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)

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

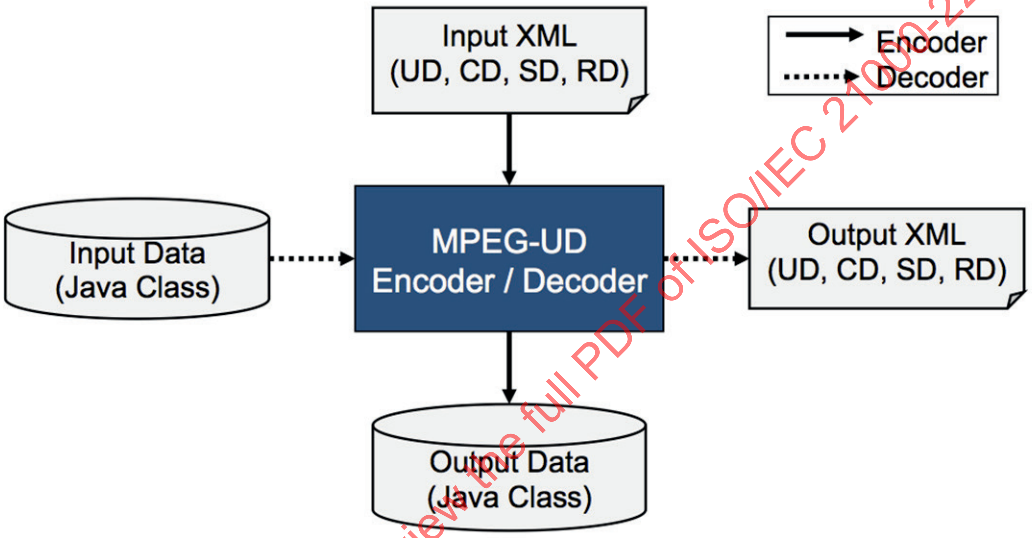


Figure 2 — Flow diagram of MPEG-21 UD reference software

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

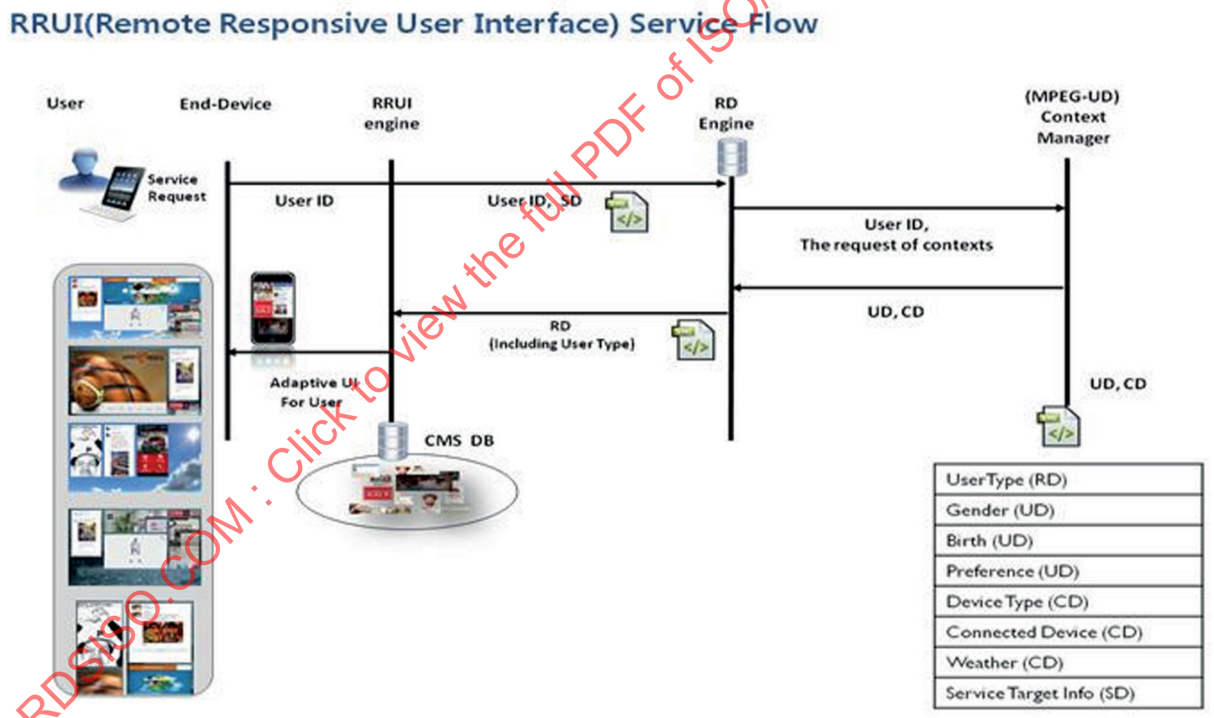


Figure 3 — RRUI service flow

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.

Step 4. The RD Engine creates RD using UD/CD/SD and sends them to the RRUI Engine.

Step 5. The RRUI engine creates a suitable UI instance based on the RD in real time with using CMS, and sends it to the application.

Step 6. The user receives a suitable UI from the RRUI engine.

10.1.3 Validation

10.1.3.1 MPEG-21-UD elements

— Used UD elements: UserID, UserProfile, Preference;

— Used CD elements: UserID, DeviceCharacteristics, Weather;

— Used SD elements: ServiceTargetInformation;

— Used RD elements: ServiceUserType.

10.1.3.2 Experimental results

Suppose some people are attending a conference and one of them tries to find a restroom in the venue. In this case, the place where the person needs to go depends on gender. Showing place suggestions to user after examining his/her gender is easier to comprehend, rather than showing all restrooms in the building. For this reason, the RRUI system provides a customized service for different users with a different interface.

First, the RRUI system checks contexts from a specific agent providing UD. UD includes information about the user. For this scenario, SD has information about "ServiceTargetModel", and there are two factors deciding it. The first one is whether user is an attendee or not, and the second one is the gender of user. The RD engine can figure out "ServiceUserType" using information in UD and analysing the rule in SD (ServiceTargetModel). The RRUI service can finally make a responsive user interface based on "ServiceUserType" in RD. The application scenario is as in Figure 4.

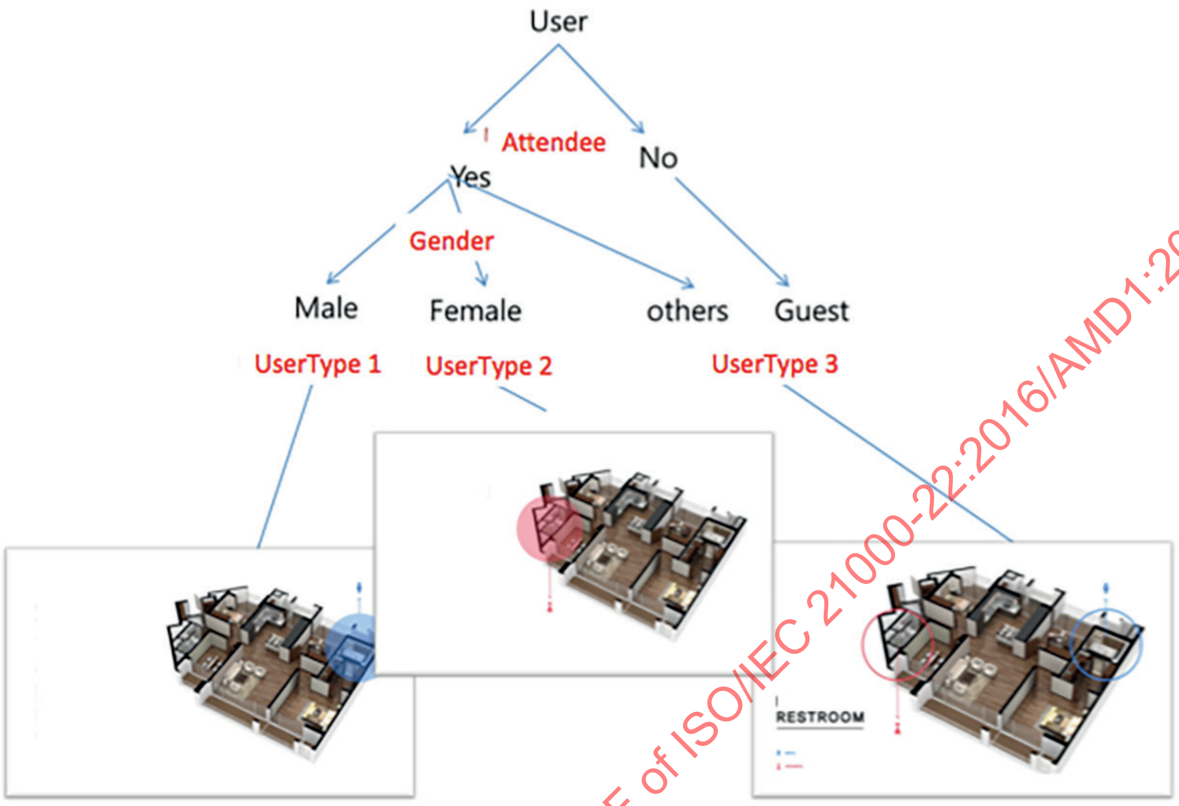


Figure 4 — Example of ServiceUserType

If a user(ID: user_001) who is an attendee and a female that wants to know information about restroom, RD engine decides the ServiceUserType (UserType2) of the user according to the rule (Figure 4) described in SD of RRUI service. And RRUI application sends a responsive user interface (Figure 5) to the user_001.

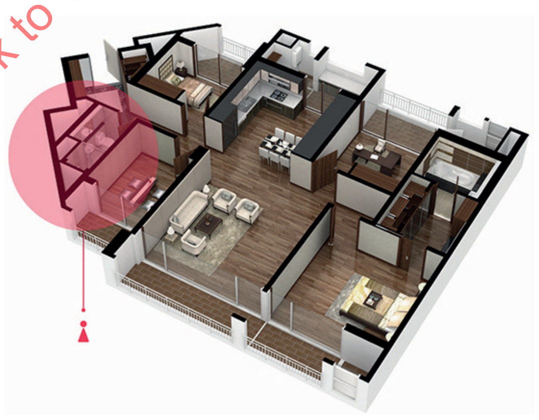


Figure 5 — Final user interface for User_001

This example application can consider another use case, such as various user interfaces for an application launcher, as shown in Figure 6.

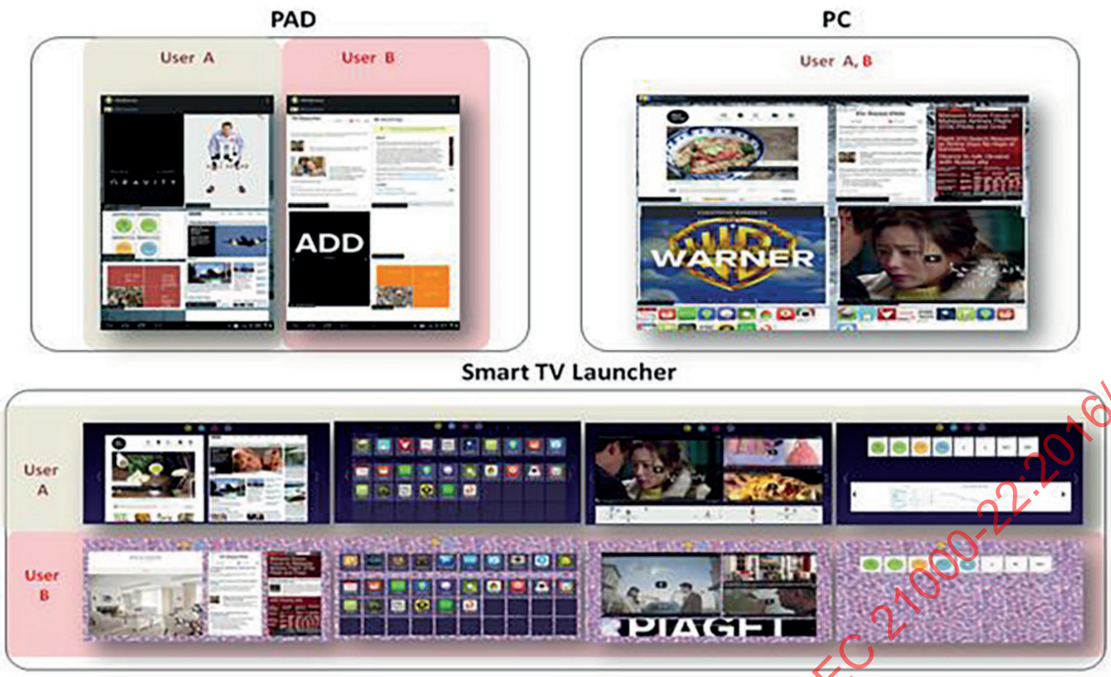


Figure 6 — Various user interfaces for launcher (another use case)

10.2 Application 2: Lossless audio service

10.2.1 General

The specific method to describe the lossless audio of users should be considered.

— The contents provider requests the accessibility profile of the user who is in need of lossless audio service to the mobile audio player or smartphone or Web.

— The database system of lossless audio is very important in terms of how to classify that audio information, such as author, title, and genre.

— Most smartphone manufacturers launch new handsets capable of surfing the Internet using Wi-Fi wireless and 4G technology. If a user does not use a classified database system, when a user needs to find a favourite audio file, the user will waste time searching through unnecessary files.

— The personal contents provider makes lossless audios, using the user description and various authoring tools.

10.2.2 Workflow

A simple service of lossless audio contents with some of UD, CD and SD of lossless audio and elements enabled is presented.

10.2.3 Validation

10.2.3.1 MPEG-21-UD elements

— **Used UD elements:** UserID, UserProfile,UsageHistory, Preference, Emotion, Schedule, Activity, Intension, Preference, AudioPresentationPreferences LosslessAudioPresentationPreference, GeneralAudioPreferencesType, CreationInfo, LosslessAudioFormat, LossyAudioFormat, AudioFileSize, AudioMusicPreference;

— **Used CD elements:** ContextIdentification, ValidTimeDuration, Season, DeviceCharacteristics, NetworkInfo, Weather, OtherEnvironmentalInfo, Priority, LosslessAudioEnvironment, AuthorOfCreating, AuthoringTool, AuthoringAlgorithm;

— **Used SD elements:** ServiceID, ServiceGeneralInformation, ServiceTargetInformation, ServiceInterfaces, InternalServices, Priority, IsServiceAvailable, ServiceObjectInfomation, DatabaseOfMultimedia, LosslessAudioDBType, LossyAudioDBType, VideoDBType, Title, Singer, AuthorOfMusic, KindOfMusic, KindOfFileformat, FirstLineOfMusic, AccessHistoryOfUser, NetworkOfUser.

10.2.4 Experimental results

The user's descriptions of lossless audio contents are present in a webpage. Each elements of UD, CD and SD are classified as subpages. The database system of UD, CD and SD for lossless audio is composed of gathering information by portal site, as shown in Figures 7 and 8.

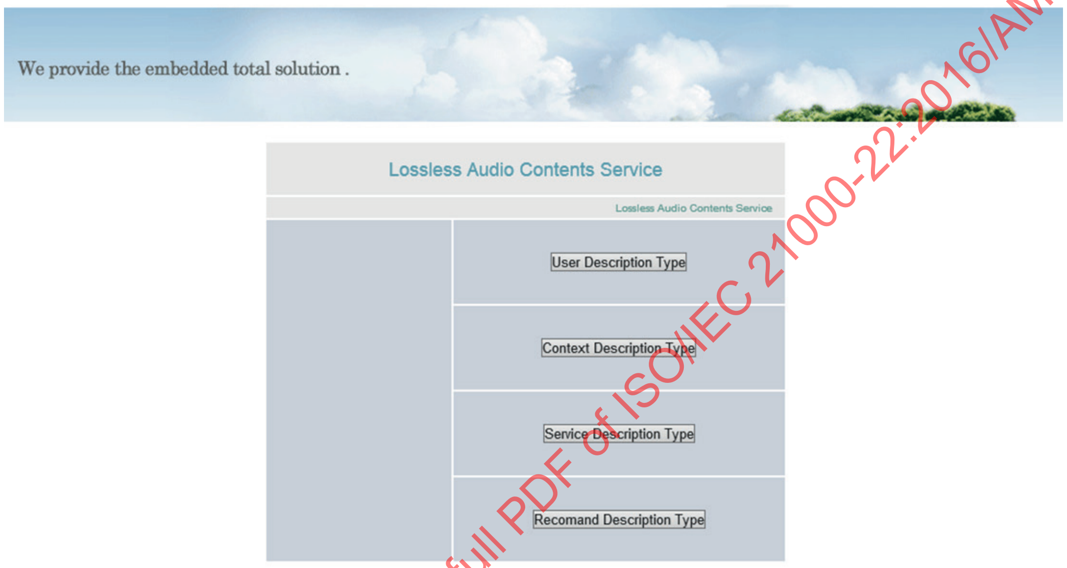

We provide the embedded total solution .
Lossless Audio Contents Service
Lossless Audio Contents Service
User Description Type
Context Description Type
Service Description Type
Recomand Description Type

Figure 7 — Main page of lossless audio contents service

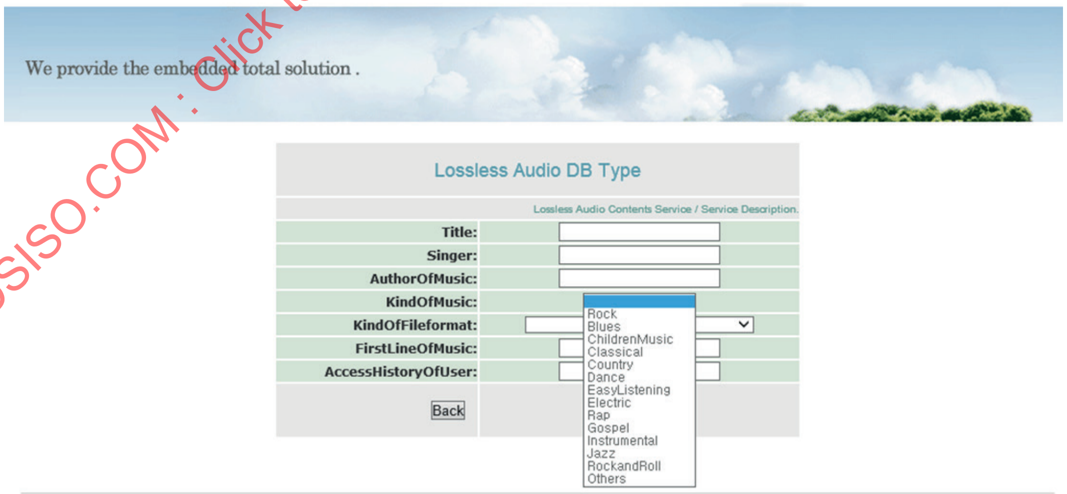


Figure 8 — SD page of lossless audio contents service

10.3 Application 3: Visual communication system

10.3.1 General

Visual objects can be used not only for visual communication messenger, but also in various Social Network Services (SNS). Since these are used in mobile environments, the importance of UI to be more accessible to the visual objects increases. In this context, a recommendation service for visual objects that may be used plays an important role. There are many different recommendation services developed for different purposes, and these recommendation services can be used in different services such as SNS, messenger, etc. With MPEG-21 UD, these recommendation services can serve the user for their different needs (see Figure 9).

A simple scenario demonstrating a program of recommending visual objects (VOs) with respect to a given VO specified by the user in UD is presented in 10.3.

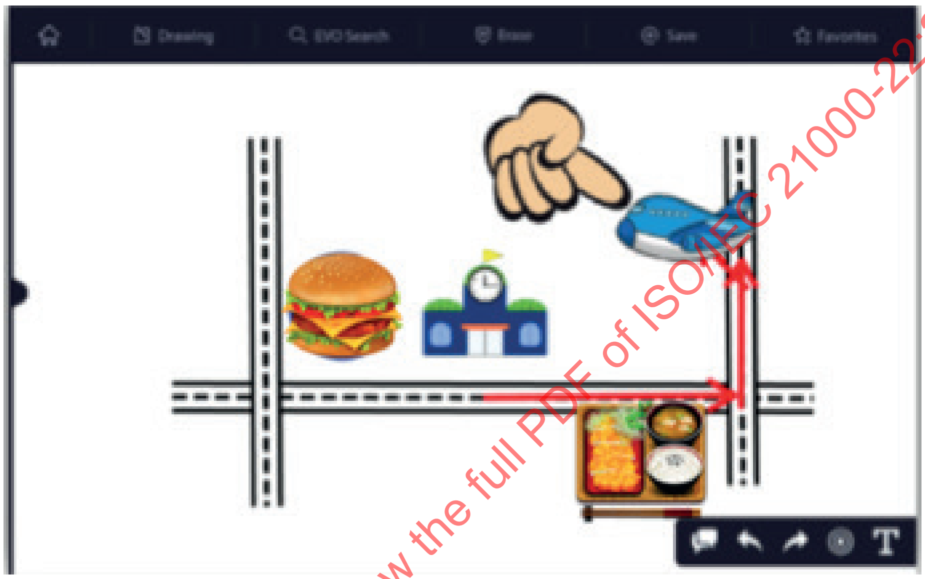


Figure 9 — Conceptual model of visual communications

10.3.2 Workflow

Step 1 A user requests for visual objects associated to restaurants in the visual communication (VC) application. A restaurant is also given in the form of visual objects. This request is described in UD through the VC application.

Step 2 As a list of visual objects associated to restaurants, the VC service provider transfers visual objects related to the user's intention to the recommendation engine.

Visual objects can include information about activity and URI to generate suitable recommendation services in the recommendation engine.

Step 3 From UD, the recommendation engine takes the user service preference information.

Step 4 From CD, the recommendation engine takes the geographic location information specified by the user.

Step 5 Among the visual objects associated to restaurants given by the VC service provider, the recommendation engine selects those that are in the neighbourhood of the user-specified geographic location in CD with user's preference in UD. The selected ones are described in RD, the output of the recommendation engine.

Step 6 The VC application shows the list of recommendations in RD to the user.

— The user can access the URI of recommended visual objects to get more detailed information of a recommended service.

— Visual object can be displayed on the screen of a device according to activity type of visual objects to help users understand.

10.3.3 Validation

10.3.3.1 MPEG-21-UD elements

The elements of UD, CD, SD and RD of ISO/IEC 21000-22 used in the demonstration are as follows.

— UD : UserID, UserProfile, Intention, Preference;

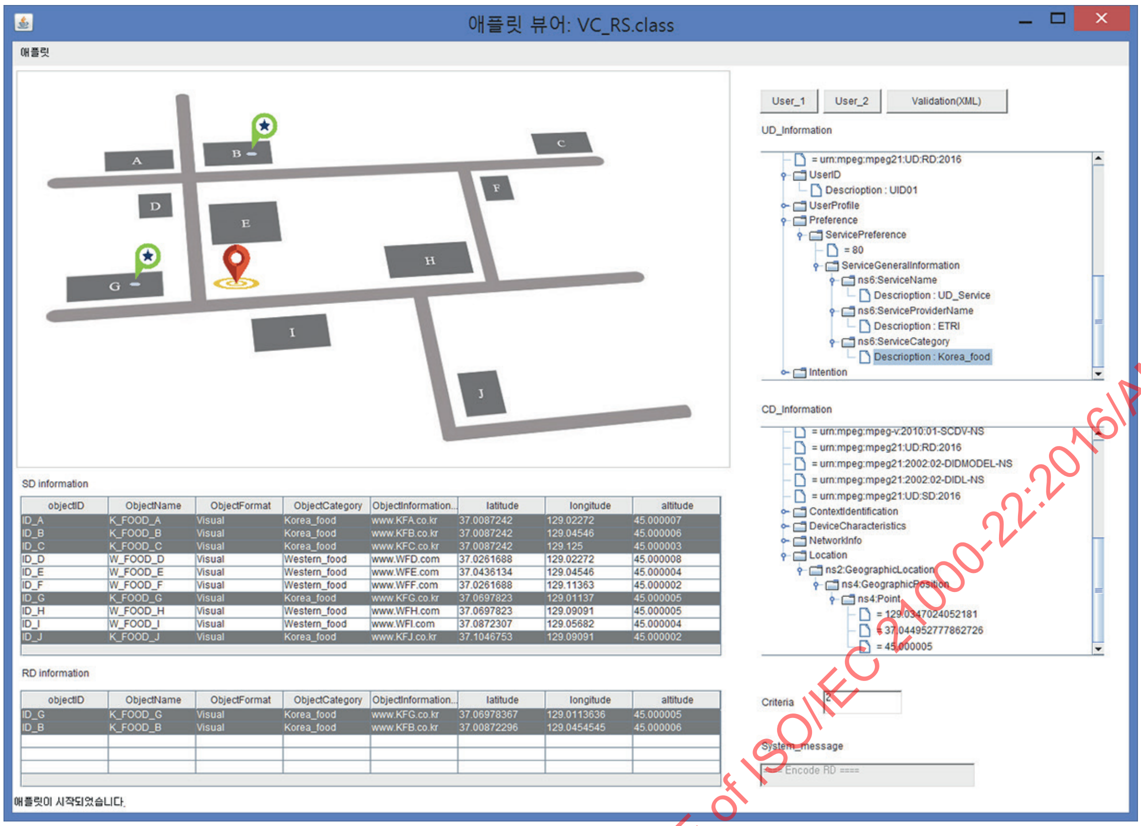
— CD : UserID, DeviceCharacteristics, Location, NetworkInfor;

— SD : ServiceGeneralInformation, ServiceObjectsInformation, ServiceObject, Object;

— RD : RecommendationID, ObjectInformation, RecommendationInformation.

10.3.3.2 Experimental results

Figure 10 shows an implemented VC demonstration program on Eclipse. When a user's location is changed on the street map, the RD engine recommends restaurants considering the location and service category of the object. The number of restaurants recommended by the RD engine is limited by criteria (see Figure 10).



UD_Information

```
= urn:mpeg:mpeg21:UD:RD:2016
UserID
    Descrioption : UID01
UserProfile
Preference
    ServicePreference
        = 80
        ServiceGeneralInformation
            ns6:ServiceName
                Descrioption : UD_Service
            ns6:ServiceProviderName
                Descrioption : ETRI
            ns6:ServiceCategory
                Descrioption : Korea_food
Intention
```

CD_Information

```
= urn:mpeg:mpeg-v:2010:01-SCDV-NS
= urn:mpeg:mpeg21:UD:RD:2016
= urn:mpeg:mpeg21:2002:02-DIDMODEL-NS
= urn:mpeg:mpeg21:2002:02-DIDL-NS
= urn:mpeg:mpeg21:UD:SD:2016
ContextIdentification
DeviceCharacteristics
NetworkInfo
Location
    ns2:GeographicLocation
        ns4:GeographicPosition
            ns4:Point
                = 129.0347024052181
                = 37.044952777862726
                = 45.000005
```

SD information

objectID	ObjectName	ObjectFormat	ObjectCategory	ObjectInformation...	latitude	longitude	altitude
ID_A	K_FOOD_A	Visual	Korea_food	www.KFA.co.kr	37.0087242	129.02272	45.000007
ID_B	K_FOOD_B	Visual	Korea_food	www.KFB.co.kr	37.0087242	129.04546	45.000006
ID_C	K_FOOD_C	Visual	Korea_food	www.KFC.co.kr	37.0087242	129.125	45.000003
ID_D	W_FOOD_D	Visual	Western_food	www.WFD.com	37.0261688	129.02272	45.000008
ID_E	W_FOOD_E	Visual	Western_food	www.WFE.com	37.0436134	129.04546	45.000004
ID_F	W_FOOD_F	Visual	Western_food	www.WFF.com	37.0261688	129.11363	45.000002
ID_G	K_FOOD_G	Visual	Korea_food	www.KFG.co.kr	37.0697823	129.01137	45.000005
ID_H	W_FOOD_H	Visual	Western_food	www.WFH.com	37.0697823	129.09091	45.000005
ID_I	W_FOOD_I	Visual	Western_food	www.WFI.com	37.0872307	129.05682	45.000004
ID_J	K_FOOD_J	Visual	Korea_food	www.KFJ.co.kr	37.1046753	129.09091	45.000002

RD information

objectID	ObjectName	ObjectFormat	ObjectCategory	ObjectInformation...	latitude	longitude	altitude
ID_G	K_FOOD_G	Visual	Korea_food	www.KFG.co.kr	37.06978367	129.0113636	45.000005
ID_B	K_FOOD_B	Visual	Korea_food	www.KFB.co.kr	37.00872296	129.0454545	45.000006

Criteria 2

System_message

==== Encode RD ====

애플릿이 시작되었습니다.

Figure 10 — Visual communication demonstration program

As displayed UD information, the user likes "Korea_food". On the street map, the user stands between "G" and "I". The RD engine of VC demonstration program makes a recommendation considering such a situation.

In the SD information, rows shaded in a grey colour indicate that a record belongs to a category that has the value "Korea_food". The others indicate records corresponding to "Western_food".

In the RD information, records recommended by the RD engine of the VC demonstration program are listed.

Information of UD and CD are mapped to an XML tree. The XML tree for UD displays information of UD1.xml or UD2.xml.

10.4 Application 4: Translation preferences

10.4.1 General

In a language translation system, interoperability between different translation engines is a crucial factor to cover as many languages as possible. UD can be used to facilitate the interoperability in that source languages and target languages described in UD provide information to find a best match automatically among possible translation engines.

In this document, we present a demonstration of the use of the translation preferences, multilingual translation application. The translation preference elements describe the user's preferences for translation services, such as voice gender and source/target languages.

Consider a translation application – a free, speech-based multilingual translation for a smart phone. It is initially aimed at dealing with translation in the travel domain but can be used for general purposes. Currently English, Korean, Chinese and Japanese are the available languages but more languages such as Spanish, German and French are being developed. It is developed for face-to-face translation with Bluetooth communication between two users. In this new platform, UD was applied for using translation preferences. The user's preferred source and target languages can be represented in the

UD to be referenced as the source/target languages when providing the translation service. In addition, the preference for voice gender of the translated voice output can be also represented as one of the translation preference attributes.

10.4.2 Workflow

The workflow of the demo considers Michelle as a Korean tourist addressing a Spanish-speaking person in the following steps (see Figure 11):

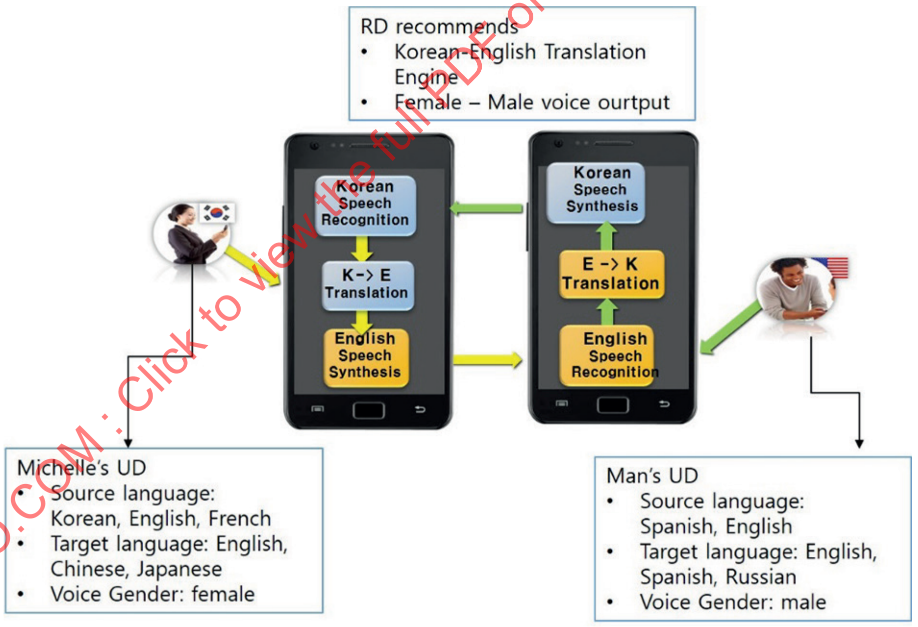
Step 1 In Michelle's UD, her preferred source languages are Korean, English and Chinese. The preferred target languages are English, Japanese and Chinese.

Step 2 She turns on the application and tries to connect to the Spanish speaking person for translation service.

Step 3 The man speaks Spanish but no Spanish translation service is available. However, his second preferred source language is English. The app recommends Korean-English translation and chooses the relevant translation engine. (This is the RD part.)

Step 4 The translation service is started and the translated output is in a woman's voice since Michelle set the preferred voice gender as woman.

Step 5 Michelle can find the metro station with the help of the man.



RD recommends
• Korean-English Translation Engine
• Female – Male voice ourtput
Korean Speech Recognition
K-> E Translation
English Speech Synthesis
Korean Speech Synthesis
E -> K Translation
English Speech Recognition
Michelle's UD
• Source language: Korean, English, French
• Target language: English, Chinese, Japanese
• Voice Gender: female
Man's UD
• Source language: Spanish, English
• Target language: English, Spanish, Russian
• Voice Gender: male

Figure 11 — Using the app where the two users have language preferences

10.4.3 Validation

10.4.3.1 MPEG-21-UD elements

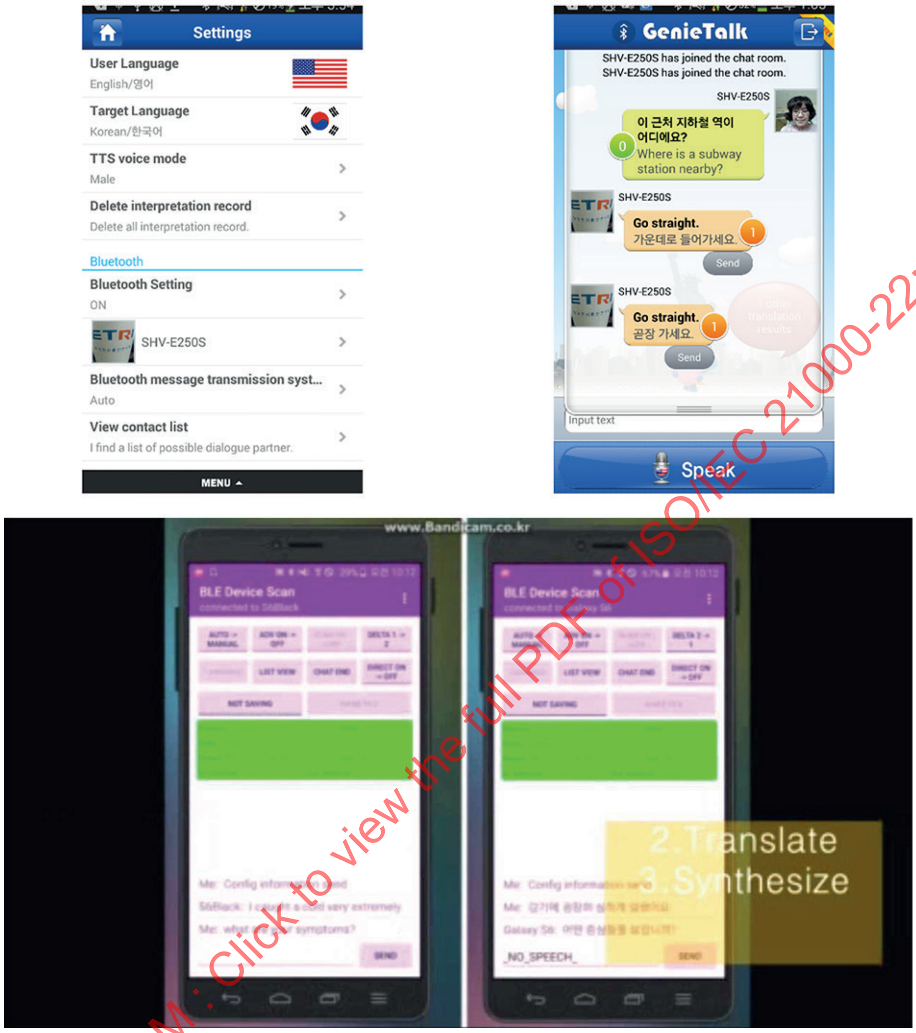
The elements of UD and RD of ISO/IEC 21000-22 used in the demonstration are as follows.

— UD: TranslationPreference, SourceLanguagePreference, TargetLanguagePreference, VoiceGenderPreference;

— RD: RecommendationInformation.

10.4.3.2 Experimental results

Using the translation preference elements of UD, the service provider selects the best match for a source language and target languages and the corresponding translation engine is used with a preferred output voice gender, which is in this case is a female voice for Michelle and a male voice for the Spanish man (see Figure 12).



Settings

User Language
English/영어

Target Language
Korean/한국어

TTS voice mode
Male

Delete interpretation record
Delete all interpretation record.

Bluetooth

Bluetooth Setting
ON

SHV-E250S

Bluetooth message transmission syst...
Auto

View contact list
I find a list of possible dialogue partner.

MENU

GenieTalk

SHV-E250S has joined the chat room.
SHV-E250S has joined the chat room.

SHV-E250S
이 근처 지하철 역이 어디에요?
Where is a subway station nearby?

SHV-E250S
Go straight.
가운데로 들어가세요.
Send

SHV-E250S
Go straight.
곧장 가세요.
Send

Input text

Speak

Figure 12 — Example

10.5 Application 5: Recommending multimedia services

10.5.1 General

Considering state-of-the-art technology, providing a huge amount of services and an ‘application’ of everything is considered a realistic and imminent scenario. Consequently, the problem of a web applications ‘jam’ can be identified as a consequence. In this scenario, recommending users with contents and services they are actually looking for (thus matching) their needs represents a crucial point. In this situation, MPEG-21 UD plays a strategic role in ensuring interoperability between different applications and services.

This demonstration aims to show how two sample apps, e.g. *App 1* and *2*, could mutually exchange standard descriptions which can be exploited for recommendation actions.

10.5.2 Workflow