일반논문-06-11-1-10

데이터방송 멀티 애플리케이션 개발

김 현 순[a)‡], 권 재 광[a)], 강 대 갑[a)]

Development of Multi-applications for Data Broadcasting

Hyun-Soon Kim[a)‡], Jae-Kwang Kwon[a)], Dae-Kap Kang[a)]

요 약

단일 애플리케이션 송출 환경에서의 데이터방송에서는 특정 시간, 특정 채널에 대해 하나의 애플리케이션만을 서비스할 수 있었다. 이러한 단점을 극복하기 위하여 본 논문에서는 ACAP(Advanced Common Application Platform) 규격을 따르는 범용의 멀티 애플리케이션 구조/운용 방법 개발 및 이를 지원하기 위한 기존 데이터방송 시스템의 개선을 제안한다. 멀티 애플리케이션 서비스가 제공되면 방송사가 특정 시간, 특정 채널에서 하나 이상의 애플리케이션을 자유롭게 편성하여 서비스하므로 사용자는 다양한 애플리케이션을 기호에 맞게 선택, 이용할 수 있다. 본 논문에서는 실험용 멀티 애플리케이션 개발 및 실험을 통하여 제안한 멀티 애플리케이션 구조를 검증하고 추후 본 방송 적용에 적합한 운용 방법을 제시한다. 애플리케이션 측면에서 멀티 애플리케이션의 핵심은 일반 애플리케이션들에 대한 관리를 담당하는 매니저(혹은 루트) 애플리케이션이므로 이에 대한 기능 및 구조에 대하여 먼저 기술하고, 송출 시스템 및 수신기와의 정합을 성공적으로 마친 실험용 멀티애플리케이션에 대한 실험 결과를 통하여 제안한 방식이 적합한 멀티 애플리케이션 모델임을 보인다.

Abstract

In the environment of single application data broadcasting, only one application can be serviced at a specific time on one channel. To overcome this, we developed the structure and the method of operation for multi-applications which are fully conformant to the ACAP (Advanced Common Application Platform), and modified data broadcasting system to support multi-applications. In multi-application environment, broadcasters can service multiple applications simultaneously at a specific airtime on one channel so users can enjoy services selectively according to their preferences. In this paper, we present an example of multi-application service which was developed to make an experiment before servicing them to users on the air. The core of the multi-application is a manager application which manages other ordinary applications, so we describe the function and structure of the manager application, and then present the experimental results to show that the proposed method is the proper model for multi-applications.

Keyword: data broadcasting, multi-application, manager application.

I. 서 론

방송의 디지털화로 TV를 보면서 경매, 게임, 물품 구매, 부가 정보 보기 등의 데이터방송 서비스를 사용할 수 있게 되었다. 지상파 데이터방송 표준인 ATSC(Advanced Television Systems Committee)-ACAP[1,2]이 표준으로 확정됨에 따라, 지상파 방송사들은 실험 방송을 통하여 수신기 및 방송 송출 시스템에 대한 기술적인 정합실험을 실시

a) 한국방송 방송기술연구팀
Broadcast Technical Research Team, Korean Broadcasting System(KBS)
‡ 교신저자 : 김현순 (soon71@kbs.co.kr)

하고 다양한 애플리케이션을 서비스하면서 본 방송 대비에 박차를 가하고 있다[3].

그러나 대부분 특정 시간, 특정 채널에 대해 하나의 애플리케이션만을 서비스하고 있거나 방송사에서 하나 이상의 애플리케이션을 송출하고 있다고 하여도 특정 수신기에서는 그 중 하나만 수신되어 서비스되는 등으로 인하여 멀티 애플리케이션에 대한 필요성이 대두되고 있다.

즉 현재 지상파 방송사들의 데이터방송 실험 방송에는 애플리케이션 측면에서 멀티 애플리케이션이라는 개념이 도입되지 않아, 여러 개의 애플리케이션을 실험 서비스하는 방송사에서도 단순히 일반 단일 애플리케이션 여러 개를 동일 시간, 동일 채널에 송출하는 형태로 서비스하고 있다. 이 경우, 단일 애플리케이션 여러 개를 서비스하는 방송사는 사용자가 원하는 애플리케이션을 선택하여 시청할 것을 기대하지만, 수신기의 종류에 따라 그 서비스를 수행하지 않을 수도 있다. 국내 특정 수신기의 경우 여러 개의 애플리케이션이 송출되고 있을 때 이들의 이름을 보여주는 단순한 애플리케이션 리스트를 보여주고 사용자가 선택하여 실행할 수 있도록 하고 있는 경우도 있지만, 하나의 애플리케이션만을 수신하여 보여주는 수신기를 가지고 있는 사용자는 다수의 애플리케이션들이 서비스되고 있는지조차 알 수 없을 것이다.

만약 모든 수신기에서 애플리케이션 리스트 기능을 제공한다고 해도 수신기마다 리스트를 보여주는 방법, 구체적인 처리 방법 등이 달라 사용자에게 혼란을 초래할 것이다. 또한 수신기가 제공하는 애플리케이션 리스트 UI(User Interface)는 애플리케이션 이름들을 나열하여 선택하도록 하는 정도의 수준에서 크게 벗어나지 못한다. 그러므로 방송사는 자사의 특성을 반영하여 다양한 디자인의 애플리케이션 리스트를 제공하고 개편 등의 특정 시점에 변경된 디자인을 반영할 수 있는 새로운 방법을 필요로 한다.

본 연구에서는 국제 규격에 위배되지 않는 멀티 애플리케이션 구조/운용 방법을 제안하고, 본 방송에 대비하여 개발한 실험용 멀티 애플리케이션을 통하여 제안한 방법을 검증하고 적합한 서비스 모델을 제시한다. 본 연구에서 제안한 멀티 애플리케이션은 단순히 하나 이상의 애플리케이션을 동시에 제공하는 송출 측면에서의 방법과 달리 애플리케이션 측면에서 구현한 것이므로 다양한 수신기에 대하여 동일한 결과로 멀티 애플리케이션 서비스가 가능하도록 한다.

Ⅱ. 멀티 애플리케이션 개발

1. 멀티 애플리케이션 기능 개요

애플리케이션 측면에서 멀티 애플리케이션 서비스 환경을 제공하기 위한 핵심은 매니저 애플리케이션이다. 멀티 애플리케이션 서비스를 제공하기 위하여 방송사는 기존에 서비스하던 형태의 일반적인 애플리케이션들을 관리하기 위한 매니저 애플리케이션도 함께 송출한다. 매니저 애플리케이션은 현재 사용 가능한 일반 애플리케이션들에 대한 리스트를 화면에 표시하고 이들의 라이프사이클을 관리하는 기능을 하는 특수한 형태의 애플리케이션이다. 즉 매니저 애플리케이션은 다른 애플리케이션들을 시작, 종료, 중단, 파괴시키는 등의 역할을 수행한다. 실험용 멀티 애플리케이션 제작 경험을 바탕으로 한 매니저 애플리케이션의 주요 기능은 아래와 같다.

■ **'애플리케이션 지정자' 기능**

데이터방송 애플리케이션은 A/V와 함께 전송되는 부가 데이터이므로, A/V에 대한 시청을 최대한 방해하지 않아야 한다. 이를 위하여 데이터방송이 서비스되고 있을 때, 이를 이용할지 여부를 시청자가 선택할 수 있도록 데이터방송 애플리케이션이 서비스되고 있다는 지정자가 필요하다. 지정자는 최대한 A/V 시청을 방해하지 않고 시청자가 쉽게 리모콘의 특정 키를 눌러 애플리케이션을 이용할 수 있도록 제작되어야 한다.

■ **'애플리케이션 리스트'를 화면에 표시 및 갱신**

매니저 애플리케이션은 데이터방송 송출 시스템이 수신기로 송출하는 일반 애플리케이션들에 대한 정보를 수신기 미들웨어가 제공하는 API(Application Programming Interface)를 이용하여 읽어 화면에 표시해야 한다. 또한

송출 시스템이 특정 애플리케이션을 중단시키거나 새로운 애플리케이션 서비스를 시작할 경우, 수신기 미들웨어의 애플리케이션 변경 이벤트를 감지하여 애플리케이션 리스트를 갱신하여야 한다.

■ **일반 애플리케이션에 대한 제어 기능**

사용자가 특정 일반 애플리케이션을 선택할 경우 매니저 애플리케이션은 해당 일반 애플리케이션을 실행시키고 자기 자신은 pause 상태로 되어야 한다. 이 외에도 일반 애플리케이션 실행에 실패했을 때의 처리 등 애플리케이션 실행에 대한 제어 기능을 수행해야 한다.

■ **리모콘의 exit 및 애플리케이션 리스트 키 등 특정 키에 대한 처리**

사용자가 데이터방송 서비스를 사용하는 중에 리모콘에 하드웨어적으로 들어가 있는 exit 키 및 애플리케이션 리스트 키 등을 눌렀을 경우의 처리를 해 주어야 한다. 예를 들어, 시청자가 특정 데이터방송 애플리케이션 서비스를 사용하는 중에 리모콘의 exit 키를 누르면 매니저 애플리케이션은 해당 서비스 실행을 종료하여야 한다.

이러한 기능들을 가능하게 하기 위해서 방송사에서는 매니저 애플리케이션의 'control code', 'visibility' 값을 'autostart', '00'으로 각각 셋팅하여 송출하고, 일반 애플리케이션에 대한 이들 값을 'present', '11'로 각각 셋팅하여 송출한다. 수신기는 'autostart'인 매니저 애플리케이션을 수신하자마자 바로 실행시킨다. 수신기에 의해 자동 실행된 매니저 애플리케이션은 현재 실행 가능한 'present' 애플리케이션들에 대한 정보(애플리케이션에 대한 이름, ID, priority, visibility, control code)에 대한 정보를 읽어와 그들에 대한 리스트를 화면에 표시하고 관리를 시작한다. 일반 애플리케이션들의 'visibility' 값은 '11'로 셋팅되어 있으므로 스크린을 통해 사용자에게 보여지고, 매니저 애플리케이션도 애플리케이션 리스팅 API를 통하여 이들을 감지할 수 있다.

AIT(Application Information Table)가 'control code', 'visibility', '애플리케이션 이름', '애플리케이션 ID' 등 현재 사용 가능한 애플리케이션들에 대한 모든 필요한 정보를 수신기에 전달한다. 데이터 인코더가 이러한 값들을 AIT에 셋팅하여 수신기로 보내면 매니저 애플리케이션은 수신기가 제공하는 AppsDatabase, AppAttributes 등의 자바 API를 사용하여 읽어 온다.

그림 1은 매니저 애플리케이션의 일반 애플리케이션들에 대한 관리 기능을, 그림 2는 매니저 애플리케이션에 의한 애플리케이션 리스트 화면을 보여 준다. 그림 1, 2에 도시된 바와 매니저 애플리케이션은 서비스 중인 일반 애플리케이션들의 리스트를 화면에 제공하고 일반 애플리케이션에 대한 네비게이션 기능을 제공한다. 사용자는 화면에 제공된 애플리케이션들의 리스트에서 원하는 애플리케이션을 선택하여 자신이 원하는 서비스를 이용하게 된다.

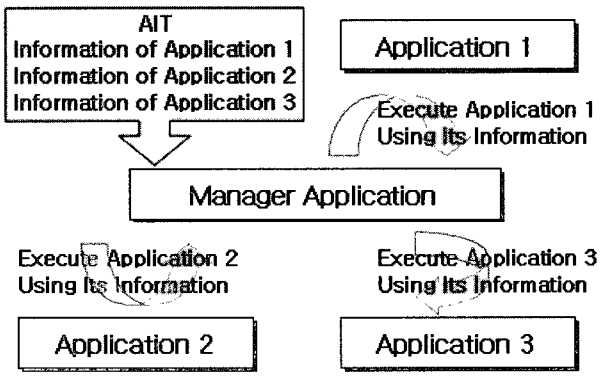


그림 1. 매니저 애플리케이션에 의한 일반 애플리케이션 실행
Fig. 1. Execution of applications by a manager application

그림 2에서 사용자는 리모콘의 상/하 키를 이용하여 네비게이션하다가 원하는 애플리케이션에 포커스가 오면 선택 키를 눌러 원하는 애플리케이션을 실행시킨다. 사용자에 의해 특정 일반 애플리케이션이 선택/실행되면 매니저 애플리케이션은 자신이 점유하고 있던 화면과 포커스를 해당 일반 애플리케이션에게 할당하고 자신을 화면에서 숨긴다. 사용자가 일반 애플리케이션을 사용하다가 특정 메뉴를 이용하여 매니저 애플리케이션을 실행시키거나, 리모콘의 exit 버튼을 이용하여 애플리케이션에서 빠져나가는 이벤트가 발생하면 해당 애플리케이션은 이러한 이벤트 정보를 수신기에 전달하고 매니저 애플리케이션은 수신기로부터 이들 이벤트를 받는 즉시 해당 일반 애플리케이션을 화면에서 지운 후 자기 자신을 실행시킨다.

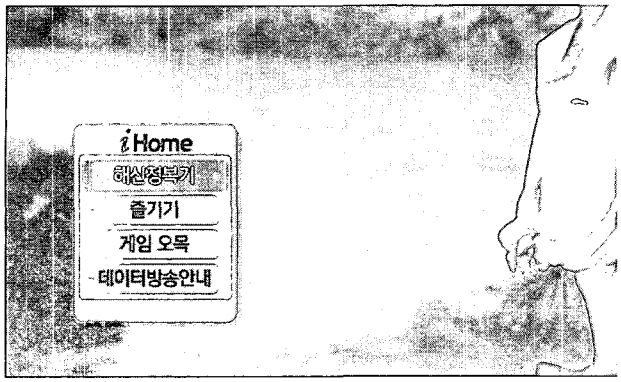


그림 2. 매니저 애플리케이션에 의한 애플리케이션 리스트 표시
Fig. 2. The list of applications displayed by a manager application

지금까지 매니저 애플리케이션을 중심으로 하여 멀티 애플리케이션의 기능을 살펴보았다. 지금까지 살펴본 멀티 애플리케이션 실행의 원리는 TV 화면상에 하나의 애플리케이션만 보인다는 전제 하에서 살펴본 것이다. 즉 PC의 윈도우즈 OS 환경 하에서처럼 화면에 동시에 여러 개의 애플리케이션들을 실행하는 것이 아니라, 특정 시간에 하나의 애플리케이션만을 디스플레이하는 경우이다. 이는 현 수신기가 리소스 사용 등에 대한 제약으로 인하여 한 화면에 하나의 애플리케이션만을 디스플레이하도록 구현되어 있기 때문이다.

이러한 제약으로 인하여 매니저 애플리케이션이 화면을 점유하여 애플리케이션 리스트를 표시하고 있다가, 사용자가 특정 애플리케이션을 선택하면 해당 애플리케이션이 화면에 표시되고 매니저 애플리케이션이 표시하는 애플리케이션 리스트 화면은 pause 상태로 되어 화면에서 사라지게 된다. 실행하던 애플리케이션에서 빠져 나와 다른 애플리케이션에 대한 서비스를 사용하고자 할 경우, 사용자는 현재 사용하던 애플리케이션 UI 상에서 제공하는 매니저 애플리케이션 실행 버튼을 눌러 매니저 애플리케이션의 애플리케이션 리스트 화면이 디스플레이 되도록 한 다음, 리스트에서 원하는 애플리케이션 다시 선택하여 새로운 애플리케이션을 실행시켜야 한다. 즉, 화면상에 표시되는 애플리케이션은 하나이므로 다른 애플리케이션간의 이동은 매니저 애플리케이션을 경유한다.

만약 한 화면에 하나의 애플리케이션만 표현한다는 수신기의 제약 사항이 추후 제거된다면 조금 더 다양한 형태로 멀티 애플리케이션을 서비스할 수 있을 것이다. 즉 한 화면에 동시에 여러 개의 애플리케이션을 실행시키거나, 화면에 active 상태인 애플리케이션이 실행되고 있을 때 pop-up 형태의 애플리케이션을 긴급 송출하여 active 상태 애플리케이션 위에 디스플레이 하는 등 다양한 시나리오가 가능할 것이다.

2. 매니저 애플리케이션 구조 및 실행 원리

매니저 애플리케이션은 일반 애플리케이션 각각에 대한 정보를 수신기로부터 읽어와 이들 정보를 이용하여 각각의 애플리케이션들을 구분하고 관리한다. 예를 들어 'organization ID', 'application ID' 등은 일반 애플리케이션 각각에 대하여 고유하게 할당되어 송출되므로, 매니저 애플리케이션은 이들을 수신기로부터 읽어와 애플리케이션들을 구분한다. 그림 3에서 매니저 애플리케이션이 이들 정보를 수신기로부터 획득하는 과정을 나타내었다.

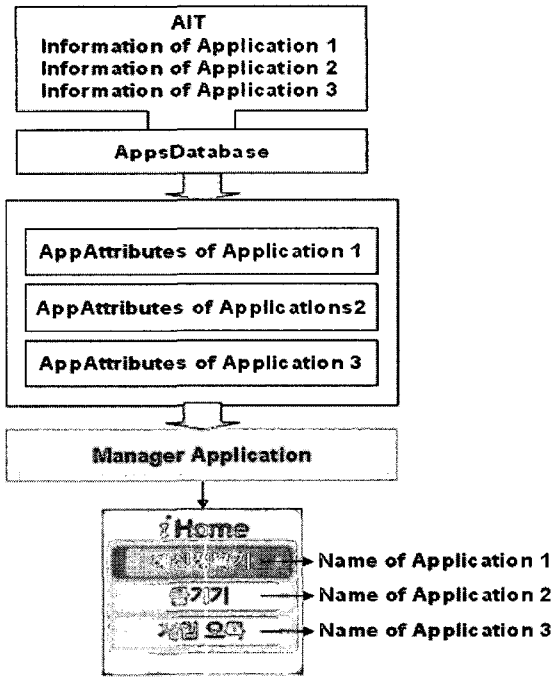


그림 3. 애플리케이션 정보 획득 과정
Fig. 3. Acquisition of the information for applications

그림 3에서 알 수 있듯이, 매니저 애플리케이션은 이들을

획득하기 위하여 수신기가 제공하는 AppAttributes, AppsDatabase 등을 포함하는 org.dvb.application.* 패키지 내의 API들을 이용한다[4]. AppsDatabase는 수신기에서 제공하는 애플리케이션들의 정보를 관리하는 자바 객체로, 매니저 애플리케이션은 AIT의 애플리케이션들에 대한 정보를 AppsDatabase를 사용하여 얻는다.

아래 소스는 애플리케이션 정보 획득 과정에 대한 일부 소스 코드이다. 매니저 애플리케이션은 AppsDatabase가 제공하는 getAppAttributes(), getAppProxy(AppID)를 통하여 각각의 애플리케이션에 대한 AppAttributes 객체 및 AppProxy 객체를 얻는다. 매니저 애플리케이션은 AppAttributes 객체를 얻은 후 이 객체의 함수들을 사용하여 애플리케이션 정보를 읽어 온다. AppsAttributes 객체는 getIdentifier(), getName(), getPriority() 등의 함수를 제공하는데, 각각의 애플리케이션에 대한 AppID, 이름, 우선순위를 반환한다.

AppID 객체는 애플리케이션에 대한 고유한 아이디를 표현하기 위한 객체이다. 아래 소스 코드에서 알 수 있듯이,

```
if(app_database == null)
{
  app_database = AppsDatabase.getAppsDatabase();
}
CurrentServiceFilter cur_filter = new CurrentServiceFilter();
if((app_database != null) && (app_database.size() != 0))
{
  Enumeration attributes = app_databse.getAppAttributes(cur_fil
  ter);
  if(attributes != null)
  {
    while(attributes.hasMoreElemetns())
    {
      info = (AppAttributes)attributes.nextElement();
      if(info != null && info.getIsServiceBound() &&
info.isVisible())
      {
        app_id = info.getIdentifier();
        aid = app_id.getAID();
        oid = app_id.getOID();
        app_proxy = app_database.getAppProxy(app_id);
        try{
          app_name = new String(info.getName().getBytes());
        }catch(Exception e){
          e.printStackTrace();
        }
```

매니저 애플리케이션은 최종적으로 이 객체의 getAID(), getOID() 함수를 이용하여 애플리케이션들을 구분하고 이들 애플리케이션들을 대리할 AppProxy를 얻기 위한 매개변수로 사용한다. 요약하면 매니저 애플리케이션은 AppsDatabase에서 AppAttributes 객체를 얻고 AppAttributes 객체로부터 AppID를 포함한 정보를 얻어 이들 정보를 이용하여 화면에 애플리케이션 리스트를 표시하고 관리를 한다.

화면에 표시된 애플리케이션 리스트 중에서 하나를 시청자가 선택하여 실행할 경우, 매니저 애플리케이션은 해당 애플리케이션이 시작되도록 한다. 그림 4에서 세 개의 일반 애플리케이션 중 'application 1'을 사용자가 선택/실행할 경우에 해당 애플리케이션이 실행되는 과정을 나타내었다. 그림 4에서와 같이 매니저 애플리케이션은 사용자가 선택한 애플리케이션을 실행하기 위하여 AppProxy라는 객체를 사용한다.

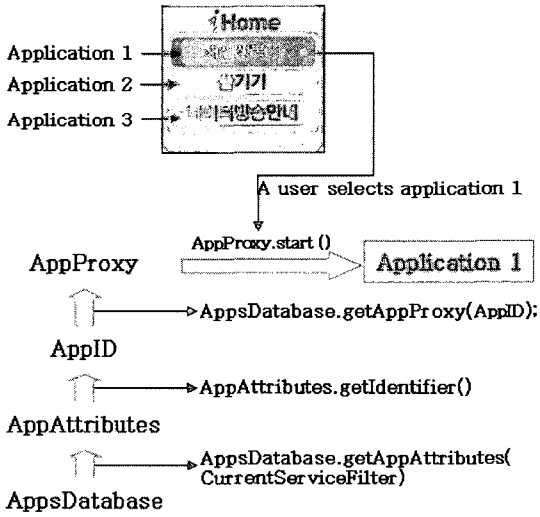


그림 4. 애플리케이션 시작 과정
Fig. 4. Steps for starting an application

위 소스 코드에서 살펴본 바와 같이, AppsDatabase, AppAttributes, AppID를 순서대로 얻어 온 후 최종적으로 AppsDatabase.getAppProxy(AppID)를 통하여 AppProxy를 미리 얻어 놓고, 사용자가 특정 애플리케이션을 실행시키면 즉시 AppProxy.start()를 호출하여 해당 애플리케이션

을 시작시킨다. AppProxy는 특정 애플리케이션의 동작 상태를 제어하는 proxy 역할을 하도록 수신기에서 제공하는 애플리케이션 대리자(Proxy)이다. AppProxy는 AIT에 시그널 되는 각 애플리케이션 엔트리와 일대일 대응 관계를 가진다. AppProxy는 해당 애플리케이션을 시작, 종료, 일시 중지시키는 등의 함수를 제공한다.

매니저 애플리케이션은 일반 애플리케이션의 상태 변화를 감지하고 처리하기 위하여 AppStateChangeEvent-Listener를 구현하여, 애플리케이션 상태가 변하면 수신기가 발생시키는 AppStateChangeEvent를 처리한다. 매니저 애플리케이션은 AppProxy 객체 addAppStateChange-EventListener(AppStateChangeEventListener listener) 함수를 이용하여 상태 변화를 감지한다. App- StateChange-EventListener의 stateChange(AppStateChange- Event evt) 함수는 일반 애플리케이션의 상태가 변함을 감지하고, 상태 변화가 비정상적으로 이루어질 경우 매니저 애플리케이션이 자기 자신을 다시 실행시켜 애플리케이션 리스트 화면을 실행하도록 구현되었다.

송출 시스템에서 현재 서비스 중인 애플리케이션을 중단시키거나 새로운 애플리케이션을 송출할 경우, 매니저 애플리케이션은 이들 이벤트를 처리한다. 이를 위하여 매니저 애플리케이션은 AppsDatabaseEventListener를 구현하여 AppsDatabase 내용이 변할 경우 수신기가 발생시키는 AppsDatabaseEvent를 처리한다. 현재 서비스 중인 애플리케이션 엔트리 중 하나가 서비스 중단되거나 변경될 때, 엔트리가 새로 추가될 때 수신기는 AppsDatabaseEvent를 발

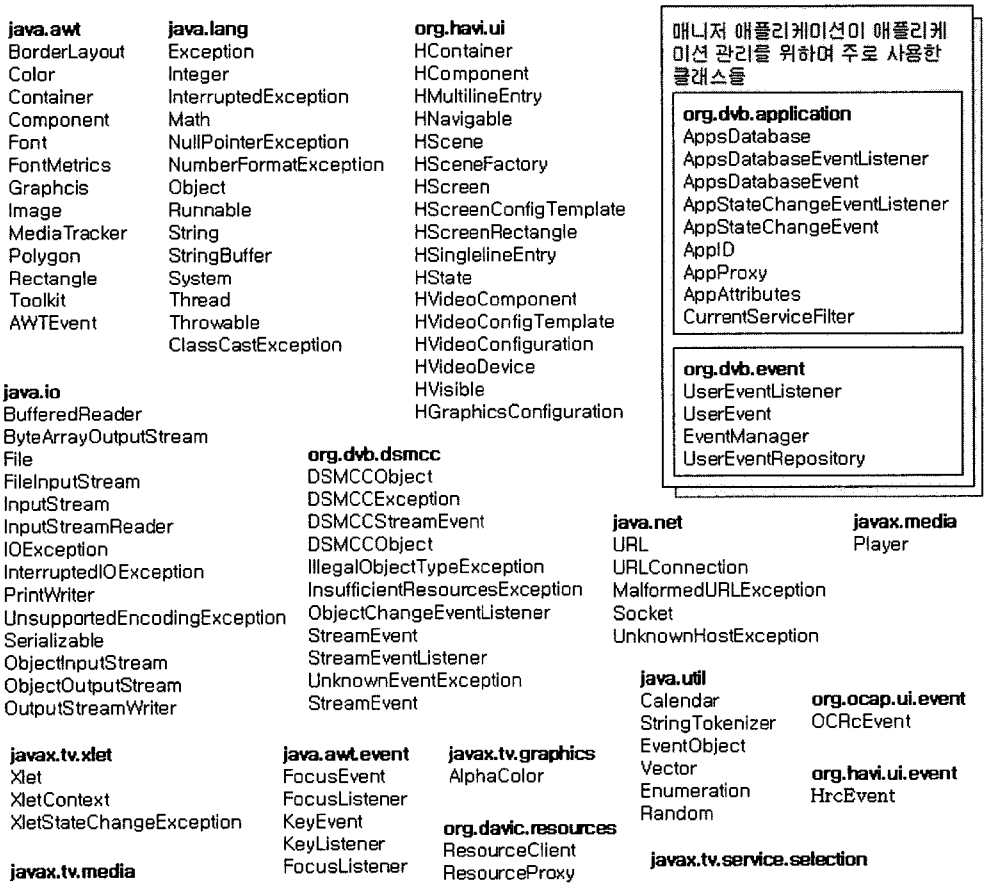


그림 5. 애플리케이션 제작에 사용한 클래스들
Fig. 5. Classes used for applications

생시킨다. 매니저 애플리케이션은 AppsDatabaseEvent가 발생하면 자신이 가지고 있는 애플리케이션들에 대한 모든 정보 및 화면에 표시한 애플리케이션 리스트를 갱신한다.

기존의 단일 애플리케이션 구현과 비교하여, 멀티 애플리케이션 구현 시 org.dvb.application.* 패키지 내의 자바 객체 등을 추가적으로 사용하였다. 애플리케이션 제작에 주로 사용한 객체들을 그림 5에서 정리하였다.

Ⅲ. 실험 결과

본 연구에서는 실험용 멀티 애플리케이션을 개발하여 멀티 애플리케이션 고유 기능에 대한 동작 실험뿐만 아니라, 송출 시스템 및 수신기의 멀티 애플리케이션 지원 기능에 대한 정합 실험을 실시하였다. 멀티 애플리케이션은 다른 일반 애플리케이션들을 관리하는 매니저 애플리케이션이라 이름붙인 특수한 콘텐츠를 포함하여 총 6종의 다양한 콘텐츠로 구성되었다. 이렇게 다양하게 콘텐츠를 구성한 이유는 단방향 콘텐츠 동작, 양방향 콘텐츠 동작 등 기존의 데이터방송 기본 기능에 대한 정합을 강화하고 '멀티 애플리케이션 서비스', '리턴채널 데이터방송 송출[5], '고객 반응분석 시스템 정합' 등 고도화된 신규 기능에 대한 검증을 가능하게 하기 위함이다.

총 6종 콘텐츠 및 이들을 이용한 주요 정합 실험 항목은 아래와 같다.

■ **'매니저 애플리케이션'**
- 두 개 이상의 애플리케이션을 송출하고 제어하는 기능을 실험하여 데이터방송 시스템의 멀티 애플리케이션 송출 기능 점검
- 데이터방송 콘텐츠가 있음을 사용자에게 알리는 지정자
- 데이터방송 리스트 화면
- 5종의 일반 애플리케이션에 대한 제어 기능
- 사용자가 리모콘의 exit와 애플리케이션 리스트 버튼을 눌렀을 경우의 처리
- 사용자가 매니저 애플리케이션이 표시하고 있는 서브 애플리케이션 리스트 중의 하나를 눌렀을 때, 서브 애플리케이션이 실행될 때까지 로딩 중이라는 메시지를 표시하는 로딩바 기능
- 매니저 애플리케이션 개발에 추가 사용된 다양한 자바 클래스에 대한 정합 실험(주로 org.dvb.application.* 패키지 내의 클래스들임)

▣ **'해신 정복기' 애플리케이션 개발**
- 퀴즈 참여, 상품 구매 등을 통한 송출 시스템 및 수신기에 대한 기존의 양방향 기능 점검 강화
- 수신기 시간 정보를 이용하여 계산한 이용 시간 정보 등을 이용한 시청자 반응분석 시스템 검증 실험
- 해신 정복기의 일부 메뉴에 보안 소켓을 적용하여 콘텐츠-리턴시스템-수신기간의 보안 기능 점검

▣ **'오목' 게임, '퍼즐' 게임, '데이터방송안내'**
- 단방향 애플리케이션에 대한 기존 기능 점검 강화

▣ **'즐기기' 애플리케이션 개발**
- 대용량 데이터를 리턴채널로 전송하기 위한 데이터방송 시스템 및 수신기 기능 실험
- 대용량 데이터 전송의 효과를 점검하기 위한 풍부한 이미지와 텍스트로 구성된 메뉴로 구성, 실험

이러한 데이터방송 시스템-콘텐츠-수신기 정합 실험을 통하여 기존 데이터방송 기능 및 올해 추가된 멀티 애플리케이션 서비스 지원 등 고도화 기능의 안정성을 실험하였다. 이를 통하여 콘텐츠 제작에 사용한 주요 자바 클래스에 대한 정상 동작 유/무 확인을 포함하여 다양한 기술 사항들을 점검할 수 있었다. 이러한 점검의 결과로 지상파 데이터방송 송수신 정합 가이드라인 및 TTA 표준안 작성, 시스템의 안정화 및 고도화를 위한 실험 방송에 기여할 수 있었다.

매니저 애플리케이션은 '애플리케이션 지정자', '애플리케이션 리스트'의 두 가지 UI를 가진다. 그림 6, 7에서 이들을 각각 나타내었다. 애플리케이션 지정자는 현재 데이터방송이 서비스되고 있음을 알리는 지정자로, A/V 화면 시

그림 6. 애플리케이션 지정자
Fig. 6. Application indicator

그림 7. 애플리케이션 리스트
Fig. 7. The list of applications

청을 최대한 방해하지 않고 시청자가 데이터방송 서비스 이용 방식에 익숙해질 수 있도록 하기 위하여 15초 간격으로 나타났다가 사라지기를 반복한다. 애플리케이션 리스트는 지정자 화면에서 리모콘의 레드 컬러 키를 선택하면 나타나고, 수신기에서 현재 사용 가능한 애플리케이션 이름을 읽어와 화면에 표시한다.

애플리케이션 리스트 화면에서 원하는 서비스를 선택하면 그림 8과 같이 일반 애플리케이션의 메인 화면 중 선택한 것이 실행된다.

Ⅳ. 결 론

본 논문에서는 데이터방송에서 복수의 애플리케이션을 제공하기 위한 멀티 애플리케이션 구조/운영 방법을 제안하였다. 본 논문에서 제안한 멀티 애플리케이션은 단순히 일반 애플리케이션을 여러 개 송출하는 기존 방법과 달리, 매니저 애플리케이션이라는 특수한 구조의 애플리케이션을 일반 애플리케이션과 함께 송출하는 구조이다.

기존 방법과 같이 일반 애플리케이션만을 여러 개 송출

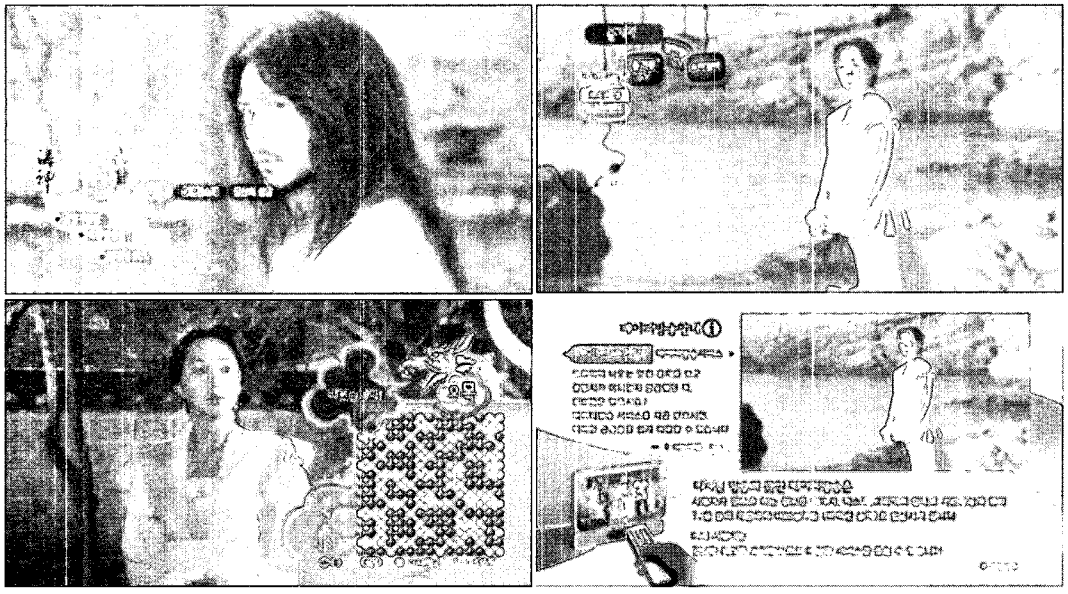


그림 8. 애플리케이션들의 메인 메뉴
Fig. 8. Main menus of applications

하면, 수신기가 이들을 처리하여 실행시키므로 하나의 애플리케이션만 표현되는 등 수신기 종류에 따라 다르게 처리된다. 제안한 방법에서는 범용의 매니저 애플리케이션이 수신기에서 가장 먼저 실행되어 일반 애플리케이션들에 대한 관리를 담당하므로, 수신기 종류에 관계없이 동일한 형태로 멀티 애플리케이션이 동작된다. 또한 기존 방법의 경우 수신기에서 제공하는 단순한 애플리케이션 리스트 UI만을 사용할 수 있지만, 제안한 방법의 경우 매니저 애플리케이션을 방송사에서 만들어 송출하므로 방송사의 특성을 반영한 다양하고 풍부한 UI를 제공할 수 있다.

제안한 멀티 애플리케이션이 본 방송에서 실시되면 사용자는 자신의 기호에 맞는 다양한 서비스를 즐길 수 있고 방송사는 편성의 자유를 가질 수 있을 것이다. 추후 본 방송에 적용되기 위해서는 매니저 애플리케이션의 지정자 화면, 애플리케이션 리스트 표현 등의 사용에 대한 심도 있는 고려가 필요하다.

참 고 문 헌

[1] ATSC Standard A/101: Advanced Common Application Platform (ACAP), 2 August 2005.

[2] ATSC homepage: http://www.atsc.org/

[3] J.-D. Kim, S. Lee, and H.-S. Kim, "Developing a fully interactive TV system and applications in compliance with ATSC ACAP," Proceedings of the NAB 2005 Broadcast Engineering, pp. 17-23, 2005.

[4] Digital Video Broadcasting(DVB), Multimedia Home Platform (MHP) version 1.0.3, ETSI TS 101 812 v.1.3.1, 2003.

[5] 이동준, 김정덕, 이상주, 이종화, "리턴채널 데이터방송 송출을 이용한 지상파 대역폭 한계 극복," 한국방송공학회 학술대회 논문집, pp. 149-152, 2005.

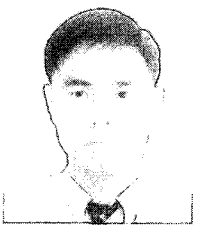
저 자 소 개

김 현 순

- 1995년 2월: 경북대학교 전자공학과 졸업
- 1997년 2월: 경북대학교 대학원 전자공학과 석사 졸업
- 2001년 2월: 경북대학교 대학원 전자공학과 박사 졸업
- 2001년 7월~현재: 한국방송 방송기술연구팀 근무
- 주관심분야 : DTV 데이터방송, DMB 서비스

권 재 광

- 1989년 2월: 연세대학교 컴퓨터학과 졸업
- 1991년 8월: 연세대학교 대학원 컴퓨터학과 졸업(석사)
- 1991년 8월~현재: 한국방송 방송기술연구팀 근무
- 주관심분야 : DTV 데이터방송, 방송통신 융합 서비스



강 대 갑

- 1986년: 부산대학교 전자공학과 졸업
- 1989년: 한국과학기술원 대학원 졸업(석사)
- 1989년~현재: 한국방송 방송기술연구팀 근무
- 주관심분야 : DTV 데이터방송, 방송통신 융합 서비스