INTERNATIONAL STANDARD

ISO/IEC 14496-11

First edition
2005-12-15

AMENDMENT 7
2010-12-15

Information technology — Coding of audio-visual objects —

Part 11: Scene description and application engine

AMENDMENT 7: ExtendedCore2D profile

Technologies de l'information — Codage des objets audiovisuels —
Partie 11: Description de scène et moteur d'application
AMENDEMENT 7: Profil ExtendedCore2D

Reference number
ISO/IEC 14496-11:2005/Amd.7:2010(E)

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

Published in Switzerland

Foreword

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

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

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

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

Amendment 7 to ISO/IEC 14496-11:2005 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

Information technology — Coding of audio-visual objects —

Part 11:
Scene description and application engine

AMENDMENT 7: ExtendedCore2D profile

In 7.2, add the following subclauses in node alphabetical order:

7.2.2.ZZ CacheTexture

7.2.2.ZZ.1 Node interface

CacheTexture **{**

Field	SFInt32	**objectTypeIndication**	0
Field	SFString	**decoderSpecificInfo**	NULL
Field	SFString	**image**	NULL
Field	SFString	**cacheURL**	NULL
Field	MFURL	**cacheOD**	[]
Field	SFInt32	**expirationDate**	0
Field	SFBool	**repeatS**	TRUE
Field	SFBool	**repeatT**	TRUE

}

NOTE For the binary encoding of this node see node coding tables in electronic attachment.

7.2.2.ZZ.2 Functionality and semantics

The CacheTexture allows carriage of visual data embedded inside the BIFS stream rather than using the OD framework. The **objectTypeIndication** field identifies the media type of the visual data. The compressed data is carried in the **image** field, as a single access unit. If a decoder configuration is needed, it may be carried in the **decoderSpecificInfo** field. The node can be used as a texture object in an **Appearance** node. The node can also be used in as a child node of a 2D or 3D container when it is only used for image caching. Additionally, the CacheTexture node allows for caching the embedded image by specifying a **cacheURL** name to be referred to by other nodes in the scene, as well as an **expirationDate** indicating the time in seconds the terminal should keep the data in its cache. If **expirationDate** is 0, the data shall not be cached. If **expirationDate** is strictly negative, the data should be cached for as long as possible. In any case, whether the data is cached or not is implementation specific.

The **cacheOD** field identifies an existing OD in the scene to be cached with the given **cacheName** and **expirationDate**. If **cacheOD** is set, **image, decoderSpecificInfo** and **objectTypeIndication** shall be ignored. Results are undefined if the OD indicated by the **cacheOD** is not a still image object such as JPEG or PNG.

The scoping of the CacheTexture node shall be done at the service level (same broadcast channel or same service URL of the initial scene). Sub-scenes opened through inline nodes are part of the same caching scope as the parent scene.

Example of cache usage

```
Shape {
 appearance Appearance {
  texture ImageTexture {
   url "some_cache_url_name"
  }
 }
}
...
CacheTexture {
 objectTypeIndication 0x6D
 image ...
 cacheURL "some_cache_url_name"
 expirationDate 3600   //one hour caching
}
....
```

7.2.2.ZZ EnvironmentTest

7.2.2.ZZ.1 Node interface

EnvironmentTest {			
eventIn	SFBool	**evaluate**	
exposedField	SFBool	**enabled**	TRUE
exposedField	SFInt32	**parameter**	0
exposedField	SFString	**compareValue**	NULL
exposedField	SFBool	**evaluateOnChange**	TRUE
eventOut	SFBool	**valueLarger**	
eventOut	SFBool	**valueEqual**	
eventOut	SFBool	**valueSmaller**	
eventOut	SFString	**parameterValue**	
}			

NOTE For the binary encoding of this node see node coding tables in electronic attachment.

7.2.2.ZZ.2 Functionality and semantics

The EnvironmentTest node enables testing a **parameter** of the terminal environment, possibly comparing their values with the **compareValue**. The evaluation of the parameter triggers different eventOuts depending on the type of the parameter:

- If the **parameter** type is Boolean, the evaluation triggers a **valueEqual** eventOut, and the **compareValue** field is ignored.
- If the **parameter** type is a number and the **compareValue** represents a number, the two values are compared and the following eventOuts are generated:
  - **valueEqual** if **parameter** and **compareValue** are equal
  - **valueLarger** if **compareValue** is strictly larger than **parameter**
  - **valueSmaller** if **compareValue** is strictly less then than **parameter**

The supported parameter types are defined in Table AMD7.1.

In any case, the **parameterValue** eventOut is triggered after evaluation.

If **evaluateOnChange** is set to FALSE, the node only evaluates upon receiving the **evaluate** eventIn; otherwise, the node evaluates on any change of **parameter** or **compareValue**.

The node evaluates and triggers events only when its **enabled** field is true.

Table AMD7.1 — Environmental parameters

Value	Definition	Type
0	Display region Aspect Ratio (larger dimension divided by smaller dimension, regardless of screen orientation)	Float
1	Portrait mode of the display region (TRUE if width<height)	Boolean
2	Display region width in pixels	Integer
3	Display region height in pixels	Integer
4	Horizontal DPI	Integer
5	Vertical DPI	Integer
6	Automotive Situation (terminal user drives a moving vehicle)	Boolean
7	User is Visually Challenged	Boolean
8	Touch Screen present on terminal	Boolean
9	Navigation Keypad present on terminal	Boolean
0x00000007-0xEFFFFFFF	ISO Reserved	
0xF0000000-0xFFFFFFFF	User Reserved	

The display region is the area onto which the BIFS content is rendered. This region may be the entire screen, some part of the screen or an off-screen memory region.

7.2.2.ZZ KeyNavigator

7.2.2.ZZ.1 Node interface


```
KeyNavigator {
    eventIn      SFBool     setFocus
    exposedField SFNode     sensor      NULL
    exposedField SFNode     left        NULL
    exposedField SFNode     right       NULL
    exposedField SFNode     up          NULL
    exposedField SFNode     down        NULL
    exposedField SFNode     select      NULL
    exposedField SFNode     quit        NULL
    exposedField SFFloat    step        0
    eventOut     SFBool     focusSet
}
```

NOTE For the binary encoding of this node see node coding tables in electronic attachment.

7.2.2.ZZ.2 Functionality and semantics

The KeyNavigator node enables simple, pre-defined 2D navigation in the scene. Each KeyNavigator is associated with an existing sensor node (TouchSensor, PlaneSensor2D...) through the **sensor** field. The first KeyNavigator node found in the scene is used to determine the initial focusable object. If the attached **sensor** node is NULL or is disabled, the focus is not attached to any visual part of the scene. Focus can be changed by using the navigation pad of the terminal as follows:

- Pressing the left key will move focus to the **left** KeyNavigator node
- Pressing the right key will move focus to the **right** KeyNavigator node
- Pressing the up key will move focus to the **up** KeyNavigator node
- Pressing the down key will move focus to the **down** KeyNavigator node
- Pressing the validation key (OK, Enter, Select...) will move focus to the **select** KeyNavigator node

- Pressing the escape key (escape, back, end call...) will move focus to the **quit** KeyNavigator node
- At any time, a KeyNavigator can be focused by sending the node a **setFocus** eventIn.

Whenever a KeyNavigator node receives the focus, it triggers a **focusSet** = **true** eventOut. When the KeyNavigator node loses the focus, it triggers a **focusSet** = **false** eventOut.

A pointing device sensor is controlled through the keypad as indicated in Table AMD7.2, with directions given in the local coordinate system of the sensor node. Processing of keystrokes by the KeyNavigator node is inhibited while the sensor is active.

NOTE The attribution of keys for activation and deactivation of the associated sensor is implementation specific.

The **step** field indicates the horizontal or vertical mouse displacement to simulate when using directional keys, and indicates the displacement in the sensor local coordinate system. If the value of **step** is less than or equal to 0, the mouse displacement is implementation specific.

Table AMD7.2 — Mapping of keys for BIFS sensor nodes

Sensor Type	focusIn	focusOut	LEFT	RIGHT	UP	DOWN
TouchSensor	**isOver=true**	**isOver=false**	N/A	N/A	N/A	N/A
PlaneSensor2D	N/A	N/A	Left move	Right move	up move	Down move
DiscSensor	N/A	N/A	Counter clockwise move	Clockwise move	N/A	N/A
PlaneSensor	N/A	N/A	Left move	Right move	up move	Down move
CylinderSensor	N/A	N/A	Counter clockwise move	Clockwise move	N/A	N/A
SphereSensor	N/A	N/A	I/S	I/S	I/S	I/S

N/A: Non-Applicable - I/S: Implementation Specific.

NOTE 1 Authors should be aware that when activating a TouchSensor node, the focus might automatically be moved to the **select** field of the associated key navigator.

NOTE 2 A terminal handling both key navigation and pointing device should automatically manage the active KeyNavigator node. When the pointing device moves over an active sensor associated with a KeyNavigator node, this KeyNavigator node should become the current focused KeyNavigator node.

NOTE 3 A terminal should trigger the key events on key down and handle key repeat, when the key is not released for some period of time.

7.2.2.ZZ Storage

7.2.2.ZZ.1 Node interface

```
Storage {
    eventIn         SFBool       forceSave
    eventIn         SFBool       forceRestore
    exposedField    SFBool       auto            TRUE
    Field           SFInt32      expireAfter     0
    Field           SFString     name            NULL
    Field           MFAttrRef    storageList     []
}
```

NOTE For the binary encoding of this node see node coding tables in electronic attachment.

7.2.2.ZZ.2 Functionality and semantics

The Storage node enables saving and restoring any field values in a scene to a private storage zone of the terminal. The **name** parameter allows defining several storage zones in a single scene. The terminal should keep the stored value for the number of seconds indicated in the **expireAfter** field, or for an undetermined period of time, up to the implementation, if this value is less than or equal to zero. The scoping of the Storage

node shall be done at the service level (e.g., same broadcast channel or same service URL of the initial scene). Sub-scenes opened through inline nodes are part of the same storage scope as the parent scene. In a same service, there shall not be more than one storage node with a given name field.

The set of node fields to be saved or restored is specified in the **storageList** field. Conceptually, saving node fields is equivalent to remembering the number of fields, their types and their values, and restoring is the opposite operation. This allows saving and restoring of node fields independently from node IDs which may vary across different scenes. The target field shall be an SF or an MF field with an underlying SF type equal to SFBool, SFInt32, SFFloat, SFTime, SFString, SFVec3f, SFVec2f, SFColor, and SFRotation. For complexity reasons, storing and restoring of SFNode/MFNode, SFImage/MFImage and SFCommandBuffer fields are not allowed.

Results are undefined if the target field types do not match between the save and the restore operations.

If **auto** is TRUE, then the terminal restores the information after decoding of the Storage object, and saves the information upon exiting the scene. If **auto** is FALSE, the terminal saves the node field values when the eventIn **forceSave** is triggered, and restores them when the eventIn **forceRestore** is triggered.

In 7.2.2.74.2, add the following:

If the major mode in the justify field of the layout is "JUSTIFY", then the layout of children starts at the "BEGIN" edge of the layout and ends at the "END" edge of the layout with space adjustments if needed. "BEGIN" and "END" are defined in the FontStyle node semantics. If wrap is false and the line is larger than the layout frame, the terminal may alter the text to indicate it has been truncated.

At the end of 7.2.2.135.2 before the final example, add the following:

If the eventIn is inMFString then the outSFString eventOut shall be created by using the first element of inMFString input if the “Sum” field is set to “false”, or the concatenation of all strings in the inMFString input if the “Sum” field is set to “true”. In this special case, FactorX and OffsetX fields are ignored.

In 7.9.2.3, replace Table 41 with the following:

Scene Graph Tools	Scene Graph Profiles									
	Basic 2D	Simple 2D	Core 2D	Extended Core 2D	Main 2D	Advanced 2D	Complete 2D	Audio	3D Audio	Complete
AcousticScene									X	
AdvancedAudioBuffer										
AnimationStream						X	X		X	X
Anchor			X	X	X	X	X		X	X
ApplicationWindow										
AudioBuffer						X	X	X	X	X
AudioDelay							X	X	X	X
AudioFX							X	X	X	X
AudioMix							X	X	X	X
AudioSwitch						X	X	X	X	X
Billboard									X	X
BitWrapper										
CacheTexture				X						
Clipper2D										
ColorInterpolator			X	X	X	X	X			X
ColorTransform										
Collision										X
CompositeTexture2D				X			X			X
CompositeTexture3D										X
Conditional			X	X	X	X	X		X	X
CoordinateInterpolator2D			X	X	X	X	X			X
CoordinateInterpolator									X	X
CoordinateInterpolator4D										
CylinderSensor										X
DirectiveSound									X	
DiscSensor					X	X	X			X
EnvironmentTest				X						
Form							X			X
Group						X	X	X	X	X
Inline			X	X	X	X	X		X	X
InputSensor			X	X	X	X				
KeyNavigator				X						
Layer2D				X	X	X	X			X
Layer3D										X
Layout				X			X			X
ListeningPoint							X	X	X	X
LOD									X	X
MediaBuffer						X				
MediaControl			X	X	X	X				
MediaSensor			X	X	X	X				
NavigationInfo										X
NormalInterpolator										X
OrderedGroup	X	X	X	X	X	X	X			X

Scene Graph Tools	Scene Graph Profiles									
	Basic 2D	Simple 2D	Core 2D	Extended Core 2D	Main 2D	Advanced 2D	Complete 2D	Audio	3D Audio	Complete
OrientationInterpolator									X	X
PathLayout										
PerceptualParameters									X	
PlaneSensor2D					X	X	X			X
PlaneSensor										X
PositionAnimator										
PositionAnimator2D										
PositionInterpolator									X	X
PositionInterpolator2D			X	X	X	X	X			X
PositionInterpolator4D										
ProximitySensor									X	X
ProximitySensor2D					X	X	X			X
QuantizationParameter			X	X	?	X	X		X	X
ScalarAnimator										
ScalarInterpolator			X	X	X	X	X			X
Script						X			X	X
ServerCommand			X	X	X	X				
Sound								X	X	X
Sound2D	X	X	X	X	X	X	X		X	X
SphereSensor										X
Storage				X						
Switch			X	X	X	X	X		X	X
TemporalTransform					X	X				
TemporalGroup					X	X				
TermCap						X	X		X	X
TimeSensor			X	X	X	X	X		X	X
TouchSensor			X	X	X	X	X		X	X
Transform									X	X
Transform2D		X	X	X	X	X	X			X
Transform3Daudio										
TransformMatrix2D				X						
Valuator			X	X	X	X	X		X	X
Viewpoint									X	X
Viewport				X						
VisibilitySensor									X	X
WorldInfo				X		X	X		X	X
Node Update			X	X	X	X	X		X	X
Route Update			X	X	X	X	X		X	X
Scene Update		X	X	X	X	X	X	X	X	X
ROUTE			X	X	X	X	X		X	X
PROTO				X		X				
Extended Updates				X						
Interpolator Compression										
PredictiveMF coding						X				