제22회 한국정보처리학회 추계학술발표대회 논문집 제11권 제2호 (2004. 11)

임베디드 리눅스 기반 UPnP AV 미디어 서버의 구현

이동훈, 배수영, 조창식, 마평수
한국전자통신연구원
{leedh95, manim75, cscho, pmah}@etri.re.kr

Implementation of UPnP AV Media Server Based on a Embedded Linux

Dong-Hoon Lee, Su-Young Bae,
Chang-Sik Cho, Pyeong-Soo Mah
Electronics and Telecommunications Research Institute

요 약

UPnP 미디어 서버는 UPnP AV 아키텍처에서 컨트롤 포인터에게 서버와 서버에 저장되어 있는 컨텐츠에 대한 정보를 제공하기 위해서 구현되는 미들웨어이다. 보아 서버와 MPlayer로 구현되는 스트리밍 환경에서 UPnP AV 아키텍처는 각 디바이스를 발견하고 정보와 서비스를 질의하며 스트리밍 서비스를 설정하고 제어할 수 있는 기능을 제공한다. 본 논문에서의 UPnP 미디어 서버는 리눅스 셋탑 환경에서 동작하며, 서비스를 제공하는 AV 컨텐츠에 대한 정보를 제공하고 질의를 처리하기 위한 컨텐츠 디렉토리(Content Directory) 서비스와 미디어 서버, 타겟 디바이스 사이에서 전송 프로토콜과 데이터 포맷을 조율하기 위한 커넥션 메니저 (Connection Manager) 서비스를 구현하였다. 미디어 서버는 XML 기반의 DIDL로 기술된 메타 데이터를 이용하여 서버의 컨텐츠 정보를 관리하며, 컨트롤 포인터의 요청을 맞게 정보를 재구성하여 전달한다.

1. 서론

홈 네트워크가 구현된 가정에서 일반 사용자는 다양한 멀티미디어 컨텐츠를 이용할 수 있다. PC 뿐만 아니라 셋탑 박스를 이용한 TV, DVD 플레이어, 화상 카메라 등 다양한 디바이스들이 홈 네트워크에 존재할 수 있으며 사용자는 이러한 디바이스들을 단일하고 직관적인 방법을 통해서 제어할 수 방법이 필요하다. 이러한 환경에 적용하기 위해서 정의된 UPnP AV 아키텍처는 현재 네트워크에 있는 디바이스를 자동으로 발견할 뿐만 아니라 그 디바이스에 정의되어 있는 서비스를 호출함으로써 다양한 AV 디바이스들을 제어하여 자신이 원하는 서비스를 이용할 수 있다[1][2][3][4].

UPnP AV 아키텍처는 서비스를 제공하는 서버 디바이스를 제어하는 미디어 서버와 컨텐츠를 재생하는 클라이언트 디바이스를 제어하는 미디어 랜더러, 그리고 미디어 서버와 미디어 랜더러의 서비스를 이용하여 AV 서비스를 설정하고 제어할 수 있는 컨트롤 포인터로 구성된다[2][3][4].

본 논문은 UPnP AV 아키텍처의 요소 중에서 미디어 서버를 리눅스 환경에서 구현하여 서비스하는 사례를 소개한다. 리눅스에서 보아 서버를 이용하여 HTTP GET 방식으로 컨텐츠를 스트리밍하며 이 데이터는 셋탑 환경에서 MPlayer를 이용하여 TV에서 재생된다. 미디어 서버는 서버에 저장되어 있는 컨텐츠에 대한 정보와 프로토콜 정보 등을 컨트롤 포인터에게 제공하며, 이러한 정보를 받은 컨트롤 포인터는 미디어 랜더러의 정보와 조율하여 서비스를 설정할 수 있다. 본 논문에서는 미디어 서버의 구현 방법과 컨텐츠의 기술 방법, 서비스의 구현 등을 소개할 것이다.

논문은 다음과 같이 구성된다. 제2장에서는 UPnP AV 아키텍처에 대한 내용을 설명하고, 제3 에서는 미디어 서버의 구현에 대해서 설명한다. 제4장에서

구현방법을, 마지막으로 제5장에서 결론을 맺는다.

2. UPnP AV 아키텍처

UPnP AV 아키텍처는 UPnP 디바이스 아키텍처를 기반으로 AV 서비스를 제어하기 위해서 필요로 하는 서비스와 상태변수 등을 정의한 표준이다. UPnP AV 아키텍처는 미디어 서버, 미디어 랜더러, 컨트롤 포인터로 구성이 되며 이들의 관계는 그림 1과 같다[2][3][4].

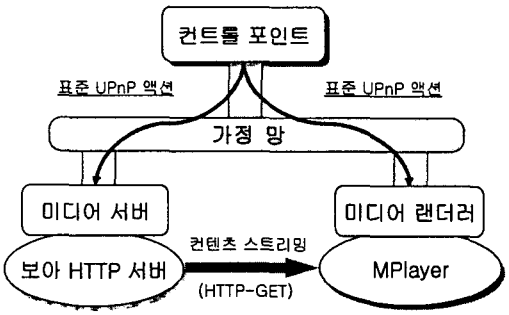

그림 1 UPnP AV 아키텍처 구현

2.1 UPnP 디바이스 아키텍처

UPnP AV 아키텍처의 각 구성 요소들은 UPnP 디바이스 아키텍처를 기반으로 한다[5]. UPnP 디바이스 아키텍처는 네트워크에 장치들이 서로 동적으로 진입하고 각각의 존재와 기능을 다른 장치들과 공유하여 제어하기 위한 미들웨어이다. UPnP 디바이스 아키텍처는 네트워크에 진입한 장치들의 존재를 발견하거나 알리기 위한 디스커버리(Discovery), 장치의 기능을 XML 기반의 기술언어로 알려주는 디스크립션(Description), 장치에 정의되어 있는 서비스를 호출하여 기능을 이용하는 컨트롤(Control), 마지막으로 장치의 상태 변화를 다른 디바이스에 알려주는 이벤트(Event) 기능 등을 가진다.

2.2 컨트롤 포인트

컨트롤 포인트는 네트워크에 존재하는 미디어 서버와 미디어 랜더러를 발견하고 이들이 지원하는 기능을 적절히 조합하여 미디어 재생 서비스를 제공한다. 홈 네트워크에는 다양한 형태의 미디어 서버와 랜더러가 존재할 수 있으며, 이러한 디바이스들에서 재생을 위한 복잡한 과정을 추상화하여 단일한 인터페이스를 제공할 수 있다. 컨트롤 포인트는 미디어 서버의 Browse() 서비스를 호출하여 서버에 저장되어 있는 컨텐츠에 대한 정보를 알 수 있고, 지원하는 프로토콜이나 파일 포맷의 정보를 획득하며, 미디어 서버와 랜더러 사이의 연결 설정을 준비하여 선택된 컨텐츠에 대한 재생을 시작할 수 있다. 뿐만 아니라 재생하는 과정에서 전송을 제어하거나 랜더링의 특성을 조절할 수 있는 기능도 제공하게 된다.

2.3 미디어 서버

미디어 서버는 컨텐츠를 스트리밍하는 서버에 관련된 정보를 처리한다[6][7]. UPnP AV 아키텍처 표준에는 세 가지의 서비스가 정의되어 있는데, 미디어 서버가 홈 네트워크에 제공할 수 있는 AV 컨텐츠에 대한 정보를 처리하는 컨텐츠 디렉토리(Content Directory) 서비스[8], 미디어 서버와 랜더러 사이에서 전송 프로토콜과 데이터 포맷을 조율하는 커넥션 메니저(Connection Manager) 서비스[9], 그리고 컨텐츠에 대한 흐름을 제어하기 위한 AV 전송 (AV Transport) 서비스이다. 본 논문의 구현에서는 HTTP 서버를 이용하여 PULL 방식으로 스트리밍이 구현되기 때문에 AV 전송 서비스는 구현되지 않는다.

서버에 저장되어 있는 컨텐츠는 XML 기반의 DIDL 기술언어로 기술된다. 미디어 서버는 이 메타데이터를 파싱하여 현재 저장되어 있는 컨텐츠에 대한 정보를 알 수 있고 컨트롤 포인트의 요청에 따라 이 정보를 재구성하여 결과 값으로 전달하게 된다.

2.4 미디어 랜더러

미디어 랜더러는 MPlayer를 이용하여 컨텐츠를 재생할 때 요구되는 정보를 처리한다[10]. UPnP AV 아키텍처 표준에 정의되어 있는 서비스는 미디어 서버와 연결을 설정하고 관리하기 위한 커넥션 메니저(Connection Manager) 서비스와 컨트롤 포인트가 현재의 랜더링 속성을 설정하고 변경할 수 있는 랜더링 컨트롤(Randering Control) 서비스, 그리고 미디어의 전송을 제어하는 AV 전송(AV Transport) 서비스이며 미디어 랜더러는 이 세 가지 서비스를 모두 구현한다.

미디어 랜더러와 MPlayer 사이의 통신은 두 모듈의 독립적인 동작을 보장하기 위해서 FIFO를 사용하며 이는 양방향 통신을 지원한다. MPlayer는 미디어 랜더러가 FIFO에 기록한 정보로 보아 서버에 접속을 시도하며 미디어 랜더러는 MPlayer가 기록한 현재 상태를 바탕으로 현재의 재생을 제어할 수 있다.

3. 미디어 서버

본 논문에서 구현하는 UPnP 미디어 서버의 구조는 그림 2와 같다.

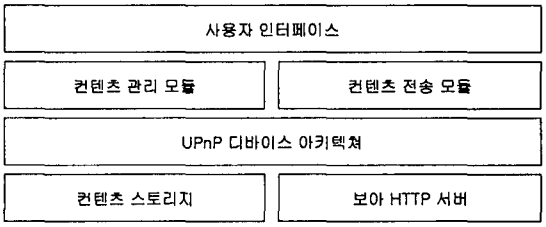

그림 2 UPnP 미디어 서버의 구조

컨텐츠 스토리지에는 스트리밍 서비스를 제공하는 컨텐츠와 각 컨텐츠에 대한 정보를 기록한 메타데이터가 있으며 보아 HTTP 서버는 MPlayer의 요청에 따라서 컨텐츠를 HTTP-GET 함수를 이용하여 스트리밍하게 된다. UPnP 디바이스 아키텍처는 UPnP의 디스커버리, 디스크립션, 컨트롤, 이벤트의 기본 동작을 수행하며 이 부분은 Intel SDK를 이용하여 구현된다[11]. 컨텐츠 관리 모듈은 컨텐츠 디렉토리 서비스를 구현한 모듈이고 컨텐츠 전송 모듈은 커넥션 메니저 서비스를 구현한다. 사용자 인터페이스는 서버 컨텐츠와 메타 데이터를 관리하기 위한 GUI를 제공한다. 각 모듈에 대한 상세 설명은 다음과 같다.

3.1 컨텐츠 관리 모듈

컨텐츠 전송 관리 모듈은 UPnP 미디어 서버에서 컨텐츠 디렉토리(Content Directory) 서비스를 구현하고 각 컨텐츠에 대해서 생성된 메타 데이터를 처리하는 모듈이다. 이 모듈은 미디어 서버가 홈 네트워크에 제공할 수있는 AV 컨텐츠에 대한 정보를 요구하는 질의를 처리할 뿐만 아니라 서버에 저장되어 있는 다양한 컨텐츠를 관리하는 일을 수행한다. 즉, 네트워크에 존재하는 컨트롤 포인터가 이 모듈에 구현되어 있는 컨텐츠 디렉토리 서비스의 액션(action)을 통하여 미디어 서버에 이용 가능한 컨텐츠의 정보를 조회할 수 있고, 서버 관리자는 서버의 스토리지 공간에 저장되는 다양한 컨텐츠에 대해서 등록하고 관련된 정보를 기술하는 등의 일을 수행할 수 있다.

컨트롤 포인터가 이용할 수 있는 액션들 중에 가장 기본이 되는 액션은 Browse()인데, 컨트롤 포인터가 액션을 호출할 때 전달하는 인자에 따라서 서버에 저장되어 있는 컨텐츠들을 조회하고 각 컨텐츠에 대한 컨텐츠의 이름, 크기, 제작자, 생성 일시와 같은 상세 정보를 요구 사항에 맞게 재구성하여 리턴하게 된다. Browse()와 같은 액션이 참조하게 되는 메타 데이터는 서버에 저장된 각 컨텐츠의 정보를 아이템(item)과 컨테이터(container) 등의 단위로 기술되며 이러한 정보를 기술은 XML 기반의 DIDL(Digital Item Declaration Language)을 사용한다. DIDL을 이용하여 기술된 언어는 XML 기반으로 작성되었기 때문에XML 파서를 이용하여 정보를 분석하여 DOM 자료구조를 생성할 수 있으며DOM API를 통하여 컨트롤 포인트가 필요로 하는 정보를 제공하고 컨트롤 포인트가 호출하는 Action을 처리하게 된다.

3.2 컨텐츠 전송 모듈

컨텐츠 전송 모듈은 UPnP 미디어 서버에서 커넥션 매니저(Connection Manager) 서비스를 구현하는 모듈이다. 이 모듈은 컨트롤 포인트에 의한 설정 과정 후에 실제로 미디어 랜더러에서 서버에 접속할 때 필요로 하는 정보를 제공하는 기능을 수행한다. 이 모듈의 가장 중요한 기능은 미디어 서버와 미디어 랜더러 사이의 매칭되는 정보를 조회하는 것이다. 여기서 매칭은 파일 포맷에 대한 매칭(예. mp3-mp3)과 전송 프로토콜에 대한 매칭(예. http-http)을 포함한다. 뿐만 아니라, 현재 네트워크 상에서 진행되고 있는 스트리밍에 관한 정보를 발견하여, 컨트롤 포인트가 서비스를 연결하고자 하는 미디어 서버와 미디어 랜더러에 이미 스트리밍이 진행되고 있는지를 점검할 수 있다.

이 모듈은 프로토콜 정보(Protocol Info)라는 값을 가지는데 이 값은 프로토콜과 네트워크, 파일 포맷, 추가 정보를 다음과 같은 형식으로 유지한다.

<protocol>':'<network>':'<contentFormat>':'<additionalInfo>

만약, HTTP-GET 방식으로 MP3 포맷을 스트리밍 한다면,"http-get:*:audio/mpeg:*" 값으로 프로토콜 정보가 설정된다.

3.3 사용자 인터페이스

사용자 인터페이스는 서버에 저장되는 컨텐츠를 관리하기 위한 GUI를 제공하는 것으로 컨텐츠에 대한 등록, 정보 수정, 삭제등과 관련된 기능을 제공하는 모듈이다. 사용자는 이 인터페이스를 통해서 지정된 컨텐츠와 메타 데이터 루트 디렉토리에 새로운 컨텐츠를 등록할 수 있을 뿐만 아니라 현재 등록된

컨텐츠에 대한 정보를 수정할 수있다. 컨텐츠에 대한 등록은 컨텐츠 루트 디렉토리에 해당하는 컨텐츠 파일을 복사할 뿐만 아니라 컨텐츠의 헤더 정보 분석과 사용자의 입력 정보를 바탕으로 메타 데이터를 생성하여 메타 데이터 루트 디렉토리에 저장한다.

각 컨텐츠의 파일 포맷을 분석하거나 사용자가 인터페이스를 통해서 입력되는 정보는 컨텐츠에 대한 타이틀, 생성일자, 생성자, 프로토콜 정보, URI, 컨텐츠의 크기, 컨텐츠 포맷 등이고 이 정보가 DIDL 표준에 따라서 기술되어 메타 데이터 서버에 저장된다. 다음은 메타 데이터 서버에 저장되는 이미지 컨텐츠에 대한 기술 예이다.

```
<item id="11" parentID="1" restricted="1">
<dc:title>Jun Ji Hyun 1</dc:title>
<upnp:class>object.item.imageItem.photo</upnp:class>
<upnp:storageMedium>UNKNOWN</upnp:storageMedi
um>
<dc:date>2004-08-30</dc:date>
<dc:creator> Naver </dc:creator>
<upnp:writeStatus> UNKNOWN </upnp:writeStatus>
<res protocolInfo= "http-get:*:image/jpeg:*" importUri
="http://129.254.181.113:62052/MediaServerContent/
">http://129.254.181.113:62052/MediaServerContent/
1.jpg</res>
```

4. 구현

미디어 서버는 리눅스 운영체제에서 C언어를 이용하여 구현되었으며, 사용자 GUI는 GTK+를 사용하였다. 서버에 저장되는 컨텐츠는 루트 컨테이너(root container)에 오디오, 비디오, 이미지 등 컨텐츠의 종류별로 자식 컨테이너(child container)를 만들고 각 컨텐츠의 컨테이너에 파일 별로 아이템(item)을 이용하여 속성을 기술하였다.

미디어 서버가 네트워크에서 실행이 되어 자신의 존재를 광고 메시지를 통해서 알리면, 광고 메시지를 수신한 컨트롤 포인터는 미디어 서버 디바이스와 정의된 서비스에 대한 정보를 요구하며 이는 XML기반의 기술문서로 컨트롤 포인트에게 전달된다. 이 기술문서에 정보에 따라서 컨트롤 포인트는 자신이 원하는 서비스의 액션을 호출하거나 자신이 구독받기를 원하는 상태 변수를 이벤팅할 수 있다.

현재 컨텐츠에 대한 정보를 조회하기 위해서 호출하는 Browse() 액션은 ObjectID 인자를 통해서 특정한 객체를 명시할 수 있고, BrowseFlag 인자를 통해서 그 객체의 메타 데이터인지 그 객체가 가진 자식 객체들에 대한 메타 데이터인지를 나타낼 수 있다. 인자 값에 따라서 컨트롤 포인트가 원하는 정보를 파악하게 되면 현재의 메타데이터를 조건에 따라서 DIDL 언어로 재구성하여 Result 인자를 통해서 컨트롤 포인트에게 전달할 수 있다.

5. 결론

본 논문은 리눅스환경에서 동작하는 UPnP AV 아키텍처의 미디어 서버에 대한 구현을 설명하였다. 미디어 서버는 UPnP 디바이스 아키텍처를 기반으로 구현되기 때문에 네트워크에 진입하면 자동으로 자신과 다른 디바이스의 존재를 파악하고 자신의 서비스를 다른 디바이스에 알릴 수 있다. 뿐만 아니라, 스트리밍 서비스에서 서버 파트의 정보를 제어하기 위해서 AV 컨텐츠에 대한 정보를 제공하고 질의를 처리하기 위한 컨텐츠 디렉토리 서비스와 미디어 서버, 타겟 디바이스 사이에서 전송 프로토콜과 데이터 포맷을 조율하기 위한 커넥션 매니저 서비스를 구현하였다. 컨트롤 포인터는 미디어 서버와 랜더러를 조율하여 사용자에게 보다 직관적이고 추상화된 방법으로 AV 서비스를 이용하고 제어할 수 있는 방법을 제공한다.

미디어 서버는 리눅스 환경에서 구현되었기 때문에 보아나 아파치와 같은 HTTP 서버뿐만 아니라 다윈 서버와 같은 실시간 스트리밍 서버 등 리눅스 환경에서 공개용 서버를 이용하여 구현되는 스트리밍 서비스에 적용될 수 있으며, UPnP 표준을 준수하기 때문에 인텔에서 제공하는 다양한 UPnP 툴처럼 다른 운영체제에서 동작하는 UPnP 디바이스 등과도 호환하여 동작할 수 있다.

참고문헌

[1] UPnP Forum, http://www.upnp.org

[2] UPnP AV Architecture V 0.83, June 25, 2002

[3] Michael Jeronimo and Jack Weast, UPnP Design by Example, Intel Press

[4] Overview of UPnP AV Architecture, Intel, July 22.2003

[5] UPnP Device Architecture V 1.0, Dec. 2003

[6] UPnP MediaServer V1.0, June 25, 2002

[7] Designing a UPnP AV MediaServer, Intel, May 20. 2003

[8] UPnP ContentDirectory V1.0, June 25, 2002

[9] UPnP ConnectionManager V1.0, June 25, 2002

[10] UPnP MediaRender V1.0, June 25, 2002

[11] Linux SDK for UPnP Devices V1.2, http://docpp.sourceforge.net, Jan 17. 2003