsbf
}		
if(facialExpressionBasisUnitFlag) {		
facialExpressionBasisUnit	8	bslbf
}		
}		

6.45.4 Semantics

Semantics of the `FacialExpressionSensorType`:

Name	*Definition*
`FacialExpressionSensorType`	Tool for describing a facial expression sensor.
`FacialExpressionBasisFlag`	This field, which is only present in the binary representation, signals the presence of the attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`NumOfFacialExpressionBasis`	This field, which is only present in the binary representation, indicates the number of facial expression basis in this sensed information.
`FacialExpressionBasis`	Describes each facial expression basis detected by the camera.
`FacialExpressionBasisType`	Tool for describing each facial expression basis.
`facialExpressionBasisIDFlag`	This field, which is only present in the binary representation, signals the presence of the attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`facialExpressionBasisValueFlag`	This field, which is only present in the binary representation, signals the presence of the attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`facialExpressionBasisUnitFlag`	This field, which is only present in the binary representation, signals the presence of the attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`facialExpressionBasisID`	Describes the identification of the associated facial expression basis based as a reference to the classification scheme term provided by FacialExpressionBasisIDCS defined in B.9.

FacialExpressionBasisID	**Description**
open_jaw	Describes the vertical jaw displacement
lower_t_midlip	Describes the vertical top middle inner lip displacement
raise_b_midlip	Describes the vertical bottom middle inner lip displacement
stretch_l_cornerlip	Describes the horizontal displacement of left inner lip corner
stretch_r_cornerlip	Describes the horizontal displacement of right inner lip corner

Name	*Definition*	
	lower_t_lip_lm	Describes the vertical displacement of midpoint between left corner and middle of top inner lip
	lower_t_lip_rm	Describes the vertical displacement of midpoint between right corner and middle of top inner lip
	raise_b_lip_lm	Describes the vertical displacement of midpoint between left corner and middle of bottom inner lip
	raise_b_lip_rm	Describes the vertical displacement of midpoint between right corner and middle of bottom inner lip
	raise_l_cornerlip	Describes the vertical displacement of left inner lip corner
	raise_r_cornerlip	Describes the vertical displacement of right inner lip corner
	thrust_jaw	Describes the depth displacement of jaw
	shift_jaw	Describes the side to side displacement of jaw
	push_b_lip	Describes the depth displacement of bottom middle lip
	push_t_lip	Describes the depth displacement of top middle lip
	depress_chin	Describes the upward and compressing movement of the chin (like in sadness)
	close_t_l_eyelid	Describes the vertical displacement of top left eyelid
	close_t_r_eyelid	Describes the vertical displacement of top right eyelid
	close_b_l_eyelid	Describes the vertical displacement of bottom left eyelid
	close_b_r_eyelid	Describes the vertical displacement of bottom right eyelid
	yaw_l_eyeball	Describes the horizontal orientation of left eyeball
	yaw_r_eyeball	Describes the horizontal orientation of right eyeball

Name	*Definition*
pitch_l_eyeball	Describes the vertical orientation of left eyeball
pitch_r_eyeball	Describes the vertical orientation of right eyeball
thrust_l_eyeball	Describes the depth displacement of left eyeball
thrust_r_eyeball	Describes the depth displacement of right eyeball
dilate_l_pupil	Describes the dilation of left pupil
dilate_r_pupil	Describes the dilation of right pupil
raise_l_i_eyebrow	Describes the vertical displacement of left inner eyebrow
raise_r_i_eyebrow	Describes the vertical displacement of right inner eyebrow
raise_l_m_eyebrow	Describes the vertical displacement of left middle eyebrow
raise_r_m_eyebrow	Describes the vertical displacement of right middle eyebrow
raise_l_o_eyebrow	Describes the vertical displacement of left outer eyebrow
raise_r_o_eyebrow	Describes the vertical displacement of right outer eyebrow
squeeze_l_eyebrow	Describes the horizontal displacement of left eyebrow
squeeze_r_eyebrow	Describes the horizontal displacement of right eyebrow
puff_l_cheek	Describes the horizontal displacement of left cheek
puff_r_cheek	Describes the horizontal displacement of right cheek
lift_l_cheek	Describes the vertical displacement of left cheek
lift_r_cheek	Describes the vertical displacement of right cheek
shift_tongue_tip	Describes the horizontal displacement of tongue tip
raise_tongue_tip	Describes the vertical displacement of tongue tip
thrust_tongue_tip	Describes the depth

Name	Definition	
		displacement of tongue tip
	raise_tongue	Describes the vertical displacement of tongue
	tongue_roll	Describes the rolling of the tongue into U shape
	head_pitch	Describes the head pitch angle from top of spine
	head_yaw	Describes the head yaw angle from top of spine
	head_roll	Describes the head roll angle from top of spine
	lower_t_midlip_o	Describes the vertical top middle outer lip displacement
	raise_b_midlip_o	Describes the vertical bottom middle outer lip displacement
	stretch_l_cornerlip_o	Describes the horizontal displacement of left outer lip corner
	stretch_r_cornerlip_o	Describes the horizontal displacement of right outer lip corner
	lower_t_lip_lm_o	Describes the vertical displacement of midpoint between left corner and middle of top outer lip
	lower_t_lip_rm_o	Describes the vertical displacement of midpoint between right corner and middle of top outer lip
	raise_b_lip_lm_o	Describes the vertical displacement of midpoint between left corner and middle of bottom outer lip
	raise_b_lip_rm_o	Describes the vertical displacement of midpoint between right corner and middle of bottom outer lip
	raise_l_cornerlip_o	Describes the vertical displacement of left outer lip corner
	raise_r_cornerlip_o	Describes the vertical displacement of right outer lip corner
	stretch_l_nose	Describes the horizontal displacement of left side of nose
	stretch_r_nose	Describes the horizontal displacement of right side of nose

Name	*Definition*
raise_nose	Describes the vertical displacement of nose tip
bend_nose	Describes the horizontal displacement of nose tip
raise_l_ear	Describes the vertical displacement of left ear
raise_r_ear	Describes the vertical displacement of right ear
pull_l_ear	Describes the horizontal displacement of left ear
pull_r_ear	Describes the horizontal displacement of right ear
push_b_lip	Describes the depth displacement of bottom middle lip
push_t_lip	Describes the depth displacement of top middle lip
depress_chin	Describes the upward and compressing movement of the chin (like in sadness)
close_t_l_eyelid	Describes the vertical displacement of top left eyelid
close_t_r_eyelid	Describes the vertical displacement of top right eyelid
close_b_l_eyelid	Describes the vertical displacement of bottom left eyelid
close_b_r_eyelid	Describes the vertical displacement of bottom right eyelid
yaw_l_eyeball	Describes the horizontal orientation of left eyeball
yaw_r_eyeball	Describes the horizontal orientation of right eyeball
pitch_l_eyeball	Describes the vertical orientation of left eyeball
pitch_r_eyeball	Describes the vertical orientation of right eyeball
thrust_l_eyeball	Describes the depth displacement of left eyeball
thrust_r_eyeball	Describes the depth displacement of right eyeball
dilate_l_pupil	Describes the dilation of left pupil
dilate_r_pupil	Describes the dilation of right pupil

Name	*Definition*
raise_l_i_eyebrow	Describes the vertical displacement of left inner eyebrow
raise_r_i_eyebrow	Describes the vertical displacement of right inner eyebrow
raise_l_m_eyebrow	Describes the vertical displacement of left middle eyebrow
raise_r_m_eyebrow	Describes the vertical displacement of right middle eyebrow
raise_l_o_eyebrow	Describes the vertical displacement of left outer eyebrow
raise_r_o_eyebrow	Describes the vertical displacement of right outer eyebrow
squeeze_l_eyebrow	Describes the horizontal displacement of left eyebrow
squeeze_r_eyebrow	Describes the horizontal displacement of right eyebrow
puff_l_cheek	Describes the horizontal displacement of left cheek
puff_r_cheek	Describes the horizontal displacement of right cheek
lift_l_cheek	Describes the vertical displacement of left cheek
lift_r_cheek	Describes the vertical displacement of right cheek
shift_tongue_tip	Describes the horizontal displacement of tongue tip
raise_tongue_tip	Describes the vertical displacement of tongue tip
thrust_tongue_tip	Describes the depth displacement of tongue tip
raise_tongue	Describes the vertical displacement of tongue
tongue_roll	Describes the rolling of the tongue into U shape
head_pitch	Describes the head pitch angle from top of spine
head_yaw	Describes the head yaw angle from top of spine
head_roll	Describes the head roll angle from top of spine
lower_t_midlip_o	Describes the vertical top middle

Name	Definition	
		outer lip displacement
	raise_b_midlip_o	Describes the vertical bottom middle outer lip displacement
	stretch_l_cornerlip_o	Describes the horizontal displacement of left outer lip corner
	stretch_r_cornerlip_o	Describes the horizontal displacement of right outer lip corner
	lower_t_lip_lm_o	Describes the vertical displacement of midpoint between left corner and middle of top outer lip
	lower_t_lip_rm_o	Describes the vertical displacement of midpoint between right corner and middle of top outer lip
	raise_b_lip_lm_o	Describes the vertical displacement of midpoint between left corner and middle of bottom outer lip
	raise_b_lip_rm_o	Describes the vertical displacement of midpoint between right corner and middle of bottom outer lip
	raise_l_cornerlip_o	Describes the vertical displacement of left outer lip corner
	raise_r_cornerlip_o	Describes the vertical displacement of right outer lip corner
	stretch_l_nose	Describes the horizontal displacement of left side of nose
	stretch_r_nose	Describes the horizontal displacement of right side of nose
	raise_nose	Describes the vertical displacement of nose tip
	bend_nose	Describes the horizontal displacement of nose tip
	raise_l_ear	Describes the vertical displacement of left ear
	raise_r_ear	Describes the vertical displacement of right ear
	pull_l_ear	Describes the horizontal displacement of left ear
	pull_r_ear	Describes the horizontal displacement of right ear

Name	*Definition*

[Facial Expression Basis ID]

The following table shall be used for binary representation.

Binary representation (8 bits)	**FacialExpressionBasisID**
0	open_jaw
1	lower_t_midlip
2	raise_b_midlip
3	stretch_l_cornerlip
4	stretch_r_cornerlip
5	lower_t_lip_lm
6	lower_t_lip_rm
7	raise_b_lip_lm
8	raise_b_lip_rm
9	raise_l_cornerlip
10	raise_r_cornerlip
11	thrust_jaw
12	shift_jaw
13	push_b_lip
14	push_t_lip
15	depress_chin
16	close_t_l_eyelid
17	close_t_r_eyelid
18	close_b_l_eyelid
19	close_b_r_eyelid
20	yaw_l_eyeball
21	yaw_r_eyeball
22	pitch_l_eyeball
23	pitch_r_eyeball
24	thrust_l_eyeball
25	thrust_r_eyeball
26	dilate_l_pupil
27	dilate_r_pupil
28	raise_l_i_eyebrow
29	raise_r_i_eyebrow
30	raise_l_m_eyebrow
31	raise_r_m_eyebrow
32	raise_l_o_eyebrow

Name	Definition	
	33	raise_r_o_eyebrow
	34	squeeze_l_eyebrow
	35	squeeze_r_eyebrow
	36	puff_l_cheek
	37	puff_r_cheek
	38	lift_l_cheek
	39	lift_r_cheek
	40	shift_tongue_tip
	41	raise_tongue_tip
	42	thrust_tongue_tip
	43	raise_tongue
	44	tongue_roll
	45	head_pitch
	46	head_yaw
	47	head_roll
	48	lower_t_midlip_o
	49	raise_b_midlip_o
	50	stretch_l_cornerlip_o
	51	stretch_r_cornerlip_o
	52	lower_t_lip_lm_o
	53	lower_t_lip_rm_o
	54	raise_b_lip_lm_o
	55	raise_b_lip_rm_o
	56	raise_l_cornerlip_o
	57	raise_r_cornerlip_o
	58	stretch_l_nose
	59	stretch_r_nose
	60	raise_nose
	61	bend_nose
	62	raise_l_ear
	63	raise_r_ear
	64	pull_l_ear
	65	pull_r_ear
	66	push_b_lip
	67	push_t_lip
	68	depress_chin
	69	close_t_l_eyelid

Name	Definition	
	70	close_t_r_eyelid
	71	close_b_l_eyelid
	72	close_b_r_eyelid
	73	yaw_l_eyeball
	74	yaw_r_eyeball
	75	pitch_l_eyeball
	76	pitch_r_eyeball
	77	thrust_l_eyeball
	78	thrust_r_eyeball
	79	dilate_l_pupil
	80	dilate_r_pupil
	81	raise_l_i_eyebrow
	82	raise_r_i_eyebrow
	83	raise_l_m_eyebrow
	84	raise_r_m_eyebrow
	85	raise_l_o_eyebrow
	86	raise_r_o_eyebrow
	87	squeeze_l_eyebrow
	88	squeeze_r_eyebrow
	89	puff_l_cheek
	90	puff_r_cheek
	91	lift_l_cheek
	92	lift_r_cheek
	93	shift_tongue_tip
	94	raise_tongue_tip
	95	thrust_tongue_tip
	96	raise_tongue
	97	tongue_roll
	98	head_pitch
	99	head_yaw
	100	head_roll
	101	lower_t_midlip_o
	102	raise_b_midlip_o
	103	stretch_l_cornerlip_o
	104	stretch_r_cornerlip_o
	105	lower_t_lip_lm_o
	106	lower_t_lip_rm_o

Name	*Definition*	
	107	raise_b_lip_lm_o
	108	raise_b_lip_rm_o
	109	raise_l_cornerlip_o
	110	raise_r_cornerlip_o
	111	stretch_l_nose
	112	stretch_r_nose
	113	raise_nose
	114	bend_nose
	115	raise_l_ear
	116	raise_r_ear
	117	pull_l_ear
	118	pull_r_ear
	119-255	Reserved
`facialExpressionBasisValue`	Describes the value of the associated facial expression basis. The minimum value shall be 0 % and the maximum value shall be 100 %.	
`facialExpressionBasisUnit`	Describes the unit of each facial expression basis. In default, a percent is used to provide the relative value with respect to the range of the value. NOTE 1 The value of each facial expression basis can be relatively obtained by the range provided by the `FacialExpressionCharacteristicsSensorType`. NOTE 2 The unit of each facial expression basis can also use the unit defined in ISO/IEC 14496-2:2004, Annex C.	

6.45.5 Examples

This first example shows a description of the facial expression sensing with the following semantics. The description of the facial expression sensor has identifier of “FES1” and the sensor references an actual sensor with ID of “FESIC1”. The basis value of facial expression with the ID of “open_jaw” shall be 50 %. Also, the basis value of facial expression with the ID of “raise_l_cornerlip” shall be 50 %.

```
<iidl:SensedInfo xsi:type="siv:FacialExpressionSensorType" id="FES1"
sensorIdRef="FESIC1" activate="true">
   <siv:FacialExpressionBasis facialExpressionBasisID="open_jaw"
facialExpressionBasisValue="50"/>
   <siv:FacialExpressionBasis facialExpressionBasisID="raise_l_cornerlip"
   facialExpressionBasisValue="50"/>
</iidl:SensedInfo>
```

This second example shows a description of the facial expression sensing with the following semantics. The description of the facial expression sensor has identifier of “FES1” and the sensor references an actual sensor with ID of “FESIC1”. The basis value of facial expression with the ID of “open_jaw” shall be 10 (mm). Also, the basis value of facial expression with the ID of “raise_l_cornerlip” shall be 5(mm).

```
<iidl:SensedInfo xsi:type="siv:FacialExpressionSensorType" id="FES1"
sensorIdRef="FESIC1" activate="true">
   <siv:FacialExpressionBasis facialExpressionBasisID="open_jaw"
facialExpressionBasisValue="10" facialExpressionBasisUnit="mm"/>
   <siv:FacialExpressionBasis facialExpressionBasisID="raise_l_cornerlip"
facialExpressionBasisValue="5"  facialExpressionBasisUnit="mm"/>
</iidl:SensedInfo>
```

6.46 Facial morphology sensor type

6.46.1 General

This subclause specifies a facial morphology sensor type which senses the initialization parameters supporting the facial expression sensor type for intelligent cameras, and is relaved to user-specific facial morphology data sensed from an intelligent camera. The applications of the sensor type may include avatar facial control and others.

6.46.2 Syntax

```
<!-- ################################################ -->
<!--  Definition of facial morphology sensor type     -->
<!-- ################################################ -->
<complexType name="FacialMorphologySensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <sequence>
            <element name="IrisDiameter" type="float" minOccurs="0"/>
            <element name="EyeSeparation" type="float" minOccurs="0"/>
            <element name="EyeNoseSeparation" type="float" minOccurs="0"/>
            <element name="MouseNoseSeparation" type="float" minOccurs="0"/>
            <element name="MouseWidth" type="float" minOccurs="0"/>
   </sequence>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>
```

6.46.3 Binary representation syntax

FacialMorphologySensor Type{	***Number of bits***	***Mnemonic***
IrisDiameterFlag	1	bslbf
EyeSeparationFlag	1	bslbf
EyeNoseSeparationFlag	1	bslbf
MouseNoseSeparationFlag	1	bslbf
MouseWidthFlag	1	bslbf

unitFlag	1	bslbf
SensedInfoBase		SensedInfoBaseType
if(IrisDiameterFlag) {		
IrisDiameter	32	fsbf
}		
if(EyeSeparationFlag) {		
EyeSeparation	32	fsbf
}		
if(EyeNoseSeparationFlag) {		
EyeNoseSeparation	32	fsbf
}		
if(MouseNoseSeparationFlag) {		
MouseNoseSeparation	32	fsbf
}		
if(MouseWidthFlag) {		
MouseWidth	32	fsbf
}		
if(unitFlag) {		
unitFlag	8	bslbf
}		
}		

6.46.4 Semantics

Semantics of the `FacialMorphologySensorType`:

Name	*Definition*
`FacialMorphologySensorType`	Tool for describing a facial morphology sensor sensed information.
`IrisDiameterFlag`	This field, which is only present in the binary representation, signals the presence of the activation attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`EyeSeparationFlag`	This field, which is only present in the binary representation, signals the presence of the activation attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`EyeNoseSeparationFlag`	This field, which is only present in the binary representation, signals the presence of the activation attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`MouseNoseSeparationFlag`	This field, which is only present in the binary representation, signals the presence of the activation attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`MouseWidthFlag`	This field, which is only present in the binary representation, signals the presence of the activation attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of the activation attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`IrisDiameter`	Describes iris diameter (by definition it is equal to the distance between upper and lower eyelid) in neutral face. Depicted as A in Figure 4.
`EyeSeparation`	Describes eye separation. Depicted as B in Figure 4.
`EyeNoseSeparation`	Describes eye-nose separation. Depicted as C in Figure 4.
`MouseNoseSeparation`	Describes mouth-nose separation. Depicted as D in Figure 4.
`Mouthwidth`	Describes mouth-width separation. Depicted as E in Figure 4.
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1.

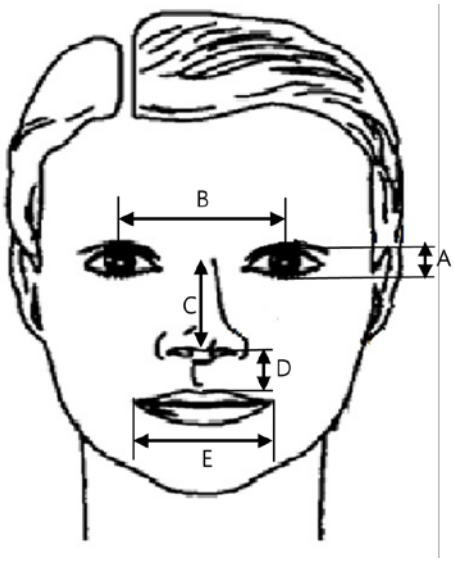


Figure 4 — Description of facial morphology parameters

6.46.5 Examples

This example shows a description of facial morphology sensing with the following semantics. The description of the facial morphology sensor has identifier of “FMS1” and the sensor references an actual sensor with ID of “FMSIC1”. The sensor shall be sensed at absTime ="100". The value of iris diameter, eye separation, eye-to-nose separation, mouth-to-nose separation and mouth width shall be 100, 100, 100, 100, and 100 (%), respectively.

```
 <iidl:SensedInfo xsi:type="siv:FacialMorphologySensorType" id="FMS1"
sensorIdRef="FMSIC1" activate="true">
       <iidl:TimeStamp xsi:type="mpegvct:AbsoluteTimeType" absTime="100" />
   <siv:IrisDiameter>
      100.0
   </siv:IrisDiameter>
   <siv:EyeSeparation>
      100.0
   </siv:EyeSeparation>
   <siv:EyeNoseSeparation>
      100.0
   </siv:EyeNoseSeparation>
   <siv:MouseNoseSeparation>
      100.0
   </siv:MouseNoseSeparation>
   <siv:MouseWidth>
      100.0
   </siv:MouseWidth>
</iidl:SensedInfo>
```

6.47 Facial expression characteristics sensor type

6.47.1 General

This subclause specifies a facial morphology sensor type which senses the parameters supporting the facial expression sensor type for intelligent cameras, and is related to the user-specific facial expression dynamic characteristics sensed from an intelligent camera. The applications of the sensor type may include avatar facial control and others.

6.47.2 Syntax

```
<!-- ################################################################ -->
<!--  Definition of facial expression characteristics sensor type -->
<!-- ################################################################ -->
<complexType name="FacialExpressionCharacteristicsSensorType">
    <complexContent>
        <extension base="iidl:SensedInfoBaseType">
            <sequence>
                <element name="FacialExpressionBasisRange"
type="siv:FacialExpressionBasisRangeType" minOccurs="0" maxOccurs="66"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<complexType name="FacialExpressionBasisRangeType">
    <attribute name="facialExpressionBasisID" type="mpeg7:termReferenceType"
use="optional"/>
    <attribute name="maxValueFacialExpressionBasis" type="float" use="optional"/>
    <attribute name="minValueFacialExpressionBasis" type="float" use="optional"/>
    <attribute name="neutralValueFacialExpressionBasis" type="float"
use="optional"/>
    <attribute name="facialExpressionBasisUnit" type="mpegvct:unitType"
use="optional"/>
</complexType>
```

6.47.3 Binary representation syntax

FacialExpressionCharacteristicsSensorType{	***Number of bits***	***Mnemonic***
FacialExpressionBasisRangeFlag	1	bslbf
SensedInfoBase		SensedInfoBaseType
if(FacialExpressionBasisRangeFlag) {		
NumOf FacialExpressionBasisRange	7	uimsbf
for(k=0; k< NumOfFacialExpressionBasisRange ; k++) {		
FacialExpressionBasisRange[k]		FacialExpressionBasisRangeType
}		
}		
}		

FacialExpressionBasisRangeType {		
facialExpressionBasisIDFlag	1	bslbf
maxValueFacialExpressionBasisFlag	1	bslbf
minValueFacialExpressionBasisFlag	1	bslbf
neutralValueFacialExpressionBasisFlag	1	bslbf
facialExpressionBasisUnitFlag	1	bslbf
if(facialExpressionBasisIDFlag) {		
facialExpressionBasisID		FacialExpressionBasisIDCSType
}		
if(maxValueFacialExpressionBasisFlag) {		
maxValueFacialExpressionBasis	32	fsbf
}		
if(minValueFacialExpressionBasisFlag) {		
minValueFacialExpressionBasis	32	fsbf
}		
if(neutralValueFacialExpressionBasisFlag) {		
neutralValueFacialExpressionBasis	32	fsbf
}		
if(facialExpressionBasisUnitFlag) {		
facialExpressionBasisUnit	8	bslbf
}		
}		

6.47.4 Semantics

Semantics of the `FacialExpressionCharacteristicsSensorType`:

Name	*Definition*
FacialExpressionCharacteristicsSensorType	Tool for describing a facial expression characteristics sensor sensed information.
FacialExpressionBasisRangeFlag	This field, which is only present in the binary representation, signals the presence of the activation attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
NumOfFacialExpressionBasisRange	This field, which is only present in the binary representation, indicates the number of facial expression basis range in this sensed information.
FacialExpressionBasisRange	Describes the range of each of facial expression basis parameters.
FacialExpressionBasisRangeType	Tool for describing a facial expression basis range.
facialExpressionBasisIDFlag	This field, which is only present in the binary representation, signals the presence of the activation attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
maxValueFacialExpressionBasisFlag	This field, which is only present in the binary representation, signals the presence of the activation attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
minValueFacialExpressionBasisFlag	This field, which is only present in the binary representation, signals the presence of the activation attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
neutralValueFacialExpressionBasisFlag	This field, which is only present in the binary representation, signals the presence of the activation attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
facialExpressionBasisUnitFlag	This field, which is only present in the binary representation, signals the presence of the activation attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
facialExpressionBasisID	Describes the identification of associated facial expression basis based as a reference to a classification scheme term provided by FacialExpressionBasisIDCS defined in B.9.
maxValueFacialExpressionBasis	Describes the maximum value of facial expression basis parameter.
minValueFacialExpressionBasis	Describes the minimum value of facial expression basis parameter.
neutralValueFacialExpressionBasis	Describes the value of facial expression basis parameter in neutral face.
facialExpressionBasisUnit	Describes the corresponding measurement units of the displacement amount described by a facial expression basis parameter.

6.47.5 Examples

This example shows a description of facial expression characteristics sensing with the following semantics. The description of the facial morphology sensor has identifier of “FECS1” and the sensor references an actual sensor with ID of “FECSIC1”. The maximum value of facial expression with the ID of “open_jaw” shall be 0(cm). The minimum value and the neutral value shall be -10 and 1.0 (cm). Also, the maximum value of facial expression with the ID of “raise_l_cornerlip” shall be 0(cm). The minimum value and the neutral value shall be -10 and 1.0 (cm).

```
<iidl:SensedInfo xsi:type="siv:FacialExpressionCharacteristicsSensorType"
id="FECS1" sensorIdRef="FECSIC1" activate="true">
   <siv:FacialExpressionBasisRange facialExpressionBasisID="open_jaw"
maxValueFacialExpressionBasis="0" minValueFacialExpressionBasis="-10"
neutralValueFacialExpressionBasis="1.0" facialExpressionBasisUnit="urn:mpeg:mpeg-
v:01-CI-UnitTypeCS-NS:cm"/>
   <siv:FacialExpressionBasisRange facialExpressionBasisID="raise_l_cornerlip"
maxValueFacialExpressionBasis="0" minValueFacialExpressionBasis="-10"
neutralValueFacialExpressionBasis="1.0" facialExpressionBasisUnit="urn:mpeg:mpeg-
v:01-CI-UnitTypeCS-NS:cm"/>
</iidl:SensedInfo>
```

6.48 Geomagnetic sensor type

6.48.1 General

The “geomagnetic sensor type” provides an absolute direction reference. This sensor is useful in a handheld device such as a mobile phone used in a navigation system or AR applications indicating the direction of the destination.

6.48.2 Syntax

```
<!--################################################ -->
<!--Definition of geomagnetic sensor type          -->
<!--################################################ -->
<complexType name="GeomagneticSensorType">
   <complexContent>
      <extension base="iidl:SensedInfoBaseType">
         <attribute name="azimuth" type="siv:azimuthType" use="optional"/>
         <attribute name="unit" type="mpegvct:unitType" use="optional"/>
      </extension>
   </complexContent>
</complexType>

<!--################################################ -->
<!--Definition of Azimuth type                      -->
<!--################################################ -->
<simpleType name="azimuthType">
   <restriction base="float">
      <minInclusive value="0"/>
      <maxInclusive value="360"/>
   </restriction>
</simpleType>
```

6.48.3 Binary representation syntax

GeomagneticSensorType{	***Number of bits***	***Mnemonic***
azimuthFlag	1	bslbf
unitFlag	1	bslbf
SensedInfoBaseType		SensedInfoBaseTypeType
if(azimuthFlag) {		
azimuth	32	fsbf
}		
if(unitFlag) {		
unit	8	bslbf
}		
}		

6.48.4 Semantics

Semantics of the `GeomagneticSensorType`:

Name	*Definition*
`GeomagneticSensorType`	Tool for describing sensed information with respect to a geomagnetic sensor.
`azimuth`	Describes the sensed value by the geomagnetic sensor in 0 to 360 degree. The value 0 means the “magnetic north pole” direction and 90 means the “east” with clockwise direction.
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The binary representation of the `UnitTypeCS` is also defined in ISO/IEC 23005-6:—, A.2.1.
`azimuthFlag`	This field, which is only present in the binary representation, signals the presence of azimuth attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.

Name	*Definition*
SensedInfoBaseType	Provides the topmost type of the base type hierarchy which each individual sensed information can inherit.

6.48.5 Examples

This example shows the description of a geomagnetic sensor sensing with the following semantics. The sensor has an ID of “GS001” and references “GSID001”. The sensor shall be activated and the value shall be azimuth="90" (degrees). The sensor shall be sensed at timestamp="50000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
   <iidl:SensedInfoList>
      <iidl:SensedInfo xsi:type="siv:GeomagneticSensorType" id="GS001"
         sensorIdRef="GSID001"          activate="true"          azimuth="90.0"
unit="urn:mpeg:mpeg-v:01-CI-UnitTypeCS-NS:degree" >
         <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
            pts="50000"/>
      </iidl:SensedInfo>
   </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.49 Proximity sensor

6.49.1 General

This subclause specifies a sensor type which senses a proximity from the sensor-specified point to a sensed object. The proximity sensor type does not specify any sensing methods such as ultrasonic, optical, and inductive technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The sensing properties of the sensor are specified in a proximity sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include robotics, security systems, and others.

6.49.2 Syntax

```
<!--##################################### -->
<!--Definition of Proximity Sensor type  -->
<!--##################################### -->
<complexType name="ProximitySensorType">
  <complexContent>
    <extension base="iidl:SensedInfoBaseType">
      <attribute name="value" type="float" use="optional"/>
      <attribute name="detected" type="boolean" use="optional"/>
      <attribute name="unit" type="mpegvct:unitType" use="optional"/>
    </extension>
  </complexContent>
</complexType>
```

6.49.3 Binary representation syntax

ProximitySensorType {	***Number of bits***	***Mnemonic***
valueFlag	1	bslbf
detectedFlag	1	bslbf
unitFlag	1	bslbf
SensedInfoBaseType	See above	SensedInfoBaseType
if (valueFlag == 1){		
value	32	fsbf
}		
if (detectedFlag == 1){		
detected	1	bslbf
}		
if (unitFlag == 1){		
unit	8	bslbf
}		
}		

6.49.4 Semantics

Semantics of the `ProximitySensorType`:

Name	*Definition*
`ProximitySensorType`	Tool for describing sensed information with respect to a proximity sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`value`	Describes the sensed distance value by the proximity with respect to the default unit or the unit defined in the unit attribute.
`detected`	When any movements are detected by the proximity sensor, the value becomes *true*; otherwise, the value becomes *false*.
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The default unit is centimetre.

Name	*Definition*
`valueFlag`	This field, which is only present in the binary representation, signals the presence of value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`detectedFlag`	This field, which is only present in the binary representation, signals the presence of detected attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.

6.49.5 Examples

This example shows the description of a proximity sensing with the following semantics. The description has identifier of “PS01” and the sensor references an actual sensor with ID of “PSID_01”. The sensor shall be activated. The distance value shall be 65 cm and the detect shall be true. The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:SensedInfo  xsi:type="siv:ProximitySensorType"  activate="true"  id="PS01"
sensorIdRef="PSID_01"  value="65"  detected="true"  unit="urn:mpeg:mpeg-v:01-CI-
UnitTypeCS-NS:Cm">
   <iidl:TimeStamp     xsi:type="mpegvct:ClockTickTimeType"     timeScale="100"
pts="60000"/>
</iidl:SensedInfo>
```

6.50 Switch sensor

6.50.1 General

This subclause specifies a sensor type which senses a switch. The switch sensor type does not specify any sensing methods such as inductive technologies like a magnetic sensor. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The sensing properties of the sensor are specified in a switch sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include robotics, security systems, and others.

6.50.2 Syntax

```
<!--################################## -->
<!--Definition of Switch Sensor type  -->
<!--################################## -->
<complexType name="SwitchSensorType">
  <complexContent>
    <extension base="iidl:SensedInfoBaseType">
      <attribute name="adjoin" type="boolean" use="optional"/>
      <attribute name="value" type="float" use="optional"/>
      <attribute name="unit" type="mpegvct:unitType" use="optional"/>
    </extension>
  </complexContent>
</complexType>
```

6.50.3 Binary representation syntax

SwitchSensorType {	***Number of bits***	***Mnemonic***
adjoinFlag	1	bslbf
valueFlag	1	bslbf
unitFlag	1	bslbf
SensedInfoBaseType	See above	SensedInfoBaseType
if (adjoinFlag == 1){		
adjoin	1	bslbf
}		
if (valueFlag == 1){		
value	32	fsbf
}		
if (unitFlag == 1){		
unit	8	bslbf
}		
}		

6.50.4 Semantics

Semantics of the `SwitchSensorType`:

Name	*Definition*
`SwitchSensorType`	Tool for describing sensed information with respect to a switch sensor.
`TimeStamp`	Describes the time that the information is acquired (sensed).
`adjoin`	If two parts of a switch sensor are joined together, *adjoin* becomes true. Otherwise, *adjoin* becomes false.
`value`	Describes the sensed value by the switch sensor with respect to the default unit or the unit defined in the unit attribute.
`unit`	Specifies the unit of the sensed value, if a unit other than the default unit is used, as a reference to a classification scheme term provided by `UnitTypeCS` defined in ISO/IEC 23005-6:—, A.2.1. The default unit is volt.
`adjoinFlag`	This field, which is only present in the binary representation, signals the presence of adjoin attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.

Name	*Definition*
`valueFlag`	This field, which is only present in the binary representation, signals the presence of value attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`unitFlag`	This field, which is only present in the binary representation, signals the presence of unit attribute. A value of “1” means the user-defined unit shall be used and “0” means the user-defined unit shall not be used.

6.50.5 Examples

This example shows the description of a switch sensing with the following semantics. The description has identifier of “SS01” and the sensor references an actual sensor with ID of “SSID_01”. The sensor shall be activated and the adjoin shall be true. The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:SensedInfo   xsi:type="siv:SwitchSensorType"   activate="true"   id="SS01"
sensorIdRef="SSID_01" adjoin="true">
   <iidl:TimeStamp     xsi:type="mpegvct:ClockTickTimeType"     timeScale="100"
pts="60000"/>
</iidl:SensedInfo>
```

6.51 Camera sensor type

6.51.1 General

This subclause specifies a basic sensor type which senses based on a camera. Various types of cameras, such as infrared cameras or spectrum cameras can be specified using this type of sensor.

6.51.2 Syntax

```
<!-- ################################################ -->
<!--  Camera Sensor Type                              -->
<!-- ################################################ -->
<complexType name="CameraSensorType">
  <complexContent>
    <extension base="iidl:SensedInfoBaseType">
      <sequence>
        <element name="CameraOrientation" type="siv:OrientationSensorType"
minOccurs="0"/>
        <element name="CameraLocation" type="siv:GlobalPositionSensorType"
minOccurs="0"/>
        <element name="CameraAltitude" type="siv:AltitudeSensorType"
minOccurs="0"/>
      </sequence>
      <attribute name="focalLength" type="float" use="optional"/>
      <attribute name="aperture" type="float" use="optional"/>
      <attribute name="shutterSpeed" type="float" use="optional"/>
      <attribute name="filter" type="mpeg7:termReferenceType" use="optional"/>
    </extension>
  </complexContent>
</complexType>
```

6.51.3 Binary representation syntax

CameraSensorType {	*Number of bits*	*Mnemonic*
CameraOrientationFlag	1	bslbf
CameraLocationFlag	1	bslbf
CameraAltitudeFlag	1	bslbf
focalLengthFlag	1	bslbf
apertureFlag	1	bslbf
shutterSpeedFlag	1	bslbf
filterFlag	1	bslbf
SensedInfoBaseType	See above	SensedInfoBaseType
if (CameraOrientationFlag){		
CameraOrientation	See above	OrientationSensorType
}		
if (CameraLocationFlag){		
CameraLocation	See above	GlobalPositionSensorType
}		
if (CameraAltitudeFlag){		
CameraAltitude	See above	AltitudeSensorType
}		
if (focalLengthFlag){		
focalLength	32	fsbf
}		
if (apertureFlag){		
Aperture	32	fsbf
}		
if (shutterSpeedFlag){		
shutterSpeed	32	fsbf
}		

if (filterFlag){		
Filter	4	bslbf
}		
}		

6.51.4 Semantics

Semantics of the `CameraSensorType`:

Name	*Definition*
`CameraSensorType`	Tool for describing sensed information with respect to a camera sensor.
`CameraLocation`	Describes the location of a camera using the structure defined by `GlobalPositionSensorType`.
`CameraAltitude`	Describes the altitude of a camera using the structure defined by `AltitudeSensorType`.
`CameraOrientation`	Describes the orientation of a camera using the structure defined by `OrientationSensorType`.
`focalLength`	Describes the distance between the lens and the image sensor when the subject is in focus, in terms of millimetres (mm).
`aperture`	Describes the diameter of the lens opening. It is expressed as F-stop, e.g. F2.8. It may also be expressed as f-number notation such as f/2.8.
`shutterSpeed`	Describes the time that the shutter remains open when taking a photograph in terms of seconds (sec).
`filter`	Describes kinds of camera filters as a reference to a classification scheme term that shall be using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. The CS that may be used for this purpose is the `CameraFilterTypeCS` defined in ISO/IEC 23005-6:—, A.2.14.
`CameraOrientationFlag`	This field, which is only present in the binary representation, signals if camera orientation sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.
`CameraLocationFlag`	This field, which is only present in the binary representation, signals if camera location sensed information is available. A value of “1” indicates that the sensed information shall be included and “0” indicates that the sensed information shall not be included.
`CameraAltitudeFlag`	This field, which is only present in the binary representation, signals if camera altitude sensed information is available. A value of “1” indicates that the sensed information shall be included and “0”

Name	*Definition*
	indicates that the sensed information shall not be included.
`focalLengthFlag`	This field, which is only present in the binary representation, signals the presence of focal length attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`apertureFlag`	This field, which is only present in the binary representation, signals the presence of aperture attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`shutterSpeedFlag`	This field, which is only present in the binary representation, signals the presence of shutter speed attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.
`filterFlag`	This field, which is only present in the binary representation, signals the presence of filter attribute. A value of “1” means the attribute shall be used and “0” means the attribute shall not be used.

6.51.5 Examples

This example shows the description of a camera sensing with the following semantics. The description has identifier of “CSID001”. The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second. The focal length of the sensor is 50 (mm) and the aperture is F2.8 and the shutter speed of the sensor is 1/250 (sec) and uses an UV filter. The location information of the camera sensor has 37.23 N of the latitude and 131.23 E of the longitude. The orientation information of the camera sensor has Ox="2.0" (radian), Oy="-0.5" (radian), and Oz="1.0" (radian).

```
<iidl:InteractionInfo>
  <iidl:SensedInfoList>
    <iidl:SensedInfo xsi:type="siv:CameraSensorType" id="CSID001" activate="true"
focalLength="50" aperture="2.8" shutterSpeed="0.004" filter="urn:mpeg:mpeg-v:01-
SI-CameraFilterTypeCS-NS:UV">
      <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
      <siv:CameraOrientation xsi:type="siv:OrientationSensorType" unit="radian">
        <siv:Orientation>
          <mpegvct:X>2.0</mpegvct:X>
          <mpegvct:Y>-0.5</mpegvct:Y>
          <mpegvct:Z>1.0</mpegvct:Z>
        </siv:Orientation>
      </siv:CameraOrientation>
      <siv:CameraLocation xsi:type="siv:GlobalPositionSensorType"
longitude="131.23" latitude="37.23"/>
    </iidl:SensedInfo>
  </iidl:SensedInfoList>
</iidl:InteractionInfo>
```

6.52 Spectrum camera sensor type

6.52.1 General

This subclause specifies a sensor type which senses light spectrum data. The spectrum camera sensor type does not specify any sensing methods such as ultrasonic, optical, and inductive technologies. Therefore, any measurement specific to the particular sensing technologies is not the scope of the sensor type. The sensing properties of the sensor are specified in the camera sensor capability in ISO/IEC 23005-2. The applications of the sensor type may include robotics, user interface, haptics, and telepresence.

6.52.2 Syntax

```
<!-- ################################################ -->
<!--  Spectrum Camera Sensor Type                     -->
<!-- ################################################ -->
<complexType name="SpectrumCameraSensorType">
  <complexContent>
    <extension base="siv:CameraSensorType">
      <sequence>
        <element name="Spectra" type="mpeg7:DoubleMatrixType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

6.52.3 Binary representation syntax

SpectrumCameraSensorType {	***Number of bits***	***Mnemonic***
CameraSensorType		CameraSensorType
heightSize		vluimsbf5
widthSize		vluimsbf5
dimensionSize	9	uimsbf
for(i=0;i<heightSize;i++){		
for(k=0;k<widthSize;k++){		
for(m=0;m<dimensionSize;m++){		
Spectra[i][k][m]	32	fsfb
}		
}		
}		
}		

6.52.4 Semantics

Semantics of the SpectrumCameraSensorType:

Name	*Definition*
SpectrumCameraSensorType	Tool for describing sensed information with respect to a spectrum camera sensor.
Spectra	Describes the sensed spectrum values by the spectrum camera sensor. The spectrum data is represented by an image size (i.e., a height and a width) and a spectrum dimension. The maximum size of the spectrum dimension is 301 since this covers a visible spectrum range between 400 nm and 700 nm for every 1 nm. The spectrum dimension can be subsampled linearly. For example, if the sampling rate is every 10 nm, the spectrum dimension becomes 31. Henceforth, the spectrum data can be represented by a three dimensional matrix that the size is *height x width x spectrum dimension*. A spectrum data (i.e., one color) can be represented by a three dimension matrix with size of *1 x 1 x spectrum dimension*.
heightSize	This field, which is only present in the binary representation, describes a height of an image.
widthSize	This field, which is only present in the binary representation, describes a width of an image.
dimensionSize	This field, which is only present in the binary representation, describes a dimension of spectrum data.

6.52.5 Examples

This example shows the description of a spectrum camera sensing with the following semantics. The description has identifier of “SCSID01”. The sensed spectrum values is matrix. The size of the spectra matrix is 1x1x31. The sensor shall be sensed at timestamp="60000" where there are 100 clock ticks per second.

```
<iidl:InteractionInfo>
  <iidl:SensedInfoList>
    <iidl:SensedInfo xsi:type="siv:SpectrumCameraSensorType" id="SCSID001"
activate="true">
      <iidl:TimeStamp xsi:type="mpegvct:ClockTickTimeType" timeScale="100"
pts="60000"/>
      <siv:Spectra mpeg7:dim="1 1 31">
        0.107219898 0.101596882 0.0992108 0.103684803
        0.116491943 0.130614778 0.140921731 0.149217541
        0.154775693 0.161859824 0.169272991 0.174340486
        0.173159037 0.163239738 0.157651944 0.160515332
        0.166958233 0.167805282 0.16460807 0.204031188
        0.256942818 0.295577819 0.32206277 0.339098991
        0.352361207 0.363327769 0.372403743 0.381562788
        0.391658505 0.401602906 0.409962343
      </siv:Spectra>
    </iidl:SensedInfo>
  </iidl:SensedInfoList>
</iidl:InteractionInfo>
```