INTERNATIONAL STANDARD

ISO/IEC 21000-22

First edition
2016-11-15

AMENDMENT 1
2018-07

Information technology — Multimedia framework (MPEG-21) —

Part 22:
User Description

AMENDMENT 1: Reference software for MPEG-21 user description

Technologies de l'information — Cadre multimédia (MPEG-21) —

Partie 22: Description de l'utilisateur

AMENDEMENT 1: Logiciel de référence pour MPEG-21 description de l'utilisateur

Reference number
ISO/IEC 21000-22:2016/Amd.1:2018(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

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.

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 on 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 the following URL: www.iso.org/iso/foreword.html.

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

A list of all parts in the ISO/IEC 21000 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.

Information technology — Multimedia framework (MPEG-21) —

Part 22: **User Description**

AMENDMENT 1: Reference software for MPEG-21 user description

Introduction

Add new text at the end of the Introduction:

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of a patent (US 14/668,073: Apparatus and method for recommending service).

ISO and IEC take no position concerning the evidence, validity and scope of this patent right. The holder of this patent right has assured ISO and IEC that he/she is willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with ISO and IEC. Information may be obtained from:

Si-Hwan JANG
Electronics and Telecommunications Research Institute (ETRI)
218 Gajeongro
Yuseong-gu
Daejeon
305-700 Korea

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 above. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Clause 8

Add the following new clauses after Clause 8:

9 Reference software

9.1 General

This clause provides a specific implementation including MPEG-21 UD encoder, decoder and validator that behaves in a conformant manner.

9.2 Development environment

Item	Contents
Operating system	Windows 7 Professional 64bit
CPU	Intel Core i5-4570
Memory	8 GB
Development tools	Eclipse (Mars.1) JAXB (2.2.11)
Development language	JAVA 1.8

9.3 Structure of reference software

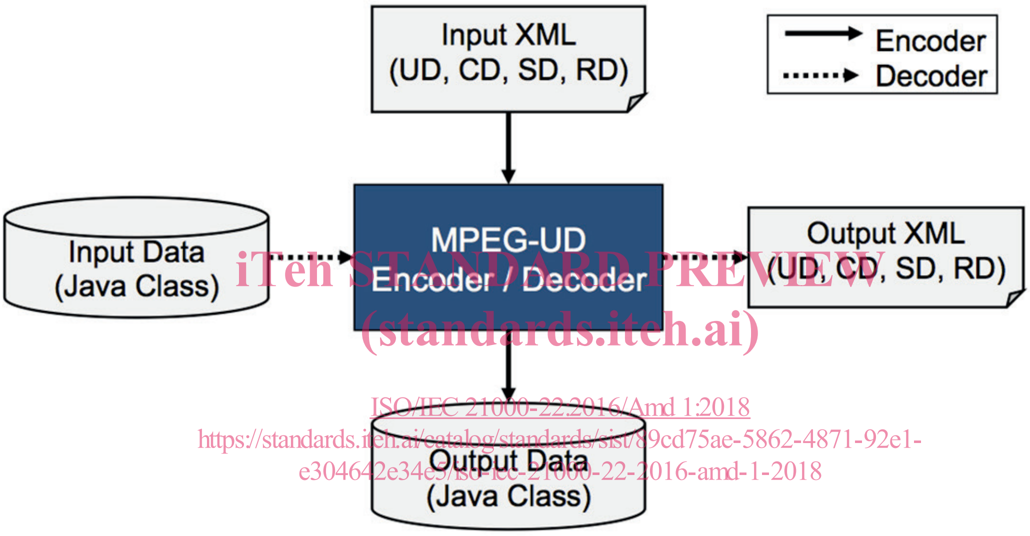

Figure 2 — Flow diagram of MPEG-21 UD reference software

9.4 Reference software classes and method

9.4.1 General

Figure 2 provides a flow diagram of the reference software, as referenced in Annex C, which is available at http://standards.iso.org/iso-iec/21000/-22/ed-1/en/amd/1.

9.4.2 Encoder

The main functionality of the encoder is to generate a standard MPEG-21 UD XML data to an output stream or a file. The encoder is divided into two types of functions *New* and *Make*.

The *New* functions does not need any input parameters and it creates a java instance of requested description type. The return value can be UD, CD, SD and RD class by a new function.

The *Make* function has two input parameters; the first is a type of description instance and the second can be a type of OutputStream and/or File.

- **newUD()** – Create UserDescriptionType JAVA Instance
- **newCD()** – Create ContextDescriptionType JAVA Instance
- **newSD()** – Create ServiceDescriptionType JAVA Instance

- **newRD()** – Create RecommendationDescriptionType JAVA Instance
- **MakeXML(T c, OutputStream/File o)**
  - Parameters
    - c – The c parameter can be one of the following T types:

      UserDescriptionType;

      ContextDescriptionType;

      ServiceDescriptionType;

      RecommendationDescriptionType;
    - o – The o parameter is of the OutputStream type specifying the output stream for the XML data or file type specifying the location where the XML data is to be stored depending on the specified parameter type (OutputStream or File).

9.4.3 Decoder

The main functionality of the decoder is to extract the XML information (e.g., UD, CD, SD and RD). The decoder supports two different parsing methods, one is a URL and the other is a web address.

The first method is to decode UD, CD, SD and RD automatically by a given XML URL. The second method is to decode UD, CD, SD and RD using its *own* function.

- Decode(String pathToFile)
  - Parameters
    - pathToFile– The pathToFile is of String type and its specifies the location of the XML file to read XML data
  - Return Value – Mpeg-21 UD Class (included UD, CD, SD and RD Instance)
- DecodeUD(String pathToFile)
  - Parameters
    - pathToFile– The pathToFile is of String type and its specifies the location of the XML file to read XML data
  - Return Value – UserDescriptionType Class
- DecodeCD(String pathToFile)
  - Parameters
    - pathToFile– The pathToFile is of String type and its specifies the location of the XML file to read XML data
  - Return Value – ContextDescriptionType Class
- DecodeSD(String pathToFile)
  - Parameters
    - pathToFile– The pathToFile is of String type and its specifies the location of the XML file to read XML data
  - Return Value – ServiceDescriptionType Class

- DecodeRD(String pathToFile)
  - Parameters
    - pathToFile– The pathToFile is of String type and its specifies the location of the XML file to read XML data
  - Return Value – RecommendationDescriptionType Class

9.4.4 Validator

The validator is developed to check the validation of an input document. The validator only has one function: *ValidationCheck*. If the input document is invalid, the validator shows an error message on the display device.

- ValidationCheck(String pathToFile)
  - Parameters
    - pathToFile– The pathToFile is of String type and its specifies the location of the XML file to read XML data
  - Return Value – false(0) if the XML file is valid; otherwise, true(1)

9.5 Example using the encoder

The following example shows how to use the encoder. First, it creates instances like UniqueIDType, DeviceProfileType, DisplayType, UserDescriptionType etc. Second, it sets a value in the generated instances and then it sets UserId and UserProfile. Finally, it creates an XML as OutputStream or File.

```
// TODO: add construction code here,
// Create Instance
UniqueIDType udid = new UniqueIDType();
DeviceProfileType dpt = new DeviceProfileType() ;
DisplaysType displaysType = new DisplaysType() ;
DisplayType displayType = new DisplayType() ;
DisplayCapabilityType dcType = new DisplayCapabilityType () ;
ScreenSize sc = new ScreenSize() ;
// Make MPEG-21 UD Class Instance
UserDescriptionType ud = Encoder.newUD() ;
// Set Values
udid.setValue("udID_001");
sc.setHorizontal(1024);
sc.setVertical(768);
dcType.setScreenSize(sc);
displayType.getDisplayCapability().add(dcType) ;
displaysType.getDisplay().add(displayType) ;
dpt.setDevice(displaysType);

//setting the UserID and UserProfile
ud.setUserID(udid);
ud.setUserProfile(dpt);

// Make XML Data
// Case 1 : Output Stream
OutputStream os = System.out;
Encoder.MakeXML(ud, os);

// Case 2 : File
Encoder.MakeXML(ud, new File("testXML.xml"));
```

9.6 Example using the decoder

The following example shows how to use the decoder. The decoder supports two different parsing methods. The first method is to decode automatically a given URL in XML format by using the decode function. The second method is to decode using a specific decode function corresponding to a specific description type like DecodeUD for UD, DecodeCD for CD, DecodeSD for SD and DecodeRD for RD.

```
MpegUD mud = null ;
// Parsing Xml into Java Class data
// Case 1
mud = Decoder.Decode("sample_rd.xml");

// Case 2
UserDescriptionType ud = Decoder.DecodeUD("sample_ud.xml");
ContextDescriptionType cd = Decoder.DecodeCD("sample_cd.xml");
ServiceDescriptionType sd = Decoder.DecodeSD("sample_sd.xml");
RecommendationDescriptionType rd = Decoder.DecodeRD("sample_rd.xml");;

// TODO: add construction code here,
if (mud.ud != null) {
    System.out.println("<UserID>");
    System.out.println("Value : " + mud.ud.getUserID().getValue());
    System.out.println("---------------------------");
}
```

9.7 Example of the validator for reference software

The following example shows how to use the validator for the description. If the return value is false(0), the description, XML document (like UD, CD, SD and RD), is valid. Otherwise, it is invalid.

```
URL url = new URL("sample_rd.xml");

boolean retVal = false;
retVal = Validator.ValidationCheck (url) ;

if (!retVal) {
    System.out.println("Document Valid!!");
}
```

10 Implementation guidelines

This clause contains six example applications to help with the implementation of this document.

10.1 Application 1: Remote Responsive User Interface

10.1.1 General

The Remote Responsive User Interface (RRUI), see Figure 3, is to provide suitable UI according to the user information. When the user connects to this service, the RRUI service engine requests information about the user to the RD engine (recommendation engine), which is located both in and out of the application. The RD engine collects the information and produces RD to help make the best UI with the context manager. This information can consist of UD, CD, and SD. The UD can include a variety of the user's information, such as a basic profile information (birth, gender, etc.) and preference. The CD can include user environment contexts, such as a connected device information and weather. The SD can include marketing and service strategy from a service provider point. The RD engine creates the RD using collected UD, CD and SD. The RD can include the user type/pattern information that help make a more suitable user interface, UD and CD. And then, it sends RD to the RRUI engine. This RRUI engine can create a suitable UI based on RD and send it to the user.

Various user contexts for each user can be aggregated from many context providers, such as agents of device (GPS sensor, illumination sensor) and many service providers (such as Social Network Services provider, portal service provider). However most context providers produce user information based on their individual format. For this reason, it is difficult to aggregate and reuse information directly for a specific application.

To provide a more intelligent service considering both the user environment and the intention of the service provider, user information and service information from many sources should be used easily by an application. For this reason, the contexts for the user and service provider requires standardization to be used commonly by most applications which can't get information directly.

10.1.2 Workflow

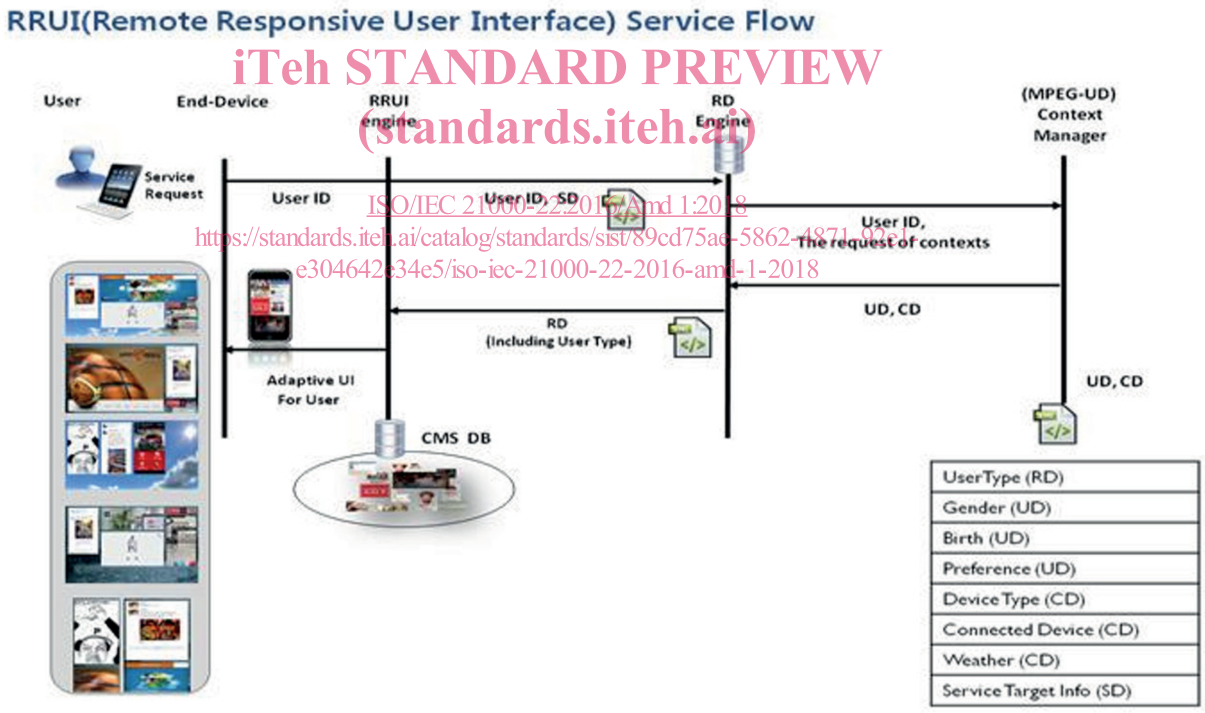

Figure 3 — RRUI service flow

Step 1. The user connects the RRUI application through his/her client device.

Step 2. RRUI Engine sends user ID and SD to the RD Engine.

Step 3. The RD Engine collects the information (UD/CD from context manager, SD from RRUI engine) to produce RD.