INTERNATIONAL STANDARD

**ISO/IEC 23007-3**

First edition
2011-07-15

# Information technology — Rich media user interfaces

## Part 3:
## Conformance and reference software

*Technologies de l'information — Interfaces d'utilisateur au support riche —*

*Partie 3: Conformité et logiciel de référence*

Reference number
ISO/IEC 23007-3:2011(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

## 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. 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.

ISO/IEC 23007-3 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

ISO/IEC 23007 consists of the following parts, under the general title *Information technology — Rich media user interfaces*:

— *Part 1: Widgets*

— *Part 2: Advanced user interaction (AUI) interfaces*

— *Part 3: Conformance and reference software*

# Introduction

User interfaces represent a crucial feature for many consumer devices and services. User interfaces have recently evolved to support more media types including audio, video, 2D or 3D graphics and have become rich media user interfaces. User interfaces are also moving towards a collection of small dedicated applications, called Widgets.

Additionally, more and more devices are capable of displaying rich media user interfaces, from desktop computers, to mobile devices, to home appliances including TV sets. In this heterogeneous environment, users expect a homogeneous, unified experience when interacting with their devices.

In ISO/IEC 23007, Widgets can be processed by entities running on different devices, called Widget Managers, in charge of processing and managing the life cycle of the widgets supporting communications with other entities locally or remotely deployed and enabling widget mobility across devices.

The objective of ISO/IEC 23007 is to provide normative interfaces between Widgets and Widget Managers, to allow Widgets from different service providers to run, communicate and be transferred within a unique framework.

ISO/IEC 23007, rich media user interfaces is also known as MPEG-U. ISO/IEC 23007-1 addresses the normative aspects of the MPEG-U Widgets. In particular, it specifies widget packaging formats, aspects for widget communications with external entities and for widget mobility. It also contains a technical annex describing a list of use cases and examples to address such use cases. ISO/IEC 23007-2 addresses advanced user interaction (AUI) interfaces. This part of ISO/IEC 23007 addresses reference software and conformance aspects.

# Information technology — Rich media user interfaces —

# Part 3: Conformance and reference software

## 1 Scope

This part of ISO/IEC 23007 specifies how to test whether data and decoders meet requirements specified by ISO/IEC 23007-1. It also describes the procedures for testing the conformance of widgets and widget managers to the requirements defined in ISO/IEC 23007-1.

In this part of ISO/IEC 23007, widget generators are not addressed specifically. A generator can be said to be an ISO/IEC 23007 generator if it generates widgets compliant with the syntactic and semantic requirements specified in ISO/IEC 23007-1.

## 2 Normative references

The following referenced documents are indispensable for the application 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 23007-1, *Information technology — Rich media user interfaces — Part 1: Widgets*

## 3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 23007-1 apply.

## 4 Conformance of Widgets

### 4.1 Introduction

Widgets use standard representations for describing manifests, W3C WPC packages, ISOFF packages and context information. This clause describes how to check conformance of all these representations.

### 4.2 Conformance of manifests

The first step for checking conformance of the MPEG-U XML manifest syntax is to validate the XML document against the relax NG for the configuration document (http://www.w3.org/TR/widgets/#relax-ng-for-the-configuration-document) provided by the W3C WPC specification. If this step fails, the widget manifest is considered a non-conformant MPEG-U manifest.

The second step is to check conformance according to the MPEG-U syntax. The electronic attachments of ISO/IEC 23007-1 provide an XML Schema for that purpose. A conformant W3C WPC configuration document is a valid MPEG-U manifest if the XML schema validation succeeds.

The next step is to verify that the scene constructs referenced in the manifest are present in the scene description pointed to by the manifest. This verification is specific to the scene description language used.

### 4.3 Conformance of packages

A widget package is conformant to ISO/IEC 23007 if it is conformant to the W3C WPC specification or if it is a conformant ISOFF-based widget package.

An ISOFF-based widget package is conformant to ISO/IEC 23007 if the following rules are validated:

- it is a compliant ISO/IEC 14496-12 file or a compliant ISO/IEC 21000-9 file;
- it uses the brand declared in ISO/IEC 23007-1 either in the list of compatible brand or in the major brand;
- it contains a conformant manifest as a primary item and stored in an XML box, with a internet MediaType and handler type as defined in ISO/IEC 23007-1;
- all resources declared in the manifest with IRI relative to the manifest are described in the item location box.

### 4.4 Conformance of context information

A widget context information document is conformant to ISO/IEC 23007 if it is a valid XML file according to the XML Schema provided as an electronic attachment to ISO/IEC 23007-1.

### 4.5 Conformance of the MPEG-U widget scripts

Scripts of MPEG-U widgets shall obey the constraints specified in Clause 9 of ISO/IEC 23007-1:2010, as well as constraints specified in the WAE specification.

## 5 Conformance of Widget Managers

Widget managers are responsible for the processing widgets, e.g. for providing MPEG-U communication capabilities to widgets and for managing context information. Normative behaviours of Widgets managers are demonstrated by the Reference Software.

The following behaviours are defined as non normative:

- Composition and presentation of the scene description language used in the widget;
- Presentation of available widgets to the user and use of a selection mechanism for activation of the full or simplified representation;
- Transitions between the full and simplified representation;
- Management of Communication privileges;
- Widget communication activation and deactivation.

For some of these non-normative behaviours, utility software is provided.

Conformant MPEG-U widget managers shall be W3C Widgets-compliant user agents, and obey constraints specified in ISO/IEC 23007-1 as well.

# 6 Conformance Sequences

## 6.1 Introduction

Electronic attachments are organized in the following fashion:

- directory mp4: widgets packaged in MP4 files (extension .mgt)
- directory wgts: widgets packaged in ZIP files (extension .wgt)
- directory widgets: unpackaged widgets

Widgets are implemented in SVG and ECMA-Script unless specified otherwise.

## 6.2 Sequences

### 6.2.1 animatedIcon

This widget has an active icon, animated with scripting.

Directory: widgets/animatedIcon

### 6.2.2 c1

This c1 widget and its component c1comp test many features of MPEG-U, including activation and deactivation of a component, binding of multiple interfaces, exchanging various kinds of messages… c1 displays success messages numbered from 000 to 023. One message appears twice as 0--. Any number missing in the sequence means there is a bug.

Directory: widgets/c1 & widgets/c1comp

### 6.2.3 c2

This c2 widget and its component c1comp test some features of MPEG-U, in particular the saving and restoring of a preference. c1 displays success messages numbered from 000. One message appears twice as 0--. Any number missing in the sequence means there is a bug.

Directory: widgets/c2 & widgets/c1comp

### 6.2.4 core

This core widget and its component comp test the predefined messages from the core interfaces specifically.

Directory widgets/core/main & widgets/core/comp

### 6.2.5 upnp_light_switch

Tests the connection with an UPNP service. Intended for use with the Intel Network Light.

Directory: widgets/upnp_light_switch

### 6.2.6 light_switch_pair

These two widgets test the connection of two widgets in local or remote mode. One acts as a switch and the other as a light bulb.

Directory: widgets/light_switch_pair/comwidin and widgets/light_switch_pair/comwidout

### 6.2.7 clock

This widget is a digital clock implemented in BIFS. It uses preferences and localization.

Directory: widgets/clock

### 6.2.8 dlnaplayercontrol and simpleplayercontrol

These two widgets control an UPnP AVRenderer, the first with scripting, and the second with a purely declarative scene and limited functionality.

Directory: widgets/dlnaplayercontrol and widgets/simpleplayercontrol

### 6.2.9 internet_radio and internet_radio_control

These two widgets implement a radio playing streamed audio from the Internet, and a very simple control panel for the radio. The radio widget uses preferences to reload its previous execution context.

Directory: widgets/internet_radio and widgets/internet_radio_control

### 6.2.10 widget_spy

This widget defines interfaces of 4 other widgets above and of GPAC renderers and shows the binding status of these interfaces, thus implementing a test for the presence/absence of the other widgets.

Directory: widgets/widget_spy

## 6.3 Coverage of MPEG-U features

| Feature | Sequence | Positive test |
|---|---|---|
| ZIP packaging | mp4/animatedIcon.wgt | Y |
| ZIP packaged widget without manifest | special/z1.wgt | N |
| ISO-FF packaging | wgts/animatedIcon.mgt | Y |
| unpackaged | widgets/animatedIcon | Y |
| unpackaged widget without manifest | special/z1/ | N |
| Communication: input messages | widgets/light_switch_pair | Y |
| Communication: output messages | widgets/light_switch_pair | Y |
| Communication: reply messages | widgets/light_switch_pair | Y |
| Multiple instances | widgets/light_switch_pair/comwidin | Y |
| Localization | widgets/clock | Y |
| Widgets with invalid manifest | W3C invalid widgets from the W3C sequences | N |
| invalid Widgets in ZIP packages | W3C invalid widgets from the W3C sequences | N |

| | | |
|---|---|---|
| valid widgets (display of metadata of the widget) | any valid widget | Y |
| widget Life Cycle Events | widgets/core/comp | Y |
| matching interfaces (Testing dynamic matching of interfaces by adding, removing, adding one of the widget in a pair of compatible widgets) | widgets/light_switch_pair | Y |
| bindAction | widgets/widget_spy | Y |
| unbindAction | widgets/widget_spy | Y |
| multiple bindings true | widgets/upnp_light_switch | Y |
| multiple bindings false | widgets/light_switch/comwidin | Y |
| API MPEGWidget.onInterfaceBind | widgets/MPEGWidgetInterfaceTesting | Y |
| API MPEGWidget.onInterfaceUnbind | widgets/MPEGWidgetInterfaceTesting | Y |
| API MPEGWidget.onActivation | widgets/MPEGWidgetInterfaceTesting | Y |
| API MPEGWidget.activateComponentWidget | widgets/MPEGWidgetInterfaceTesting | Y |
| API MPEGWidget.deactivateComponentWidget | widgets/MPEGWidgetInterfaceTesting | Y |
| API MPEGWidget.getInterfaceHandlersByType | widgets/c1 | Y |
| API InterfaceHandler.bound | widgets/MPEGWidgetInterfaceTesting | Y |
| API InterfaceHandler.type | widgets/MPEGWidgetInterfaceTesting | Y |
| API InterfaceHandler.invoke | widgets/c1 | Y |
| API InterfaceHandler.msgHandlerFactory | widgets/c1 | Y |
| API InterfaceHandler.invokeReply | widgets/c1comp | Y |
| API MsgHandler.msgName | widgets/c1comp | Y |
| API MsgHandler.interfaceHandler | widgets/c1comp | Y |
| discardable widget | widgets/c1comp | Y |
| widget manifest with multiple content elements | widgets/animatedIcon | Y |
| contextConfiguration | widgets/internet_radio | Y |
| overriden preferences | widgets/internet_radio | Y |
| serviceProvider | widgets/light_switch_pair/comwidin | Y |
| component | widgets/c1 | Y |
| component with connectTo | widgets/c1 | Y |
| required Interfaces | widgets/c1 | Y |

| | | |
|---|---|---|
| inputAction with a node | widgets/light_switch_pair/comwidout | Y |
| inputAction with a function | widgets/c1 | Y |
| outputTrigger with a node | widgets/light_switch_pair/comwidout | Y |
| outputTrigger with an event | widgets/core/comp | Y |
| default input | widgets/simpleplayercontrol | Y |
| input with setAttribute | widgets/light_switch_pair/comwidin | Y |
| input without setAttribute | widgets/internet_radio | Y |
| default output | widgets/core/main | Y |
| output with attributeModified | widgets/simpleplayercontrol | Y |
| output without attributeModified | widgets/c1 | Y |
| widget with multiple interfaces | widgets/c1 | Y |
| widget with multiple output messages | widgets/c1 | Y |
| widget with multiple input messages | widgets/internet_radio | Y |
| widgets with multiple messageIn elements with the same name | widgets/light_and_switch_pair/comwidin | Y |
| widgets with multiple messageOut elements with the same name | widgets/simpleplayercontrol | Y |
| widget with components using src with URL | widgets/c1 | Y |
| activateTrigger | widgets/c1 | Y |
| deactivateTrigger | widgets/c1 | Y |
| activateFailureAction | widgets/c1 | Y |
| requiredInterface | widgets/c2b | Y |
| Predefined communications: messageIn setSize | widgets/core/main | Y |
| Predefined communications: messageIn show | widgets/core/main | Y |
| Predefined communications: messageIn hide | widgets/core/main | Y |
| Predefined communications: messageIn activate | widgets/core/main | Y |
| Predefined communications: messageIn deactivate | widgets/core/main | Y |
| Predefined communications: messageOut setSize | widgets/core/main | Y |
| Predefined communications: messageOut show | widgets/core/main | Y |
| Predefined communications: messageOut hide | widgets/core/main | Y |

| | | |
|---|---|---|
| Predefined communications: messageOut getAttention | widgets/core/main | Y |
| Predefined communications: messageOut requestDeactivate | widgets/core/main | Y |
| Predefined communications: messageOut showNotification | widgets/core/main | Y |
| Predefined communications: messageOut placeComponent | widgets/c2 | Y |
| widget with the same simplified and full representation | widgets/clock2 | Y |
| widget with one interface with required=‘true’ | widgets/c1comp | Y |
| ISO-FF widget with ‘mwgt’ as major brand | special/animatedIcon1.mgt | Y |
| ISO-FF widget with ‘mwgt’ as a compatible brand | special/animatedIcon2.mgt | Y |
| ISO-FF widget with additional resources stored as items in the file | any in mp4 | Y |
| widget to test support for the invoke method | widgets/c2 | Y |
| widget to test support for the MsgHandler callback function | widgets/dlnaplayercontrol | Y |
| widget with a script function in the mw:inputAction attribute and an invokeReply | widgets/c1comp | Y |
| widget with scene construct using XML namespace prefixes | widgets/internet_radio | Y |

# 7 Reference software description

## 7.1 Overview and architecture

The reference software is composed of several languages combined together through utility software as described in Figure 1. The utility software provides an implementation of the non-normative behaviours, and allows exercising the reference software in a typical environment. The utility software is the GPAC player [1], an Open Source multimedia framework distributed under the LGPL license.

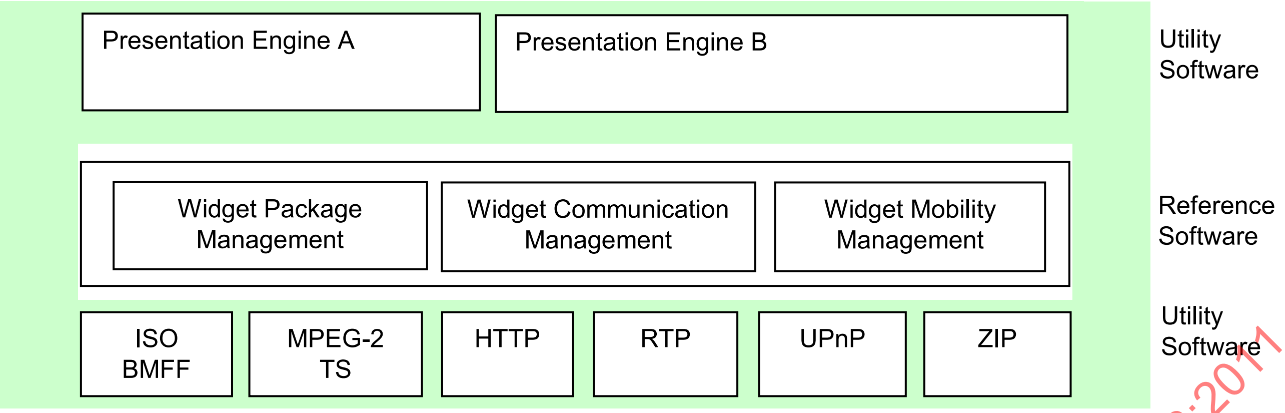

**Figure 1 — Architecture of the MPEG-U Reference Software**

The reference software tests all features defined in ISO/IEC 23007-1. It acts as a conformant W3C WPC User Agent and as a ISO/IEC 23007-1 Widget Manager.

The reference software is structured as follows:

- Widgetman: a C dynamic library acting as a module of the utility software
- mpegu-core.js: a javascript file handling all communications between widgets
- mpegu-wm.js and mpegu-wm.xmt: an example of a ISO/IEC 23007-1 Widget Manager implementation, provided in MPEG-4 XMT-A as defined in ISO/IEC 14496-11, allowing visualization of widgets and user interactions.

The scene description formats supported by the Reference software are the same as the scene description formats supported by the utility software. The reference software is distributed under the MPEG license as indicated in the source code.

## 7.2 Reference software description

### 7.2.1 WidgetManager shared library

This shared library (***gm_widgetman.so/dll/dylib***) is in charge of the following tasks:

- Handling of packaged (ZIP and ISO File Format) widgets,
- Handling of unpackaged widgets,
- Validation of the widget manifest,
- Localization of widget resources,
- Validation of interfaces,
- Providing persistent storage,
- Providing ContextInformation serialization and parsing,
- Implementation of 'widget' and 'MPEGWidget' JavaScript interfaces for the widget scene,
- Modification and monitoring of the different scene structures in the widget for event dispatching.

This shared library also provides JavaScript wrappers to widgets, interfaces and messages, which are exposed to the WidgetManager message broker and UI implementation. The details of these wrappers are given in the message broker javascript file (***mpegu-core.js***).

This module is mostly independent from the scene description used, with the exception of the modification and monitoring of the different scene structures in the widget itself, which must be scene description aware.