Agent-Based Intelligent Multimedia Broadcasting within MPEG-21 Multimedia Framework

Munchurl Kim, Jeongyeon Lim, Kyeongok Kang, and Jinwoong Kim

It is expected that an intelligent broadcasting service (IBS) will be able to provide broadcast programs based on user preference and program-associated information (metadata) in order to assist users in easy navigation of the program content being broadcast. In this way, users will be able to access program content anytime, anywhere, and in the manner they wish. This type of IBS will be a basis for future broadcasting services such as customized broadcasting or personal casting.

In this paper, we introduce an agent-based multimedia broadcasting framework using the Foundation for Intelligent Physical Agents (FIPA) and MPEG-7 technologies within MPEG-21. We use a FIPA implementation called FIPA open source as a platform for exchanging user preferences and program information as FIPA messages between a server and its clients. The user preference is modeled as the User Preference description scheme in MPEG-7 multimedia description schemes. We discuss a framework structure and implementation for the IBS.

Keywords: Intelligent broadcasting, MPEG-21, user preference.

Manuscript received Apr. 9, 2003; revised Feb. 8, 2004.

This work was supported in part by the Ministry of Information and Communication of Korea.

Munchurl Kim (phone: +82 42 866 6137, email: mkim@icu.ac.kr) and Jeongyeon Lim (email: jylim@icu.ac.kr) are with the School of Engineering, Information and Communications University (ICU), Daejeon, Korea.

Kyeongok Kang (email: kokang@etri.re.kr) and Jinwoong Kim (email: jwkim@etri.re.kr) are with Digital Broadcasting Research Division, ETRI, Daejeon, Korea.

I. Introduction

Under the digital broadcasting environment, the number of program channels is increasing, making many new broadcast programs available for TV viewers. Furthermore, the program contents are becoming richer with the evolution of content authoring technologies, and user interaction with these new contents is expected to become more natural as well as more popular. The current broadcast services normally provide for non-specific target groups. From a user's perspective, the excessive number of broadcast channels and program contents makes it difficult for TV viewers to select and navigate programs of their interest at a TV terminal [1].

In order to provide user-friendly environments for TV terminals, an electronic program guide is normally offered so that viewers can easily look over broadcast program information for the provided channels such as, program titles, program emission times, program synopses, and so on. However such information is not tailored to expose TV viewers to an excessive amount of information [1]. Thus, the personalization of TV is becoming a necessary and essential part of the future of intelligent broadcasting services. In order to enhance the current broadcasting service technologies towards more customized or personalized services, intelligent broadcast service technologies are needed. Therefore, more specific information on program contents will also become necessary [2], [3].

In this paper, we address an agent-based intelligent multimedia broadcasting framework for intelligent broadcasting services (IBS). The IBS framework adopts an agent software platform called Foundation for Intelligent Physical Agents open source (OS), which is a FIPA-compliant implementation. [4].

For the personalization of TV services, understanding users' interests in TV programming and their consumption behaviors is very important to a broadcasting server which, via a server agent, will receive user preference information from a client agent at a client terminal. The user preference data are delivered as messages via agent communication channels which are supported by the agent platform, FIPA-OS, between the server agent and the client agents. We use the user preference description scheme in MPEG-21 digital item adaptation (DIA) specifications as description models for user preferences on program contents, personal information, user terminals, and user environments [5]. A description of such context information about user preferences, personal information, user terminals and user environments is transmitted as a context digital item (XDI) by a client agent to the server agent via the agent communication channels over a FIPA agent platform. The delivered context information is parsed at the server, and the appropriate program contents and program information are then streamed to the client in a personalized way according to the XDI.

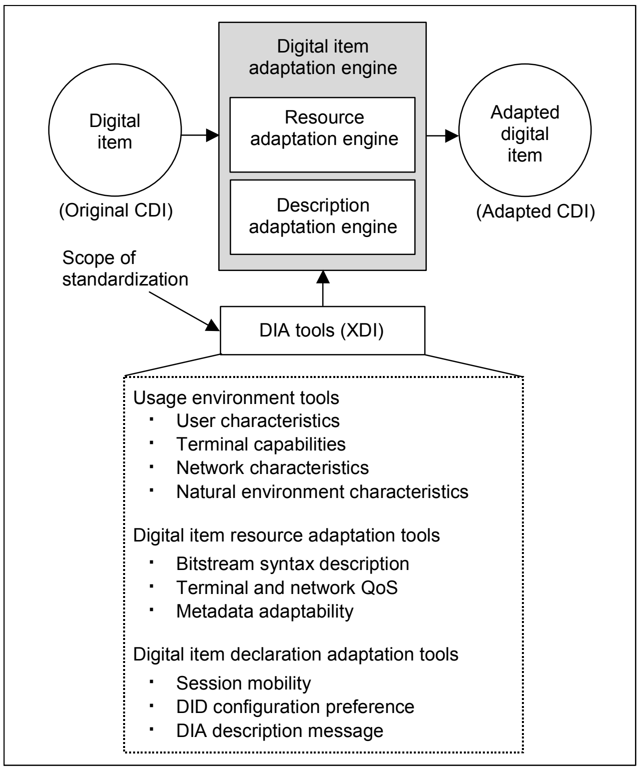
This paper is organized as follows. In section II, we briefly introduce the MPEG-21 multimedia framework and DIA for universal multimedia access [5]. A description model of user preference and usage history, which are specified in MPEG-21 DIA, is described in section III. We also address the FIPA agent platform and its usage for intelligent broadcasting services in conjunction with the MPEG-21 multimedia framework in section IV. In section V, we describe the structure and implementation of the IBS framework. The experimental results are presented in section VI. Finally, the conclusion for the IBS is addressed in section VII.

II. MPEG-21 Multimedia Framework and Digital Item Adaptation

The MPEG-21 multimedia framework aims at a universally accessible and uniquely consumable environment for multimedia under various conditions such as user characteristics, network characteristics, terminal capabilities, and so on. The MPEG-21 multimedia framework makes it possible to define inter-operable, transparent access to advanced multimedia content between terminals and networks as well as the provision of network and terminal resources to form user communities which can create and share reliable and flexible multimedia contents with agreed-upon or contracted quality [6].

MPEG-21 provides an integrated framework for the creation, transaction, delivery, and consumption of digital items [6]. A digital item (DI) is a fundamental unit for the distribution and transaction of digital items in the MPEG-21 multimedia framework, and is defined as a structured digital object in a standard representation with a declaration and identification of its resources (multimedia contents). In order to provide such an integrated multimedia framework, MPEG-21 defines a set of normative specifications: Digital Item Declaration in part 2; Digital Item Identification in part 3; Intellectual Property Management and Protection in part 4; Rights Expression Language in part 5; Rights Rata Dictionary in part 6; Digital Item Adaptation in part 7; Reference Software in part 8; File Format in part 9; Digital Item Processing in part 10; Persistent Association Technology in part 11; Testbed for MPEG-21 Resource Delivery in part 12; and Scalable Video Coding in part 13 [6].

MPEG-21 DIA specifies the syntax and semantics of the defined tools which are needed for universally accessible and uniquely consumable multimedia environments. The current MPEG-21 DIA provides the tools needed for the usage environment, resource adaptability, and digital item adaptation as depicted in Fig. 1.


Digital item adaptation engine
Digital item
Resource adaptation engine
Adapted digital item
(Original CDI)
Description adaptation engine
(Adapted CDI)
Scope of standardization
DIA tools (XDI)
Usage environment tools
・ User characteristics
・ Terminal capabilities
・ Network characteristics
・ Natural environment characteristics
Digital item resource adaptation tools
・ Bitstream syntax description
・ Terminal and network QoS
・ Metadata adaptability
Digital item declaration adaptation tools
・ Session mobility
・ DID configuration preference
・ DIA description message
Fig. 1. Concept of digital item adaptation (DIA) and DIA tools.

The usage environment tools include the descriptions for user characteristics, terminal capabilities, network characteristics and natural characteristics. The resource adaptability tools are used for describing the structural information for bitstreams and information about metadata adaptability using Binary Syntax Description Language. And, DIA declaration tools provide a mechanism to preserve a user's state for interaction with a DI and the configuration preference tools for the user's intention of Digital Item Declaration choice configuration preferences. These tools are used together to constitute the MPEG-21 DIA framework [5].

III. User Preference Model

Understanding the characteristics of TV viewers is an essential part of intelligent broadcasting services. User characteristics can be modeled in terms of user preferences, such as creation, source, classification filtering, browsing, and user behaviors that record a user's consumption history of broadcast program contents.

MPEG-21 DIA provides a user description model for use as a content preference description scheme in the usage environment tools. This content preference description scheme contains the user preference and the usage-history description scheme, which are adopted from the MPEG-7 multimedia description schemes (MDS) and TV Anytime specifications [7]. The user preference description scheme defines a structured description model for a user's preference in browsing and filtering/searching. [7]. The usage-history description scheme defines a structured model for describing the usage history of content users. The description data for a user's preference and content usage history can be used in TV terminals such as a set-top box or PC.

Figure 2 shows a content filtering, searching, and browsing application using user preferences and usage history information. Content broadcasters provide AV program contents with their associated metadata, which describe the contexts of the contents. At a user terminal, a filtering software agent may be run to analyze the metadata and select the program channels or contents that are preferred by the user. For a searching scenario, a user can query his or her favorite program title using the filter and search engine which then analyzes the metadata, and he or she can then find the relevant program information such as the channel number, program title, emission time, casters, and so on. In addition to the filtering and searching functionalities for metadata, a browsing functionality can also be possible based on the user preferences. Some users prefer to consume a summarized audiovisual skim before watching the entire content. Users can retrieve and browse their preferred contents based on their preferences. For example, an audiovisual summary of a video can be browsed in a poster manner with several key thumbnail images, or it can be played with a short video skim.

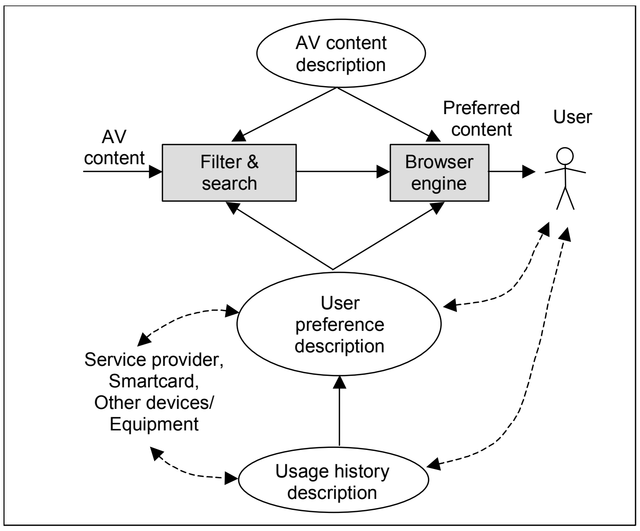

AV content description
Preferred content
User
AV content
Filter & search
Browser engine
User preference description
Service provider, Smartcard, Other devices/ Equipment
Usage history description

Fig. 2. A filtering, search and browsing application using user preference and a usage history description [7].

User preferences are very valuable information for user-oriented or personalized applications. However, user preferences need to be defined in a somewhat quantitative way. One simple solution is to let users set the values for given preference types. In this case, the users should be aware of the preference types available and the semantics of the specific values these types are set to.

It is more appropriate to compute user preferences based on usage history of the program contents. By analyzing the usage history data, the user preference values can be adaptively computed according to the user preferences as they change over time.

As shown in Fig. 2, the usage history description can be used at a terminal application in order to automatically compute the user preference values which are used in the filter and search engine or browser engine. The MPEG-7 MDS and TV Anytime metadata specifications define the syntax and semantics of a user preference and usage history schema, but the extraction methods for user preference values are not specified in any standardized way.

1. User Preference Description Scheme

Figure 3 shows a hierarchical structure of a User Preference description scheme (DS), and Table 1 shows User Preference types.

The User Preference DS contains the Browsing Preference DS, Filtering and Searching Preference DS, and the User Identifier. The User Identifier is used to identify users for user preference description data. The Filtering and Searching Preference DS specifies creation preference, source preference and classification preference. The Browsing Preference DS has the Summary Preference DS which contains the SummaryTypePreference, PreferredSummaryTheme, and other summary types. Figure 4 shows the syntax of the User Preference DS specified by

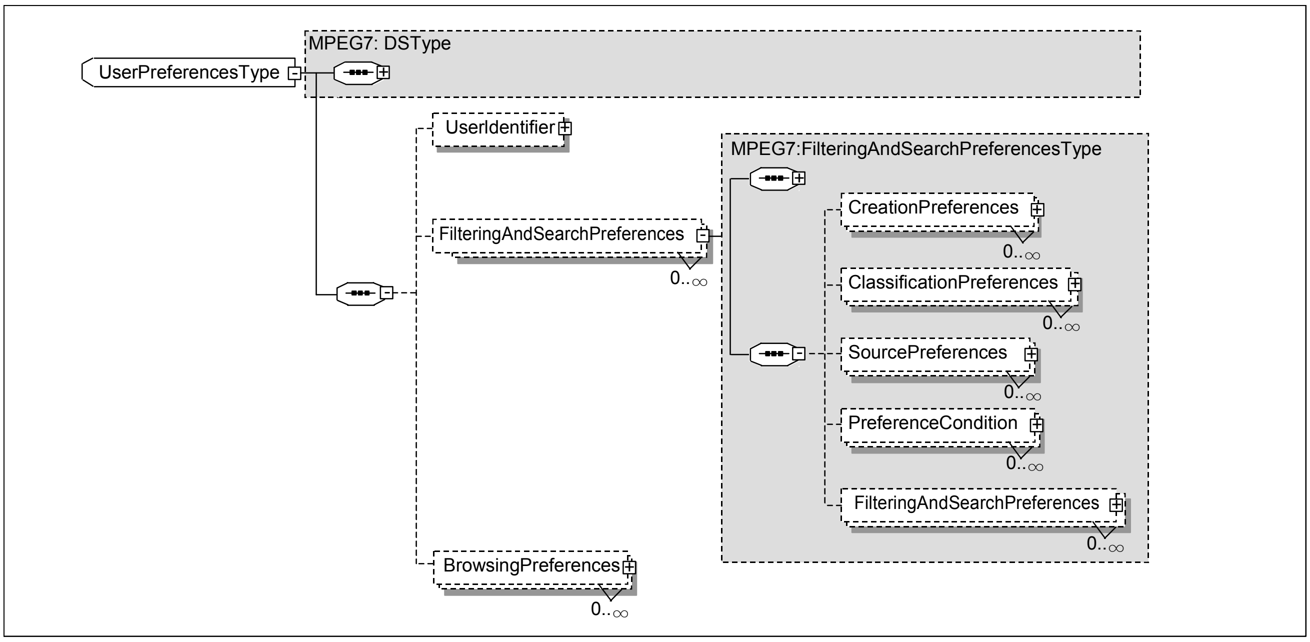


Fig. 3. Hierarchical structure of User Preference description scheme.

Table 1. User Preference types.

Browsing Preferences DS	FilteringAndSearchPreferences DS		
	Creation Preferences	Classification Preferences	Source Preferences
· Summary Type Preference · Preferred Summary Theme	· Title · Creator · Keyword · Location · DatePeriod	· Country · DatePeriod · Language · Genre · Subject · MediaReview · Parental Guidance	· Publication Type · Publication Source · Publication Place · Publication Date · Publishier · MediaFormat

```
<complexType name="UserPreferencesType">
  <complexContents>
    <extension base="mpeg7:DSType">
      <sequence>
        <element name="FilteringAndSearchPreferences"
                    type="mpeg7:FilteringAndSearchPreferencesType"
              minOccurs="0" maxOccurs="unbounded"/>
        <element name="BrowsingPreferences"
                    type="mpeg7:BrowsingPreferencesType"
                    minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="allowAutomaticUpdate"
                  type="mpeg7:userChoiceType"
                  use="default" value="false"/>
    </extension>
  </complexContents>
</complexType>
```

Fig. 4. Syntax of User Preference DS.

MPEG-7 Description Definition Language.

2. Usage History Description Scheme

The Usage History DS specifies the syntax and semantics for describing usage behaviors for various contents. A user's usage behavior, or action types, include *Record, Play, Pause, Fast Forward, Fast Backward*, and so on. Such user actions have different meanings in terms of their preferences about the content. For example, if a user presses *Record*, it can be interpreted that the content being recorded is a preferred content.

Figure 5 illustrates a hierarchical structure of the Usage History DS. The Usage History DS contains the User Action History DS and User Identifier descriptor. The User Identifier descriptor is used to distinguish the usage-history description data of various users. The User Action History DS contains the Observation Period descriptor, which describes the total duration of a user's consumption of contents, and the User Action List DS. The User Action List DS contains the Action Type descriptor that describes the action types, *playback, stop, pause, Fastforward*, and so forth. The User Action DS provides detailed information about an individual user action such as the time of occurrence, duration of use, associated program identifier, and references to related content descriptions and material. The *numInstance* describes the total number of user actions, and the *totalDuration* describes the total time taken to consume specific program content. The User Action DS describes a user action with the time of occurrence,

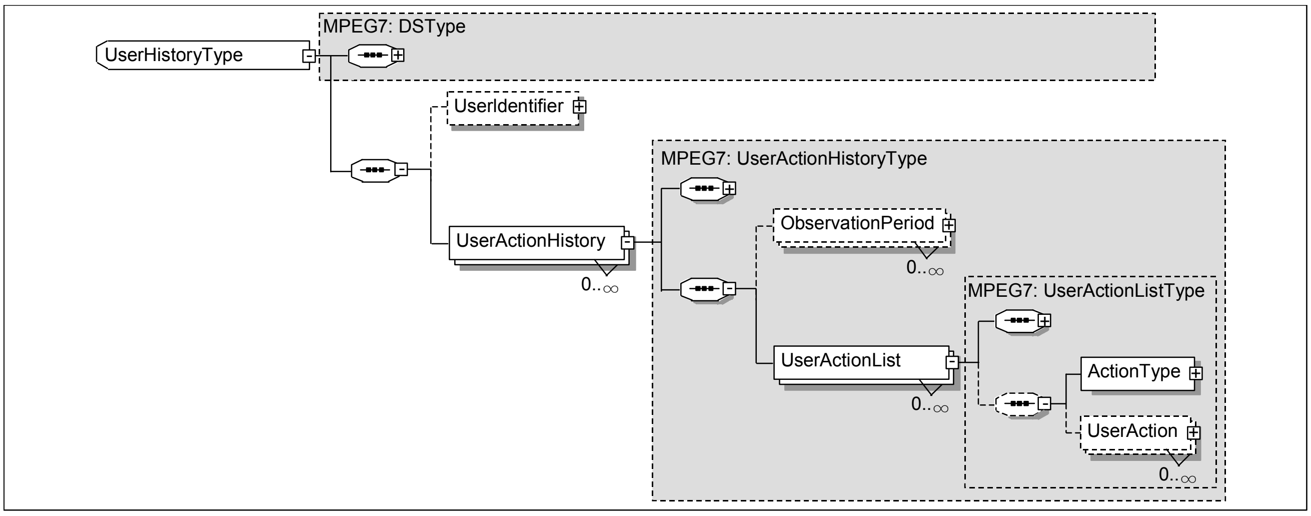


Fig. 5. Hierarchical structure of the Usage History description scheme.

duration, associated program identifier, and references to a related content description and its associated content.

Figure 6 indicates the syntax of the Usage History DS and User Action List DS.

```
----- User Action History DS -----
<complexType name="UserActionHistoryType">
  <complexContent>
    <extension base="mpeg7:DSType">
      <sequence minOccurs="0">
        <element name="ObservationPeriod"   type="mpeg7:TimeType"
                        minOccurs="1" maxOccurs="unbounded"/>
        <element name="UserActionList"
                      type="mpeg7:UserActionListType"
                      minOccurs="1" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="protected" type="mpeg7:userChoiceType"
                     use="default" value="true"/>
    </extension>
  </complexContent>
</complexType>

----- User Action List DS -----
<complexType name="UserActionListType">
  <complexContent>
    <extension base="mpeg7:DSType>
      <sequence minOccurs="0">
        <element name="ActionType"   type="mpeg7:TermUseType"/>
        <element name="UserAction" type="mpeg7:UserActionType"
                          minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="numInstances"
                        type="nonNegativeInteger" use="optional"/>
      <attribute name="totalDuration"
                        type="mpeg7:durationType" use="optional"/>
    </extension>
  </complexContent>
</complexType>
```

Fig. 6. Syntax of the Usage History DS and User Action List DS.

3. Computation of User Preference Values

A. Statistical Approach

Preferences based on categories can be modeled with probabilities in a statistical framework. In Fig. 7, genre can be represented as a tree structure. So the preference for each genre can be represented as its probability, which reflects the frequency of the genre visited.

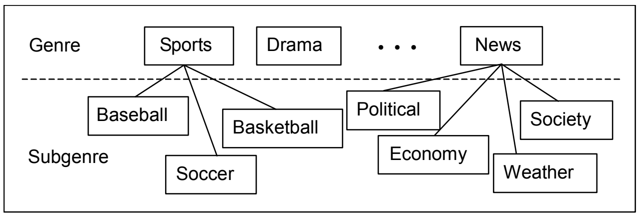


Fig. 7. An example of a genre tree.

The probability $P(g_i)$ of the *i*-th genre g_i is expressed as

$$P(g_i) = T_{g_i} / T_G ,$$

where T_{g_i} is the total time taken in watching program contents of the *i*-th genre, and T_G indicates the total time needed for viewing the program contents of all the genres. Note that

$$\sum_i P(g_i) = 1.$$

For the probabilities of the subgenres, the conditional probability $P(sg_j | g_i)$ of the *j*-th subgenre sg_j given the *i*-th genre g_i is expressed as

$$P(sg_j | g_i) = T_{sg_j} / T_{g_i} ,$$

where T_{sg_j} is the total time taken in watching the program contents of the *j*-th subgenre.

The preference for a specific genre can be modeled by the relative amount of time taken to consume all the broadcast contents belonging to that genre. The higher the probability value for a specific genre, the more preferred the genre is. The probability density function values can be obtained by histogram computation over the content watching time. This preference can be combined with the probability value of the subgenre preference to support the recommendation functionality for preference genres at the preferred times.

B. Rule-Based Approach

The preferences for such genre/subgenre, actors, and directors can be modeled as a probability. Sometimes the user behaviors need be interpreted for the computation of the preference values. For example, a user action such as *Record* while watching the program content implies the relative importance of the content. On the other hand, a user action type such as *Fast Forward* indicates a relatively lesser importance for the content. Such series of user action types are very common and can be utilized in reasoning a degree of preference for specific program content.

We set up a rule in order to interpret such user action types taken in a series as follows:

```
Case 1: Preview -> Play (high)
Case 2: Play -> Stop (middle):: refresh
Case 3: Play -> Preview (middle):: refresh
Case 4: Play -> Fast Backward -> Play (high)
Case 5: Play -> Fast Forward -> Play (middle)
Case 6: Play -> Fast Backward -> Pause -> Play (high)
Case 7: Play -> Pause -> Fast Backward -> Play (high)
Case 8: Play -> Fast Forward -> Pause -> Play (middle)
Case 9: Play -> Pause -> Fast Forward -> Play (middle)
Case 10: Any other action (low)
```

The user action types include *Preview, Play, Pause, Stop, Fast Forward, Fast Backward, Skip one frame*, and *Back one frame*. Cases 1 through 10 indicate the possible combinations of user actions in order and their corresponding significance on the content being played.

IV. FIPA

1. FIPA Standard

FIPA aims at providing a set of standard specifications for inter-operable agent applications and agent systems. The visual description of all components for FIPA is shown in Fig. 8.

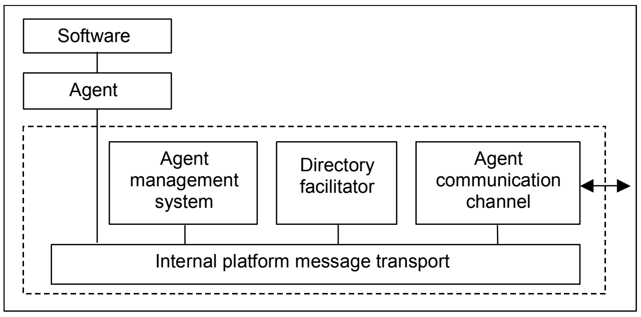

Software
Agent
Agent management system
Directory facilitator
Agent communication channel
Internal platform message transport

Fig. 8. Normative components in FIPA standard [8].

In order to provide an inter-operable platform among agents in a multi-agent management environment, FIPA specifies a minimal set of normative components. The first component is a message transport service (MTS) which consists of an agent communication channel used to route messages between agents within the platform and agents that reside on other platforms, and an internal platform message transport to route messages among agents that reside in the same platform. The second component is an agent management system that controls the creation, deletion, suspension, resumption, authentication and migration of agents on the agent platform and acts as a “white pages” directory for all agents resident on the agent platform—the mapping between globally unique agent names and local transport addresses used by the platform are maintained by this agent management system. The final component is a directory facilitator that acts as a “yellow pages” directory for the agents and maintains descriptions of the agents and their services [4], [9].

2. FIPA-OS

The FIPA-open source (OS) is a multi-agent platform which was designed and implemented to support the FIPA agent standards [9]. It not only implements the mandatory components of the FIPA architecture but also provides an agent shell, a multi-layered agent communication scheme, and a message and conversion scheme so that agent applications can be easily built up on top of it.

Messaging in FIPA-OS is handled by the MTS which is logically designed with a stack of services. Messages enter the stack and then exit into a transport layer along with an appropriate message transport protocol (MTP) which is selected by the MTS. Each service does transformation and inverse transformation for outgoing and incoming messages. The advantage of a services stack design is that every service can be tested independently [9].

MTP provides a mechanism for sending and receiving messages from one agent to another. It consists of an internal transport and external transport. The internal transport is used for agent communication within the platform. The external transport is mainly used by the Agent Communication Channel (ACC) for inter-platform communication [9]. Implementing normative components according to FIPA standards, as shown in Fig. 8, FIPA-OS implements both the internal platform message transport component and the ACC component in the MTP by using the internal transport and the external transport, respectively.

MTP in FIPA-OS currently supports the following four protocols: the internet inter-ORB (object request broker) protocol, which uses Sun's CORBA (common object request broker architecture) implementation by default; HTTP, based on FIPA HTTP specifications and currently used only by ACC; FIPAOS-RMI (remote method invocation), based on RMI; and FIPAOS-SSL-RMI, based on RMI over a SSL (secure socket layer [9].

There are several advantages of a multi-layer and multi-component design for MTS. First, this design makes every layer and component independent of each other so that they can be tested independently [9]. Second, we can replace every component with our own implementation if needed. In the next section, we will introduce MicroFIPA-OS and give a brief explanation of communication between FIPA-OS and MicroFIPA-OS.

3. MicroFIPA-OS

MicroFIPA-OS is a lightweight FIPA-OS. It was designed for small footprint devices such as mobile phones, personal digital assistants, and so on. Although it is a lightweight version, MicroFIPA-OS, like FIPA-OS, implements the standard specifications of FIPA with the support of an agent communication protocol. In this way, inter-operability between FIPA-OS and MicroFIPA-OS can be achieved so that a FIPA-OS agent can also process messages sent from agents running on the MicroFIPA-OS platform, and vice versa.

The agents deployed on MicroFIPA-OS run on devices that have some limitations in processing power, memory capability and communication flexibility. To deal with these limitations, the transport architecture of MicroFIPA-OS is optimized for small devices. The MicroFIPA-OS transport architecture is designed to be platform-centric, which means MicroFIPA-OS agents share one MTS, while FIPA-OS is agent-centric, giving each FIPA-OS agent its own MTS. Also, MicroFIPA-OS doesn't use a stack of services, but employs the protocol components of a different default MTP. And, it only allows either an internal HTTP transport or the FIPA specified HTTP protocol for inter-operability, while the FIPA-OS supports RMI or CORBA for message delivery [10].

The scenarios used for deploying FIPA-OS and MicroFIPA-OS together are shown in Fig. 9. In the first scenario, a MicroFIPA-OS agent runs as a part of a FIPA-OS platform. It requires that both MicroFIPA-OS and FIPA-OS employ the same internal transport. This means we can implement the HTTP transport for the FIPA-OS internal protocol or adopt either RMI or CORBA as an internal protocol for MicroFIPA-OS. In the second scenario, both MicroFIPA-OS and FIPA-OS stand independently by running their own agent management system and directory facilitator, implementing the same external transport to communicate with each other [10].

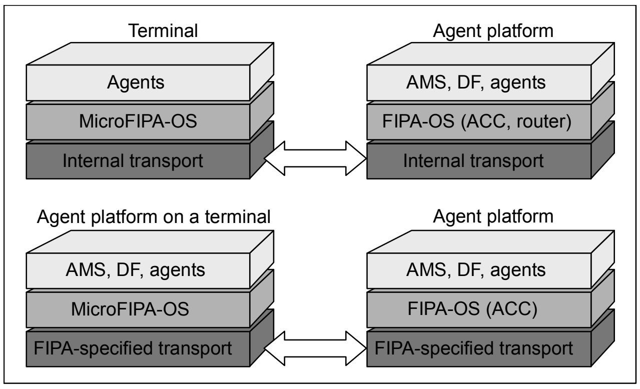


Fig. 9. Deployment of FIPA-OS and MicroFIPA-OS [10].

V. Design and Implementation of IBS Framework Based on FIPA

The IBS framework includes a server agent and several client agents in the FIPA agent platform, FIPA-OS . The server, at which a server agent is running, contains a metabase and a content archive which maintain the information about broadcast program contents and which store the program contents to be broadcast to the client sides. A client agent in the IBS framework recommends to the user the received program information from the server agent. Therefore, a user can easily navigate and browse his or her favorite programs from the tailored program information provided by the server agent based on user preference information.

Figure 10 illustrates the architecture of the IBS framework. The broadcast server includes a media streaming server, a metabase (program information database), a streaming server, a content archive and a server agent. Each client terminal has client agents running on it.

1. Broadcasting Server Agent

The broadcasting server agent provides program information

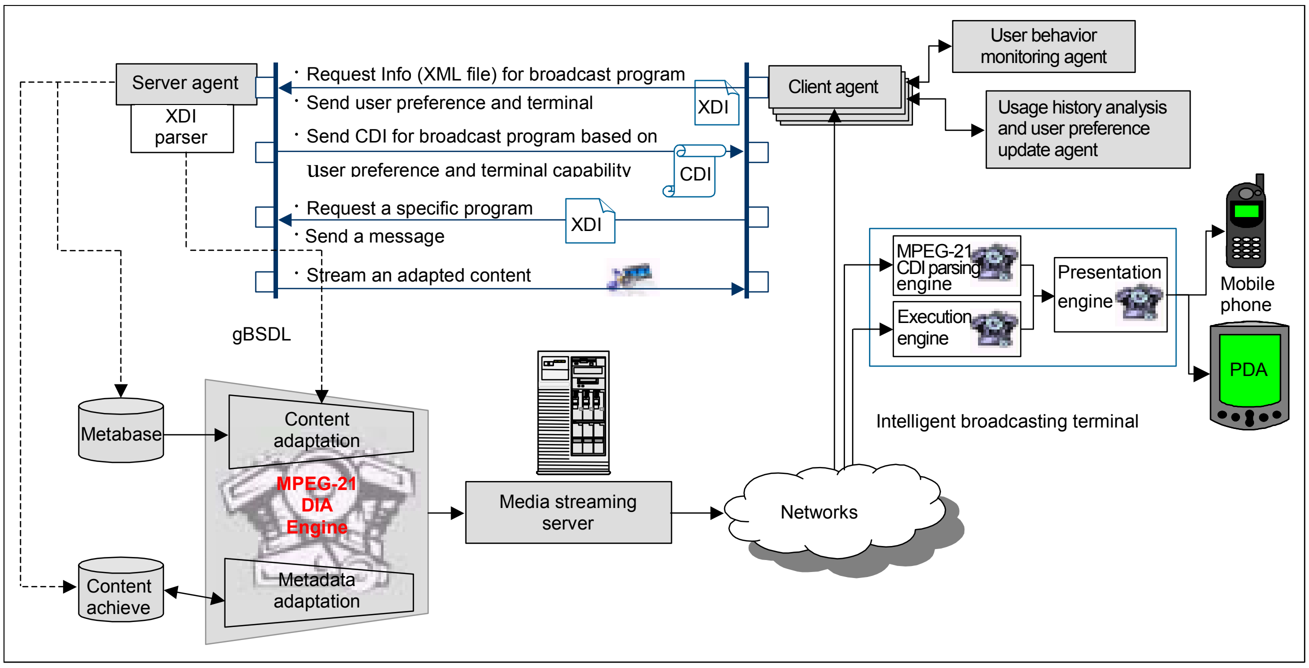

Server agent
XDI parser
· Request Info (XML file) for broadcast program
· Send user preference and terminal
XDI
· Send CDI for broadcast program based on
user preference and terminal capability
CDI
· Request a specific program
· Send a message
XDI
· Stream an adapted content
Client agent
User behavior monitoring agent
Usage history analysis and user preference update agent
MPEG-21 CDI parsing engine
Execution engine
Presentation engine
Mobile phone
PDA
Intelligent broadcasting terminal
gBSDL
Metabase
Content adaptation
MPEG-21 DIA Engine
Metadata adaptation
Content achieve
Media streaming server
Networks

Fig. 10. Architecture of agent-based IBS framework.

to client agents according to user preference in the IBS framework and requests the streaming server to stream the corresponding program contents upon user request when the user selects one of the programs delivered by the server agent and presented by the client agent. Notice that the tailored program content information is represented as an MPEG-21 content digital item (CDI) based on the MPEG-21 context digital item (XDI) which contains the user characteristics description delivered from the client. Communication between a client agent and the server agent is performed by agent communication language (ACL) messages. Here, MPEG-21 CDI and XDI are encapsulated as ACL messages, and the delivery is performed on the agent platform. The server agent controls the client agent connections and sends program information to the client agents. All information related to program resources, user preference and terminal characteristics, is represented by metadata (MPEG-21 CDI and XDI) according to the MPEG-21 Digital Item Declaration specification [11], by which the MPEG-21 CDI is instantiated. Again, metadata is encapsulated in an ACL message to deliver and communicate between inter-platforms.

The broadcasting server agent parses the XDI received from the client agents and then processes the user preference and user terminal characteristics information included in the XDI. The broadcasting server agent sends to the client agent the program information according to the user preference. Here, the broadcast program content information is represented as an MPEG-21 CDI and is encapsulated in a FIPA message by the FIPA ACL. The client agent can then receive the CDI from the server agent.

2. Client Agent

The client agent interacts with both users and the broadcasting server agent. It also interfaces with users. It presents the broadcast program lists embedded in the MPEG-21 CDI and delivered from the broadcasting server agent. Notice that the presented broadcast program lists are based on user preference, so the user can easily navigate and access his or her favorite broadcast program contents. The user-behavior monitoring agent monitors user actions by recording them as the usage history description data which are then analyzed by the usage history analysis and user preference update agent. The accumulated usage history data is maintained in the usage history repository, which is shown in Fig. 11.

The User Behavior Monitoring agent stores the usage history data analysis. The Usage History Analysis and User Preference

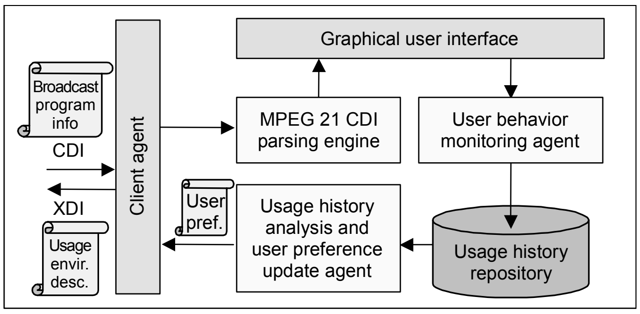


Fig. 11. Information flow at a client side.

Update agent learns a user's changes in preference by analyzing the usage history data and updates the user preference adaptively.

VI. Experimental Results

The IBS framework was implemented on Microsoft Windows XP with FIPA-OS v2.1 on JDK 1.4.0. Communication between agents was made on ACL using an MPEG-21 CDI and XDI. Program information was maintained in the metabase—we used Microsoft SQL Server 2000. To query the metabase, we used Java Database Connectivity connected with Open Database Connectivity. Our program runs on a java virtual machine so the agents can be transplanted among different OS. Metadata processing was performed by JDOM-8. To play streaming media data, Windows Media Player was used in the form of an ActiveX control. The metabase has meta data on 776 programs in 11 different genres.

1. Delivery of Context Information

Every agent attached to FIPA-OS or MicroFIPA-OS communicates by using a standard messaging system defined by FIPA. In the FIPA standard, a message is made up of two parts, a message envelope expressing transport information and a message body comprising the ACL [12]. A FIPA message envelope consists of a collection of name/value pairs that comprises the necessary parameters. It contains at least the mandatory `:to`, `:from`, `:date`, and `:acl-representation` parameters [9]. This minimal set of parameters is enough to perform essential communication. For the full set of envelope parameters, FIPA defined additional parameters such as `:comments`, `:payload-length`, `:payload-encoding`, `:encrypted`, and `:transport-behavior` [13].

ACL message representation is based on XML. It contains a set of one or more message elements. ACL must include the following minimal set of elements: *sender, receiver, content*, and *performative*. In our application, we attach both XDI and CDI in the content element of the ACL message.

When we construct the message in the agent application level, FIPA-OS implementation provides a template object that includes values which need to be filled in as necessary. All the information is encapsulated in a field/value-pair message object as shown in Fig. 12.

The FIPA message contains the context metadata in the `:content` component. As shown in Fig. 12, the context metadata such as user name, age, and preference values on content genres, and so on, resides inside the `:content` component.

```
 (request
   :sender PC-Client@http://210.107.133.128:40
   :receiver Server@http://210.107.133.70:50
   :content
    ( <?xml version="1.0 encoding="UTF-8"?>
      <Person>
        <name>yumi</name>
          <age>23</age>
          <genre preferenceValue="30">Drama</genre>
          <genre preferenceValue="21">Sports</genre>
          <genre preferenceValue="30">Movie</genre>
          <genre preferenceValue="30">News</genre>
                          ⋮
      </Person>
          ⋮
    )
    :language SL10
    :protocol fipa-request
    :ontology fipa-agent-management
)
```

Fig. 12. Presentation of FIPA message.

2. Delivery and Processing of FIPA Messages

Before a message is sent from the sender agent to the destination agent, it must be processed through several transformations. The message which is constructed in the agent application level contains only the minimum information that is familiar to users or developers. But that information is not enough for the delivery mechanism between agents.

Figure 13 shows that the transformation needed to process messages starts from the agent application level and ends at the message transport level. At the agent application level, a message is constructed only with the simple information needed for user application purposes. Then, FIPA implementation needs to add additional information such as a

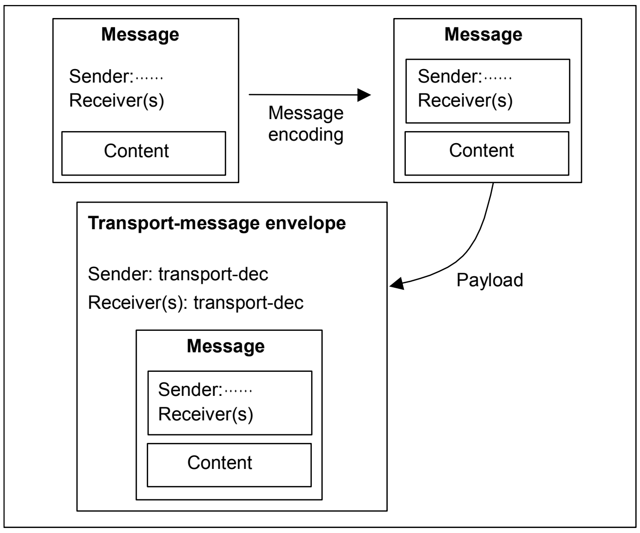


Fig. 13. Message transformation for delivery [8].

conversation ID, time-stamp, and so on, for its conversation control. Finally, for the purpose of communication efficiency, this message is packed into the payload, or payloads if needed, which have been optimized to the selected transport protocol such as HTTP or FIPAOS-RMI, and so on [8]. Reverse transformation will be applied to the message received at the destination site.

At the agent application level, a message is constructed only with the simple information needed for user application purposes. Then, FIPA implementation needs to add additional information such as a conversation ID or time stamp for its conversation control. Finally, for the purpose of communication efficiency, this message is packed into the payload which has been optimized to the selected transport protocol such as HTTP or FIPAOS-RMI [4]. Reverse transformation will be applied to the message received at the destination site.

3. Application Scenario

Figure 14 shows a sequence diagram of our application scenario. In the initial step, a client module sends to the server module an XDI metadata, which is the MPEG-21 DIA context information. The server module parses the XDI metadata, and then creates and sends to the client module the CDI metadata which carries information about titles, genres, sources, and so forth, for the contents available on the server side.

A user may select and request a preferred multimedia content from the server side to be displayed in the client GUI. The server derives the content-adaptation module to retrieve the requested CDI and adapts the retrieved original content, thus producing an adapted content according to the context information received from the client as a user's XDI. Then, the adapted content is delivered by the media streaming module to the client.

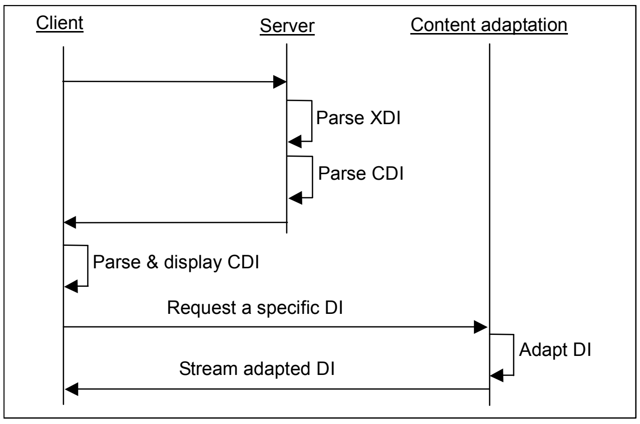


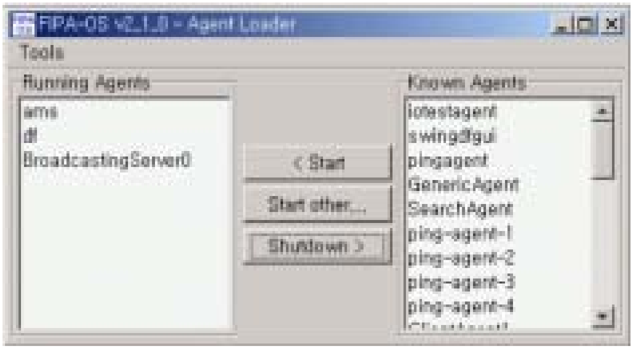
Fig. 14. Sequence diagram of application scenario.

4. System Implementation and Experiments

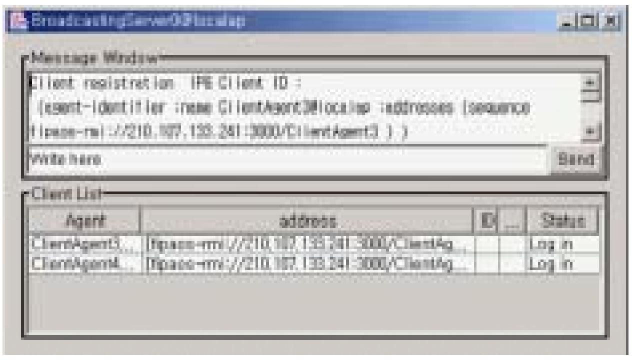
When a client agent is executed at a terminal, it connects to the BroadcastingServer agent which is activated by the server agent loader, as shown in Fig. 15(a). Notice that two other agents, *ams* (agent management system) and *df* (directory facilitator), are activated by the server agent loader in the left part of the display window in Fig. 15(a). The *ams* agent manages all the agents running on the server. The *df* agent acts like a telephone directory by providing a client agent with the possible service information from other agents.

Figure 15(b) shows a graphical user interface of a server that shows a list of the client agents connected to the server. Here, two clients are shown being served by the server agent.

The connection between a client agent and the broadcasting server agent is made by clicking a connect button in the GUI of the client terminal. At this time, the user preference data in the MPEG-21 XDI is transmitted to the broadcasting server agent in a FIPA message form. The broadcasting server agent parses the MPEG-21 XDI that includes the user preference data and then acquires from the metabase the relevant program content information conformed to the user preference. The tailored program content information is then delivered in an MPEG-21 CDI as a message to the client agent which will present it to the user.



(a) Agent loader at the server.



```
Client registration  IP6 Client ID :
 (agent-identifier :name ClientAgent3@localap :addresses (sequence
fipaos-rmi://210.107.133.241:3000/ClientAgent3 ) )
```

Agent	address	ID	...	Status
ClientAgent3...	fipaos-rmi://210.107.133.241:3000/ClientAg...			Log in
ClientAgent4...	fipaos-rmi://210.107.133.241:3000/ClientAg...			Log in

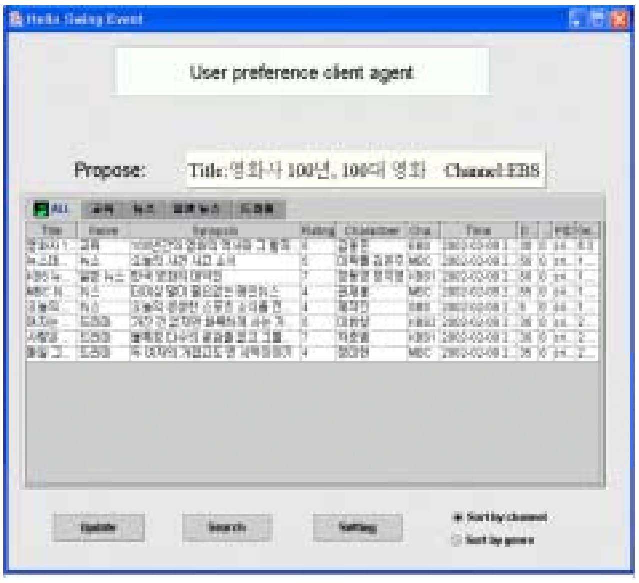
(b) Event window at the server.

Fig. 15. GUIs of a broadcasting server agent.

Figure 16 indicates an example of user preference within a particular genre. The preference on genre is indicated by the attribute “preferenceValue.” In Fig. 16, education is the favored genre with users and has a preference value of 87.

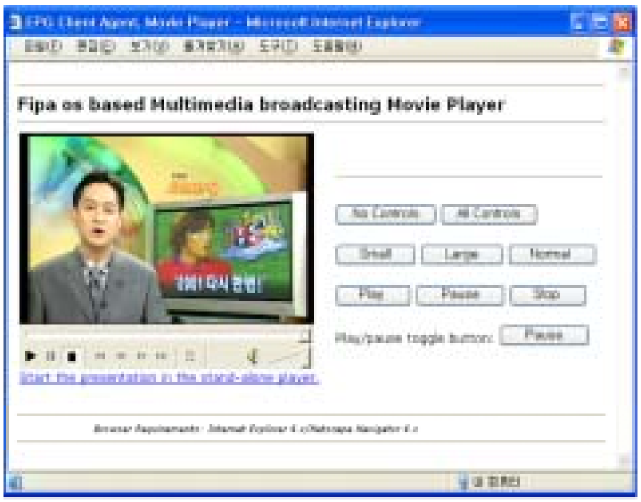
```
<?xml version=“1.0” encoding=“UTF-8”?>
<Mpeg7 xmlns= “http://www.w3.org/2000/XMLSchema-instance”
type=“complete”>
  <UserPreferences>
    <UserIdentifier protected=“true”>
      <UserName>icu</UserName>
    </UserIdentifier>
    <UsagePreferences allowAutomaticUpdate=“true”>
      <FilteringAndSearchPreferences protected=“true”>
        <ClassificationPreference>
          <Genre href=“urn:mpeg:GenreCS” preferenceValue=“62”>
            <Name>Movies</Name>
          </Genre>
          <Genre href=“urn:mpeg:GenreCS” preferenceValue=“87”>
            <Name>Education</Name>
          </Genre>
          <Genre href=“urn:mpeg:GenreCS” preferenceValue=“71”>
            <Name>Drama</Name>
          </Genre>
          <Genre href=“urn:mpeg:GenreCS” preferenceValue=“32”>
            <Name>News</Name>
          </Genre>
          <Genre href=“urn:mpeg:GenreCS” preferenceValue=“14”>
            <Name>Music</Name>
          </Genre>
                ..................
        </ClassificationPreference>
      </FilteringAndSearchPreferences>
    </UsagePreferences>
  </UserPreferences>
</Mpeg7>
```

Fig. 16. An example of user preference description data.



User preference client agent
Propose:
Title:영화사 100년, 100대 영화 Channel:EBS
ALL
Update
Search
Setting
Sort by channel
Sort by genre

Fig. 17. Graphical user interface at the client terminal.



Fipa os based Multimedia broadcasting Movie Player
No Controls
All Controls
Small
Large
Normal
Play
Pause
Stop
Play/pause toggle button:
Pause

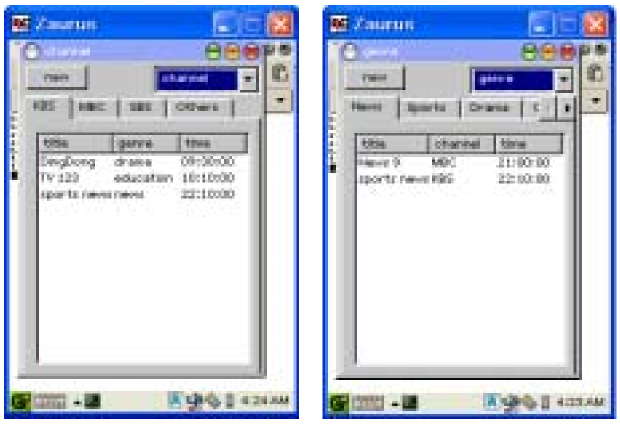
Fig. 18. The user interface to watch program.

Based on the preference values for genre in the user preference data, the program information is presented in the GUI, as shown in Fig. 17. The program information for the most and least favorite genres is displayed from the top to the bottom in the ALL tab menu in the figure. The program content information is also presented in groups starting with the most favored genre, which is presented by the menu tab just to the right of the ALL menu tab, followed by the lesser favored genre tabs, presented in order of preference.

Users can watch the program, which is selected in the program list via the client GUI at the client terminal. The client agent sends a request for the selected program to the broadcasting server agent and then receives multimedia streaming data from the streaming server.

In this implementation we used Windows Media Player for browsing the program contents being streamed from the streaming server, as shown in Fig. 18.

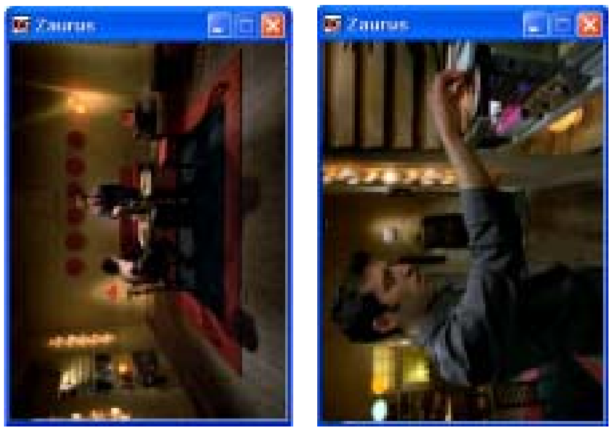
Figure 19 shows graphical user interfaces at a client terminal with a user terminal, PDA. The left GUI in Fig. 15(a) shows the user preferred program list in terms of program sources (TV channels), and the right GUI lists the user preferred programs into program genres. By presenting the tailored program information at the client side, a user can easily navigate and access his or her program contents of interest. When the user finds an interesting program and clicks on it through the GUI, the client agent sends a request signal with the usage characteristics description as an XDI, including the terminal characteristics to the server agent. Then, the corresponding program contents are streamed from the streaming server on the server side according to the decoding and display capabilities of the client terminal. Figure 19 (b) exhibits streamed user-selected program contents.



title	genre	time
DingDong	drama	09:30:00
TV 123	education	10:10:00
sports news	news	22:10:00

title	channel	time
News 9	MBC	21:00:00
sports news	KBS	22:00:00

(a) Program information list based on user preferences on TV channels and program genres.



(b) User-selected program contents being streamed.

Fig. 19. TV program viewing on a graphical user interface of a PDA.

VII. Conclusion

In this paper, we present an agent-based intelligent multimedia broadcasting framework. We use FIPA-OS as a platform for exchanging the MPEG-21 XDI of a usage characteristics description and the MPEG-21 CDI of user preferred broadcast program information as FIPA messages between the agents. The user preference is especially modeled as the User Preference description scheme in an MPEG-7 MDS.

The communications between the server and clients are made using ACL. The adoption of the agent platform can provide a flexible framework in providing intelligent broadcasting services. We expect that more sophisticated tasks can be easily developed for specific dedicated processing within the IBS framework. We believe that the IBS framework is well suited for the MPEG-21 Multimedia Framework environment by considering user, terminal and network characteristics under multimedia environments of various types and will provide a platform for target-oriented intelligent broadcasting services such as personal casting.

References

[1] Munchurl Kim, Geewoong Ryu, Beetnara Bae, Jeho Nam, Kyoungok Kang, ang Jinwoong Kim, "Intelligent Program Guide for Digital Broadcasting," *Proc. Int'l Workshop Advanced Image Technology*, Jan. 16-19, 2002, Hualien, Taiwan, pp. 257-263.

[2] Seongjoon Pak, Geewoong Ryu, and Munchurl Kim, "Agent-Based Multimedia Personalcasting," *Proc. Int'l Workshop Advanced Image Technology*, Jan. 21-22, 2003, Nagasaki, Japan, pp. 311-316.

[3] Mark T. Maybury, "Personalcasting: Tailored Broadcasting News," *Proc. 1st Workshop Personalization in Future TV*, Sonhofen, German, 2001, pp. 39-41.

[4] Nortel Network, FIPA-OS v.2.1.0 Distributions Notes, http://fipa-os.sourceforge.net/.

[5] ISO/IEC JTC1/SC29/WG11, *ISO/IEC 21000-10 FDIS: Digital Item Adaptation*, MPEG2003/N6168, Hawaii, USA, Dec. 2003.

[6] ISO/IEC JTC1/SC29/WG11, *ISO/IEC 21000-1 PDTR Second Edition: Vision, Technology and Strategy*, MPEG2003/N6269, Hawaii, USA, Dec. 2003.

[7] ISO/IEC JTC1/SC29/WG11, *ISO/IEC 15938-5 FDIS: Multimedia content Description Interface*, MPEG2001/N4242, Sydney, Australia , July 2001.

[8] FIPA, http://www.fipa.org/.

[9] *FIPA-OS Developers Guide*, http://sourceforge.net/projects/fipa-os/.

[10] *MicroFIPA-OS User Guide*, http://www.cs.helsinki.fi/group/crumpet/.

[11] ISO/IEC JTC1/SC29/WG11, *ISO/IEC 21000-2 FDIS: Digital Item Declaration*, MPEG2002/N4813, Fairfax, USA, May 2002.

[12] FIPA TC Architecture, *FIPA Agent Message Transport Service Specification*, XC00067D, Aug. 2001.

[13] FIPA TC Architecture, *FIPA ACL Message Structure Specification,* XC00061E, Aug. 2001.

[14] Jae-Gon Kim, Hyun Sung Chang, Young-tae Kim, Kyeongok Kang, Munchurl Kim, Jinwoong Kim, and Hyung-Myung Kim, "Multimodal Approach for Summarizing and Indexing News Video," *ETRI J.*, vol. 24, no. 1, Feb. 2002, pp. 1-11.

Munchurl Kim received the BE degree in electronics from Kyungpook National University, Korea in 1989, and the ME and PhD degrees in electrical and computer engineering from University of Florida, Gainesville, USA, in 1992 and 1996. After his graduation, he joined the Electronics and Telecommunications Research Institute (ETRI) where he worked in the MPEG-4/7 standardization-related research areas. Since 2000, he has been involved in the project management for the development of a data broadcasting end-to-end system for terrestrial digital broadcasting. In 2001, he joined, as Assistant Professor of School of Engineering, the Information and Communications University (ICU) in Daejeon, Korea. His research areas of interest include MPEG-4/7/21, video compression/communications, intelligent and interactive multimedia, and pattern recognition.

Jeongyeon Lim received the BE and ME degrees in information and communications engineering from Chungnam National University, Korea in 1999, and 2001. Since 2001, she has been a PhD student in Information and Communications University (ICU), Korea. Her research areas of interest include intelligent and interactive multimedia, multimedia information processing, and MPEG-4/7/21.

Kyeongok Kang received the BS and MS degrees in physics from Pusan National University in 1985 and 1988. Also, he received the PhD degree in electrical engineering at Hankuk Aviation University in 2004. He has been in ETRI since 1991, and he is now a Senior Member of the Engineering Staff and a leader of the 3D Media Research Team. His major interests are in personalized broadcasting technology based on MPEG-7 and TV-Anytime, and three dimensional audio processing technology.

Jinwoong Kim received the BS and MS degrees in electronics engineering from Seoul National University, Korea in 1981 and 1983, and the PhD degree in electrical engineering from Texas A&M University, USA in 1993. Since 1983, he has been on the research staff in the Electronics and Telecommunications Research Institute (ETRI), Korea, working for the development of a TDX digital switching system, MPEG-2 video encoder, HDTV encoder system, MPEG-7 technology and data broadcasting system. He is currently Director of the Broadcast Media Technology Group. His research interests include digital signal processing in the fields of video communications, multimedia systems, and interactive broadcast systems.