멀티미디어 스트리밍 프레임워크에서 컨텐츠 관리자의 설계 및 구현

홍 영 래[†] · 김 형 일[†] · 이 승 룡[††] · 정 병 수[††] · 윤 석 환[†††] · 정 찬 근[††††]

요 약

본 논문에서는 통합 스트리밍 프레임워크(ISSA : Integrated Streaming Architecture)의 주요 모듈인 컨텐츠 관리자의 설계와 구현에 대한 경험을 소개한다. ISSA는 유니캐스팅/멀티캐스팅 환경의 VOD 시스템과 실시간 방송시스템(라이브캐스팅)과 같은 통합 멀티미디어 스트리밍 서비스 응용을 개발하기 위해 세션 관리자, 전송 관리자, 미디어 관리자, 컨텐츠 관리자, 데이터베이스 커넥터, 게이트웨이 등의 라이브러리를 제공하는 통합 스트리밍 프레임워크이다. ISSA는 RTSP, RTP 등의 표준 멀티미디어 프로토콜을 사용함으로써 범용성을 제공하며, 강력한 실시간 멀티미디어 데이터베이스 연동 기능을 제공한다. 본 논문에서 제안하는 컨텐츠 관리자는 서버에서 제공하는 다양한 미디어 컨텐츠들에 대한 정보들을 관리하고 이를 스트리밍 서버와 웹서버, 그리고 클라이언트에게 효율적으로 알려주어 사용자가 원하는 미디어에 대해 쉽게 검색과 접근을 할 수 있다. 또한, 스트리밍에 관련된 정보를 미디어 소스와 전송 관리자에게 알려줌으로써 빠르고 효율적인 스트리밍이 이루어지도록 한다. 그리고, 기존의 컨텐츠 관리자와는 달리 데이터베이스 커넥터를 제공하여 실시간 멀티미디어 데이터베이스의 트랜잭션 처리를 지원한다.

Design and Implementation of a Content Manager in the Multimedia Streaming Framework

Young-Rae Hong[†] · Hyung-Ill Kim[†] · Sungyoung Lee[††] · Byeong-Soo Jeong[††] · Seok-Hwan Yoon[†††] · Chan-Gun Jeong[††††]

ABSTRACT

This paper describes design and implementation of a content manager in the Integrated Streaming Framework Architecture(ISSA) that is proposed by the authors. The ISSA can provide an environment to develop multimedia streaming applications under heterogeneous distributed systems. The goal of ISSA is to extend the limitations of existing streaming systems. It can support diverse media formats and high level programming environment for streaming application developers. Moreover, it is independent from underlying networks and operating systems, and compatible with the global real-time multimedia database system(BeeHive) so that streaming media is efficiently retrieved, stored, and serviced. The role of a content manager is important in the ISSA environment since it manages an information of media that are provided by the server, and allow users to access media more easily by means of conveying that information to the streaming server, Web server, and client efficiently. The proposed content manager is not only to meet these requirements, but also to provide streaming informations to media source and transport manager in order to be an efficient streaming. Furthermore, it supports database transaction processing by using the database connector.

※ 본 논문은 정보통신부 국제공동연구과제(과제번호번호 : IJRP-9803-6)에 의해 지원받았음.
† 준 회 원 : 경희대학교 대학원
†† 종신회원 : 경희대학교 교수
††† 종신회원 : 경희대학교 전자정보학부 교수
†††† 정 회 원 : 정보통신연구진흥원 책임연구원
논문접수 : 1999년 12월 29일, 심사완료 : 2000년 1월 24일

1. 서 론

스트리밍 기술은 기존의 파일 다운로드 방식과 달리 네트워크 상에서 오디오와 비디오, 애니메이션, MIDI 등의 멀티미디어 데이터를 실시간으로 전송하면서 동시에 재생하는 기술이다. 이렇게 함으로써 사이즈가 큰 미디어를 전부 다운로드 할 때까지 기다려야 하는 수고를 덜 수 있다. 현재 많이 사용되고 있는 스트리밍 제품으로는 Microsoft 사의 Windows Media Technology와 RealNetwork 사의 RealSystem G2가 있으며, 이외에 다양한 제품들이 스트리밍 서비스를 지원하고 있다.

그러나 이러한 스트리밍 상용 제품들은 다른 시스템과의 연동을 고려하지 않고 독자적인 환경에서 동작하도록 설계되어 이식성과 유연성이 부족하며 추가적인 확장이 쉽지 않다. 그리고 오디오/비디오 CODEC 등과 같은 다양한 미디어 처리 기능이 미약하며, 다양한 운영체제 그리고 네트워크 환경을 지원할 수 있는 투명성이 부족하다. 따라서, [1, 3]에서는 이 같은 제한점 개선하기 위해서 유연하고 확장과 다른 스트리밍 시스템과의 연동이 용이할 뿐만 아니라, 다양한 오디오/비디오 미디어 형식을 지원할 뿐 만 아니라 다양한 운영체제와 네트워크 환경에서 동작할 수 있는 적응력을 지닌 통합 멀티미디어 스트리밍 구조인 ISSA(Integrated Streaming Service Architecture)를 제안하였다.

ISSA는 RTSP(Real-Time Streaming Protocol)[6], RTP(Real-Time Transport Protocol)[7] 등 표준 프로토콜을 사용함으로써 범용성을 제공하며, 멀티미디어 실시간 데이터 베이스인 BeeHive[4, 5]와의 연동 기능도 제공하고 있다. 또한, ISSA는 유니케스팅/멀티케스팅 환경의 VOD 시스템과 실시간 방송시스템(라이브캐스팅)과 같은 응용을 개발하기 위한 라이브러리를 제공하는 통합 스트리밍 서비스 구조이다. ISSA의 주요 모듈로는 VOD 서비스와 방송 서비스의 세션 제어를 위해 RTSP 등을 지원하는 세션 관리자(Session manager), 멀티미디어의 전송을 위해 RTP, TCP, UDP 등을 지원하는 전송 관리자(Transport manager), 다양한 미디어 관리를 위한 미디어 관리자(Media manager), 그리고 CORBA A/V 스트리밍 서비스[8, 9]와 같은 다른 스트리밍 서비스 플랫폼과의 연동을 위한 게이트웨이(Gateway) 모듈과 BeeHive, Oracle 등의 데이터베이스 연결과 트랜잭션을 지원하는 데이터베이스 커넥터를 관리하는 컨텐츠 관리자(Contents manager)로 구성되어 있다.

한편, 미디어 컨텐츠에 대해서 사용자가 스트리밍 서비스를 받기 위해서는 서버에서 제공하는 미디어 컨텐츠에 대한 정보를 제공받아야 한다. 즉, 서버는 자신이 제공하는 미디어 컨텐츠를 관리하는 방법과 컨텐츠 정보를 클라이언트에게 전달하는 방법을 정의하는 컨텐츠 관리자를 가지고 있어야 한다. 컨텐츠 관리자는 서버에서 제공하는 미디어 컨텐츠들에 대한 정보들을 관리하고 이를 스트리밍 서버와 웹서버, 그리고 클라이언트에게 효율적으로 알려주는 기능을 한다. 이렇게 함으로써 사용자는 원하는 미디어 컨텐츠에 대해 쉽게 접근할 수 있다.

미디어 컨텐츠의 종류로는 파일 컨텐츠, 스케줄된 파일/라이브 컨텐츠, 라이브 컨텐츠, 데이터베이스 컨텐츠가 있다. 이중 데이터베이스 컨텐츠는 기존의 컨텐츠 정보만을 저장하고 검색하는 기능뿐만 아니라 실지 스트림 데이터를 포함하는 멀티미디어 데이터베이스에 저장된 미디어 컨텐츠를 의미한다. 또한, 컨텐츠 관리자는 스트리밍에 관련된 정보를 미디어 소스와 전송 관리자에게 알려줌으로써 빠르고 효율적인 스트리밍이 이루어지도록 한다. 본 논문에서 제안하고있는 ISSA 컨텐츠 관리자는 파일 컨텐츠와 데이터베이스 컨텐츠를 지원하며, 버지니아 대학에서 개발중인 실시간 멀티미디어 객체지향 데이터베이스인 BeeHive와 연동하여 스트리밍 서비스를 제공하고 있다.

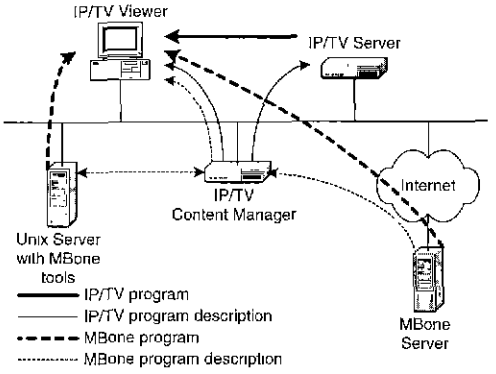
본 논문의 구성은 다음과 같다. 2장에서는 컨텐츠 관리자의 관련연구를 소개하고, 3장에서 통합 분산 스트리밍 프레임워크인 ISSA의 구조와 각 모듈의 기능에 대해서 간단히 살펴본다. 4장에서는 ISSA 컨텐츠 관리자의 설계 모델을 설명하고, 5장에서는 그것의 기능과 역할, 그리고 컨텐츠 관리자를 통해서 제어되는 데이터베이스 커넥터를 이용하여 BeeHive와 연동하는 방법에 대해서 설명하고, 6장에서 결론을 맺는다.

2. 관련 연구

Cisco 사의 IP/TV[10]는 라이브 미디어, 녹음/녹화된 미디어, 주문형 미디어, 스케줄된 미디어 등을 IP 기반의 로컬 네트워크 또는 인터넷에서 전송하는 어플리케이션이다. IP/TV는 현재 MPEG1, MPEG2, MPEG4, H.261을 포함하는 다양한 미디어 포맷을 지원한다. IP/TV는 스트리밍된 미디어를 보기 위한 뷰어(Viewer)와 다양한 미디어 컨텐츠를 관리하며 뷰어에게 미디어에

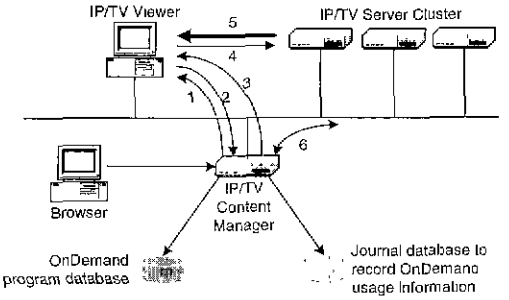
대한 정보를 제공하는 컨텐츠 관리자, 그리고 컨텐츠 관리자에 의해서 설정된 미디어 컨텐츠를 전송하고 기록하는 서버로 구성된다.

이 중 IP/TV 컨텐츠 관리자는 스케줄된 미디어와 주문형 미디어를 관리한다. 스케줄된 미디어 스트리밍은 컨텐츠 관리자에 의해 지정된 시간에 하나의 멀티캐스트 채널을 통해서 사용자에게 전송이 이루어지는 서비스이며, 멀티캐스트 채널을 이용함으로써 네트워크의 과부하 없이 무제한의 사용자에게 서비스를 할 수 있다. 이때 대상이 되는 미디어는 라이브 미디어와 저장된 미디어이다. IP/TV를 이용한 스케줄된 미디어의 전송구조는 (그림 1)과 같다. 컨텐츠 관리자는 스케줄된 미디어의 리스트를 정의하고 이 리스트는 연결된 IP/TV 뷰어에 자동으로 전달된다. 스케줄된 스트리밍 서비스가 서버에 의해서 이루어지는 반면 주문형 스트리밍 서비스는 사용자의 요구에 의해서 스트리밍이 이루어지는 서비스이다. 이는 클라이언트의 각 요구에 대해 개별 채널을 설정해야 하므로 스케줄된 스트리밍 서비스보다 많은 네트워크 대역폭을 요구하게 된다. IP/TV를 이용한 주문형 미디어 스트림의 전송 구조는 (그림 2)와 같다. IP/TV 컨텐츠 관리자는 스케줄된 스트리밍 서비스와 주문형 스트리밍 서비스에 대해서 체계적으로 다루고 있으나, 멀티미디어 데이터베이스에 대한 연결과 트랜잭션 기능은 아직 지원하지 않고 있다.



(그림 1) IP/TV 컴포넌트 - 스케줄된 미디어 스트리밍

B · media 컨텐츠 관리자[12]는 이미지, 문서, 오디오, 비디오 등의 컨텐츠를 구성하고, 컨텐츠에 대한 인덱싱, 저장, 검색 기능을 지원한다. 그리고, 흩어져 있는 다양한 컨텐츠들을 하나의 장소에서 관리할 수 있는 방법을 제공한다. 현재, B · media 컨텐츠 관리자는 단지 미디어 컨텐츠 자체에 대한 관리메커니즘을 제공하며 스트리밍 서비스와의 연관관계를 명시하지는 않는다.

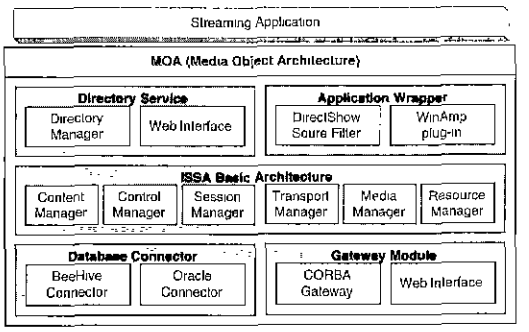


(그림 2) IP/TV 컴포넌트 - OnDemand 프로그램

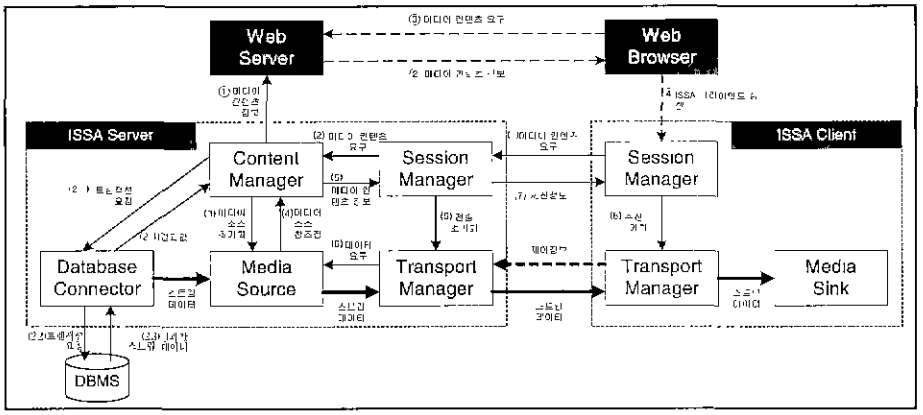
Sonera Live의 컨텐츠 관리자[11]는 미디어 파일의 빠른 서버 업로드, 자동 인코딩 기능을 제공한다. 현재 이것은 Java 환경에서 작동하며, 미디어의 관리 및 정보 검색 기능 등은 제공하고 있지 않다.

3. 통합 분산 스트리밍 프레임워크 구조

이 장에서는 ISSA에 대하여 간단히 소개한다. ISSA 프레임워크의 모델은 (그림 3)과 같이 크게 상위 계층의 스트리밍 어플리케이션과 ISSA, 그리고 ISSA와 스트리밍 어플리케이션 사이의 인터페이스인 MOA(Media



(그림 3) ISSA 프레임워크



(그림 4) ISSA 기반의 스트리밍 클라이언트/서버 모델

Object Architecture)[2]로 구성되어있다. ISSA는 분산되어 있는 미디어를 하나로 통합하여 관리하고 웹 인터페이스를 통하여 웹 브라우저로 제어가 가능하도록 하는 디렉토리 서비스, 클라이언트 애플리케이션을 기존의 미디어 애플리케이션과 연결시켜주는 Application Wrapper, 실시간 멀티미디어 데이타베이스인 BeeHive와 연동시켜주는 BeeHive 커넥터, CORBA 연동을 위한 CORBA 게이트웨이와 웹 환경과 연결시켜주는 Web 인터페이스로 구성된다.

그리고, 프레임워크의 핵심을 이루는 ISSA 기본구조는 컨텐츠, 제어, 세션, 전송, 미디어 및 자원 관리자로 구성되어있다. 컨텐츠 관리자는 파일, 데이터베이스, 라이브 미디어의 관리를 지원하고 데이터베이스와의 연동을 담당하는 데이터베이스 커넥터를 관리한다. 제어 관리자는 ISSA 서버 모듈과 외부 인터페이스를 제공해주는 역할을 해준다. 세션 관리자는 RTSP 기반의 멀티미디어 스트리밍 서비스의 세션 제어를 담당하며, 유니캐스팅과 멀티캐스팅을 지원할 수 있는 구조로 되어있다 또한. 데이터베이스의 트랜잭션 요청을 위해 RTTP(Real-Time Transaction Protocol)[9]를 지원한다. 전송 관리자는 TCP 및 UDP, 그리고 RTP/UDP 프로토콜을 이용하여 멀티미디어의 전송을 담당하며, RTCP 프로토콜을 이용하여 네트워크의 상태를 모니터링 한다 미디어 관리자는 미디어의 인코딩과 디코딩을 담당하며, 현재 MPEG1, MPEG2, MPEG4, MP3를 지원한다. 그리고, RTSP 소스필터를 개발하여 Microsoft 사의 Windows Media Player[15]와 연동을 하고 있으며, WinAmp의 플러그인을 개발하여 MP3를 서비스하고 있다. 자원 관리자는 스트리밍 시스템에서 서비스 품질(QoS - Quality of Service)의 명세화, 매핑, 모니터링, 제어 기능을 제공하며, 메모리 버퍼 관리, 스레드 스케줄링 등의 기능을 수행한다[13, 14].

(그림 4)는 ISSA 기반의 스트리밍 시스템의 클라이언트/서버의 구조를 나타낸 것이며, 웹과의 연동 방법도 보여주고 있다. 괄호 안의 숫자는 ISSA 서버와 클라이언트 간의 스트리밍 요구 순서를 나타낸 것이며, 원 안의 숫자는 웹서버를 통하여 서버에서 제공하는 미디어 컨텐츠에 대한 정보가 클라이언트에게 제공될 때의 스트리밍 요구 순서를 나타낸 것이다 즉, 서버에서 제공하는 미디어 컨텐츠에 대한 정보는 ISSA 클라이언트/서버에서 직접 처리할 수도 있고, 웹서버를 통해 클라이언트에 전달될 수도 있다.

ISSA 클라이언트는 세션 관리자를 통하여 미디어 컨텐츠를 요구하면 서버 측의 세션관리자는 이를 수신하여 컨텐트 관리자에게 요구를 하게 된다. 그러면 컨텐츠 관리자는 요구한 미디어의 유형(파일, 데이터베이스, 라이브)을 분석하여 적절한 미디어 소스를 초기화시키고 스트리밍을 시작하기 위해 전송 관리자를 초기화시킨다. 연결과 환경 설정이 완료되면 클라이언트는 스트리밍의 시작, 중지 등을 요청함으로써 인터렉티브한 스트리밍 서비스를 받을 수 있게 된다.

4. 컨텐츠 관리자 설계

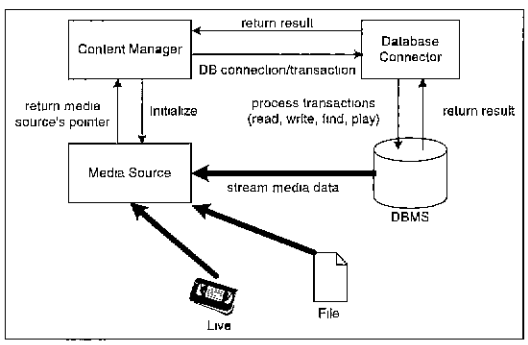
컨텐츠 관리자는 제목, 컨텐츠 유형, 위치, 포멧, 보안 등 스트리밍 서버에서 제공한 미디어 컨텐츠들에

대한 정보를 생성하고 관리하며 이를 클라이언트에게 제공하기 위해 메타파일 및 URL을 생성하는 기능을 갖는다. 컨텐츠 관리자는 미디어 컨텐츠를 유형별로 주문형 미디어 컨텐츠(파일, 데이터베이스)와 라이브 미디어 컨텐츠(비디오 카메라나 마이크에서 수신한 멀티미디어 데이터의 라이브 캐스팅), 그리고 스케줄된 미디어 컨텐츠(파일, 데이터베이스의 방송)로 구분하여 관리한다.

주문형 미디어 컨텐츠는 클라이언트의 요구와 제어에 의해서 스트리밍이 이루어지는 컨텐츠를 말하며, 파일 또는 데이터베이스 형식으로 제공된다. 라이브 미디어 컨텐츠는 마이크나 비디오 카메라를 통해서 미디어 데이터를 입력받음과 동시에 스트리밍을 하는 것으로 일시정지, 빨리 감기, 되감기 등의 클라이언트 제어는 불가능하다. 마지막으로, 스케줄된 미디어 컨텐츠는 서버에서 지정한 시간에 파일, 데이터베이스, 라이브 미디어의 스트리밍이 이루어지는 컨텐츠이다. 이 중 주문형 미디어 컨텐츠는 유니캐스팅을 지원하며, 나머지는 멀티캐스팅을 지원한다.

제안된 컨텐츠 관리자의 또 다른 기능은 멀티미디어 데이터베이스에 대한 접근과 트랜잭션을 처리하는 데이터베이스 커넥터를 관리하는 것이다. 데이터베이스 커넥터는 미디어 데이터와 정보를 포함하고 있는 데이터베이스에서 미디어 정보와 미디어 데이터를 추출하기 위한 기능들을 제공하며 BeeHive와의 연동을 위해 BeeHive 커넥터를 지원하고 있다

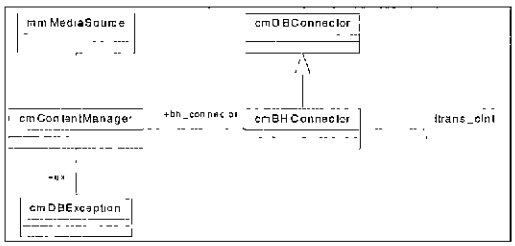
컨텐츠 관리자는 (그림 5)와 같이 미디어의 전송과 수신을 위한 기능들을 담당하는 미디어 소스, 데이터베이스와의 연결을 담당하는 데이터베이스 커넥터와 상호 통신을 한다.



(그림 5) 컨텐츠 관리자의 구조

즉, 파일명, 데이터베이스 ID 등을 파라미터로 받아 해당 미디어 소스를 초기화시키며 미디어 소스의 정보를 검색하는 기능을 제공한다 또한, 데이터베이스 커넥터와 통신함으로써 스트리밍 시작, 정보 검색, 데이터 정보 변경 등 멀티미디어 데이터베이스에서 스트리밍 서비스를 위한 트랜잭션 기능을 수행한다.

(그림 6)은 컨텐츠 관리자와 미디어 소스, 데이터베이스 커넥터와의 관계를 UML(Unified Modeling Language)의 클래스 다이어그램으로 표현한 것이다. cmContentManager는 cmBHConnector을 생성하고, cmBHConnector는 cmContentManager로부터 요청된 트랜잭션을 데이터베이스의 API를 사용하여 데이터베이스 서버에 트랜잭션을 요청하게 된다



(그림 6) 컨텐츠 관리자의 클래스 다이어그램

5. 컨텐츠 관리자의 구현

본 장에서는 컨텐츠 관리자의 주요 기능과 구현에 대하여 다룬다.

5.1 미디어 컨텐츠 관리

컨텐츠 관리자는 미디어 컨텐츠의 빠른 정보 제공과 초기화를 위해 미디어 컨텐츠에 대한 인덱스를 관리한다. 여기에서 데이터베이스에 저장된 미디어 컨텐츠는 데이터베이스에서 인덱스를 관리하게 된다. 또한, 웹을 통한 미디어 컨텐츠로 접근을 제공하기 위하여 메타파일과 URL을 자동 생성하는 기능도 제공한다. 컨텐츠 관리자가 관리하는 미디어 컨텐츠의 종류는 다음과 같다

- 주문형 미디어 컨텐츠
  주문형 미디어 컨텐츠는 사용자의 제어에 의해서 스트리밍이 이루어지는 미디어 컨텐츠를 의미한다. 즉, VCR과 같이 시작, 중지, 일시정지, 빨리감기, 되감기 등의 제어가 가능한 미디어 컨텐츠이다 이 컨텐

츠는 서버와 클라이언트 사이에 일대일의 연결이 이루어지는 유니캐스트 연결을 지원한다. ISSA에서 다루고 있는 주문형 미디어 컨텐츠로는 파일 형식의 미디어와 데이터베이스에 저장된 미디어가 있다. 파일 형식의 미디어 컨텐츠는 서버에서 컨텐츠 관리자를 통하여 미디어의 정보를 추가하고 이를 데이터베이스 또는 파일에 저장하는 방식의 컨텐츠이며, 데이터베이스에 저장된 미디어는 데이터베이스 내에 미디어에 대한 정보와 미디어 자체를 저장하고 있는 컨텐츠이다. 즉, 파일 형식의 미디어는 미디어 데이터와 정보가 따로 저장되어 있으나, 데이터베이스 미디어는 같은 위치에 존재한다. 미디어 컨텐츠를 관리하는데 필요한 정보로는 다음과 같은 것들이 있다

- 미디어 이름 : 기본값으로 파일 이름을 가지며, 사용자가 지정할 수도 있다.
- 작성자 : 미디어 컨텐츠를 작성한 작성자
- 저작권 : 미디어 컨텐츠에 대한 저작권자
- 컨텐츠 유형 : 주문형 파일
- URL : 미디어 컨텐츠의 위치 이는 미디어 파일의 메타 정보를 갖고 있는 파일의 위치이다.

컨텐츠에 대한 정보들을 기록한 뒤에는 웹이나 ISSA 클라이언트를 통해서 컨텐츠 정보를 제공할 수 있다.

- 라이브 컨텐츠

  라이브 컨텐츠는 주문형 컨텐츠와 달리 클라이언트가 시작 외에 어떠한 제어를 할 수 없는 컨텐츠이다. 라이브 컨텐츠의 입력원은 마이크나 비디오 카메라 등의 멀티미디어 디바이스이다. 라이브 컨텐츠에 대한 정보는 컨텐츠 유형을 제외하고 주문형 컨텐츠의 정보와 같다.

- 스케줄된 미디어 컨텐츠

  스케줄된 미디어 컨텐츠는 서버에 의해서 지정된 시간에 스트리밍이 시작과 종료가 결정되는 컨텐츠이다. 이 컨텐츠 유형에 포함되는 미디어는 파일, 데이터베이스, 라이브 미디어가 모두 포함될 수 있다 스케줄된 미디어 컨텐츠에 대한 정보로는 스트리밍 시작 시간과 종료 시간이 포함된다.

5.2 미디어 컨텐츠 정보 배포

컨텐츠 관리자에 의해서 작성된 컨텐츠 정보들은 웹이나 ISSA 클라이언트를 통해서 클라이언트에게 전달된다. 파일 형식의 미디어는 클라이언트에서 직접 알 수 있는 방법이 없기 때문에 메타파일을 생성하여 웹서버에게 전달하고 사용자는 웹브라우저를 통해서 미디어 컨텐츠를 요청하게 된다. 그리고, 데이터베이스에 저장된 미디어는 RTTP 프로토콜을 이용하여 미디어 정보를 검색하게 된다. RTTP를 이용하면 정보 검색뿐만 아니라 정보 변경도 가능하다. 물론 이때는 쓰기 권한이 있어야 한다. 라이브 컨텐츠인 경우도 ISSA 클라이언트에서 직접 정보를 가져오기보다는 웹을 이용하여 정보를 검색하는 것이 효과적이다.

5.3 미디어 컨텐츠 정보 검색

컨텐츠 관리자는 컨텐츠들의 정보를 바탕으로 컨텐츠가 갖고 있는 미디어 정보에 대한 빠른 검색 기능을 제공하게 된다. 미디어 컨텐츠의 정보를 얻기 위해서는 <표 1>에 나타난 대로 미디어 컨텐츠의 URL이나 미디어 소스 참조점을 매개변수로 하여 해당 함수를 호출한다.

〈표 1〉 미디어 소스의 정보 검색

```
IssaString GetMediaDesc(issaString &path);
IssaString GetMediaDesc(mmMediaSource &src);
```

그러면, 컨텐츠 관리자는 URL을 분석하여 해당 미디어의 정보를 추출한다. <표 2>는 오디오 및 비디오 미디어에 대한 정보를 요청했을 때의 결과값을 나타낸 것이다 미디어 유형이 오디오일 경우에는 제목, 저자, 인코딩 방법, 비트율, 샘플링 율(8KHz, 11.1KHz, 22.1KHz, 44.1KHz 등), 샘플 당 비트 수(8bit, 16bit 등), 채널 개수(모노, 스트레오) 등의 정보를 반환하고, 비디오일 경

〈표 2〉 오디오/비디오에 대한 정보

media_type == Audio	media_type == Video
mt=audio	mt=video
title=(title)	title=(title)
author=(author)	author=(author)
en=(encoding method)	en=(encoding method)
br=(bit rate)	bps=(bit rate)
bps=(bit per sample)	fps=(frame per second)
ch=(number of channels)	h=(frame height)
ba=(frame byte align)	w=(frame width)
fs=(frame size)	
sr=(sampling rate)	

우에는 제목, 저자, 인코딩 방법, 초당 프레임 수, 비디오 크기 등의 정보를 반환한다.

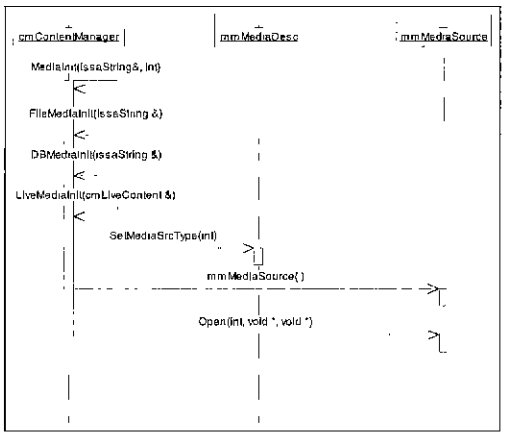
5.4 미디어 컨텐츠 스트리밍 초기화

컨텐츠 관리자는 해당 컨텐츠에 해당하는 미디어 소스(파일, 데이터베이스, 미디어 디바이스)를 초기화하고, 미디어 소스의 참조점을 반환한다. 초기화는 요청된 미디어 컨텐츠의 유형(파일, 데이터베이스, 라이브)을 분석한 후 스트리밍의 시작을 위해 해당 미디어 소스를 <표 3>의 함수들을 이용하여 수행된다.

〈표 3〉 미디어 소스의 초기화

```
mmMediaSource *
        MediaInit(issaString &path, int type);
mmMediaSource *
        FileMediaInit(issaString &abs_path);
mmMediaSource *
        DBMediaInit(issaString &id);
mmMediaSource *
        LiveMediaInit(cmLiveContent &live);
```

컨텐츠 관리자는 절대 경로를 매개변수로 받아 저장된 미디어 파일을 초기화하고 미디어 소스의 참조점을 얻는다. (그림 7)은 파일 형태의 미디어 컨텐츠를 초기화하는 과정을 시퀀스 다이어그램으로 나타낸 것이다. 미디어의 경로와 타입(파일, 데이터베이스, 라이브 등)을 매개변수로 하여 MediaInit() 메소드를 호출하면 해당 미디어 소스를 초기화하고 mmMediaSource 참조점을 리턴한다.



(그림 7) Media Source의 초기화

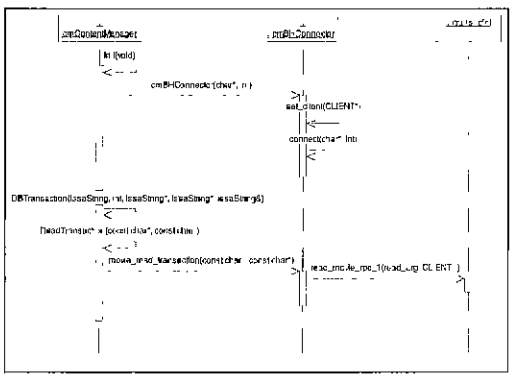
5.5 데이터베이스 트랜잭션 처리

컨텐츠 관리자는 데이터베이스와의 연동을 위해 데이터베이스 연결과 트랜잭션을 수행하는 데이터베이스 커넥터를 관리한다. 컨텐츠 관리자는 RTTP와 같은 데이터베이스 트랜잭션 프로토콜을 사용하는 세션 관리자로부터 <표 4>의 함수를 이용하여 트랜잭션 이름과 트랜잭션 관련 정보를 받아 데이터베이스 커넥터(cmDBConnector)의 트랜잭션 처리 함수를 호출한다 그러면, 데이터베이스 커넥터는 해당 데이터베이스의 API를 이용하여 데이터베이스에게 트랜잭션을 요청하고, 데이터베이스 서버는 트랜잭션을 수행한 후 결과값을 리턴한다.

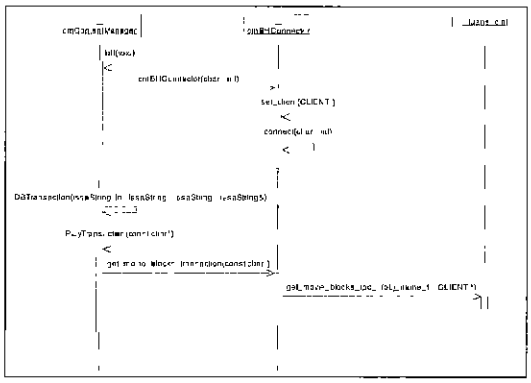
〈표 4〉 데이터베이스 트랜잭션 요청

```
int
    DBTransaction(issaString transaction_name,
                  rqst_t     request,
                  issaString &resp);
```

컨텐츠 관리자는 데이터베이스 커넥터로부터 받은 결과값을 세션 메니저에게 전달한다. (그림 8)과 (그림 9)는 BeeHive를 사용하여 READ 트랜잭션과 PLAY 트랜잭션을 수행했을 경우에 대한 시퀀스 다이어그램을 나타낸 것이다. cmContentManager에서 cmBHConnector를 초기화하면 cmBH Connector는 BeeHive의 미디어 클라이언트 부분을 담당하는 ConnectorServer와 연결을 시도한다. 연결이 설정되면, cmContentManager에서 DBTransaction() 메소드를 호출하여 해당 트랜잭션을 요청하게 된다. 이 트랜잭션은 cmBH·



(그림 8) 데이터베이스 트랜잭션 (READ)



(그림 9) 데이터베이스 트랜잭션 (PLAY)

Connector와 BeeHive의 ConnectorServer, 그리고 BeeHive의 미디어 클라이언트를 거쳐 BeeHive 서버에 트랜잭션을 전달하게 된다.

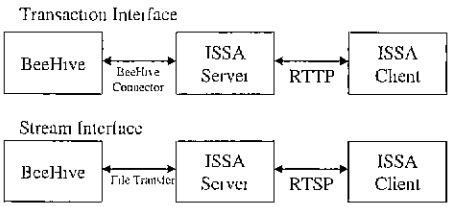
5.6절에서는 BeeHive 커넥터를 이용한 ISSA와 BeeHive 사이의 트랜잭션 인터페이스와 스트리밍 인터페이스를 살펴보도록 한다.

5.6 BeeHive 커넥터

5.6.1 BeeHive 커넥터 인터페이스 설계

ISSA와 BeeHive를 연동하기 위하여 두 시스템의 인터페이스를 정의한 것이 바로 BeeHive 커넥터이다. ISSA는 미디어 스트리밍 환경을 쉽게 개발하기 위한 프레임워크로 라이브러리 형태로 개발되었으며 클라이언트/서버 구조로 되어 있다. 클라이언트와 서버 사이의 전송 프로토콜로는 RTP, RTSP, TCP 등을 지원한다. 데이터베이스 연동을 위해서 서버쪽에 RPC(Remote Procedure Call) 기반의 인터페이스를 정의하였다. 우선 BeeHive에서 처리 가능한 트랜잭션을 정의하고, 트랜잭션 처리를 위한 RPC 인터페이스를 정의하였다. 이를 위하여 구체적인 응용 분야로 영화 데이터베이스를 선택하였다. BeeHive에서 영화 객체(Movie Object)를 저장하고 이를 다룰 수 있는 READ, WRITE, FIND 그리고 PLAY 트랜잭션을 정의하였다. ISSA와 BeeHive가 서로 동작하는 모델은 트랜잭션 인터페이스와 스트리밍 인터페이스로 나뉜다. 트랜잭션 인터페이스는 BeeHive 트랜잭션을 ISSA 클라이언트에서 요청하고 이를 처리하기 위한 일련의 과정에 대한 인터페이스이며, 스트리밍 인터페이스는 ISSA 클라이언트와 서버 사이의 미디어 스트림을 처리하기 위한 인터페이스를 의미한다. BeeHive 커넥터는 컨텐츠 관리자 내부에 위치하여, 컨텐츠 관리자에게 RTTP 프로토콜로 트랜잭션 정보를 받아 실제 해당되는 BeeHive 트랜잭션을 요구하게 된다.

ISSA와 BeeHive 간의 트랜잭션과 스트리밍 인터페이스는 (그림 10)과 같다.



(그림 10) ISSA와 BeeHive 사이의 트랜잭션 및 스트리밍 인터페이스

트랜잭션 인터페이스는 사용자로부터 트랜잭션 요청을 받는 GUI, ISSA 클라이언트와 서버 사이의 트랜잭션 처리를 위한 RTTP, ISSA 서버와 BeeHive 사이의 RPC로 구성된다. 스트리밍 인터페이스는 ISSA 클라이언트의 미디어 출력을 위한 인터페이스와 ISSA 서버와 클라이언트 사이의 RTSP, RTP 세션, ISSA 서버와 BeeHive 사이의 스트리밍 전송 인터페이스로 구성된다. ISSA 서버의 컨텐츠 관리자의 BeeHive 커넥터는 BeeHive와의 연동을 처리한다.

5.6.2 트랜잭션 인터페이스

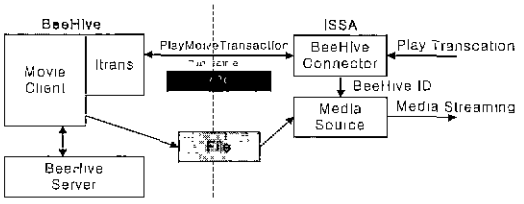
BeeHive의 트랜잭션을 ISSA 환경에서 연동하기 위하여 ISSA 구조의 컨텐츠 관리자에 BeeHive 트랜잭션 인터페이스를 추가하였다. ISSA의 컨텐츠 관리자는 데이터베이스 연동과 ISSA 클라이언트에서 요청되는 미디어 소스를 제공하는 역할을 한다. BeeHive와의 연동은 RPC를 이용하였으며, ISSA의 서버와 클라이언트 간의 트랜잭션을 위해서 RTTP을 설계하고 구현하였다. RTTP는 ISSA 환경에서 BeeHive의 트랜잭션을 요청하고 처리하는데 사용하는 것으로 BeeHive와는 직접적으로 연관되지 않는다. RTTP는 HTTP 프로토콜의 문법과 유사성이 있으며 현재는 단지 READ, WRITE, FIND, 그리고 PLAY와 같은 트랜잭션을 처리하도록 구성되어 있으나 다른 트랜잭션 처리를 할 수 있도록 쉽게 확장할 수 있다.

5.6.3 스트리밍 인터페이스

미디어 스트리밍을 위한 인터페이스는 3단계로 진화하였다. 첫 번째 단계는 파일로 저장한 뒤 파일명을 전송하는 것으로 매우 원시적인 미디어 전송 메커니즘이다 두 번째 단계는 RPC 메소드를 통하여 스트리밍 데이터를 전송 받는 단계로 이를 위하여 BeeHive 내부에서 미디어 데이터를 규격화된 크기의 데이터로 나누어 저장하고 이를 RPC 메소드를 통하여 요구하면 블록 단위로 데이터를 요구하고 전송하는 방식이다. 세 번째 단계는 미디어 데이터 블록의 BeeHive ID 리스트를 전달받아서 스트리밍 데이터를 직접 BeeHive로부터 받아오는 단계이다.

- 파일을 이용한 전달 방식

  BeeHive 쪽에서 미디어 데이터를 파일로 저장한 뒤 이를 RPC 인터페이스를 통하여 파일명을 전달하면, 이를 ISSA의 컨텐츠 관리자가 전달받은 뒤 미디어를 파일로 읽은 뒤 스트리밍한다. 이에 대한 인터페이스 구조는 다음 (그림 11)과 같다.



(그림 11) ISSA와 BeeHive 사이의 스트리밍 인터페이스 (방법 1)

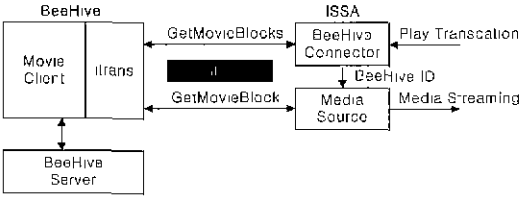
- RPC 메소드를 이용한 전달 방식

  RPC 메소드를 통하여 스트리밍 데이터를 전송 받는 단계로 이를 위하여 BeeHive 내부에서 미디어 데이터를 규격화된 크기의 데이터로 나누어 저장하도록 MovieObject를 변경하였다. 이렇게 변경된 MovieObject는 기본적으로 1Mbytes 단위의 블럭으로 저장된 미디어 데이터를 RPC 메소드를 통하여 요구하면 MovieObject 안에서 블록 단위로 데이터를 BeeHive 시스템에 요구하고 다시 이를 RPC 메소드를 통하여 전송하는 방식이다. 이를 위하여 새로운 인터페이스는 <표 5>와 같이 정의한다.

(그림 12)에서 나타난 대로 RPC 인터페이스를 통하여 미디어 블록의 개수를 얻어오고, 블록의 개수만큼

<표 5> MovieObject의 block 얻기

```
int GetMovieBlocks(BeeHive ID),
MovieBlock * GetMovieBlock(BeeHive ID,
                           int blockID),
```

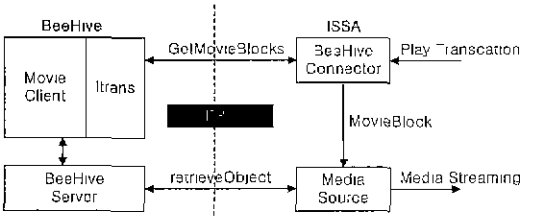


(그림 12) ISSA와 BeeHive 사이의 스트리밍 인터페이스 (방법 2)

RPC 인터페이스를 통하여 특정한 미디어 블록 데이터를 얻어오면서 미디어 데이터를 스트리밍하도록 되어 있다. 이것은 첫번째 방식에 비하여 직접 미디어 데이터를 얻어오기 때문에 진보적이긴 하나 블록 개수만큼 MovieObject를 생성해야한다는 점에서 오버헤드가 생기게 된다 따라서, 이러한 문제를 해결하기 위하여 세 번째 방식이 필요하게 되었다.

- BeeHive 객체의 기본 메소드를 이용한 전달 방식

  3번째 방식에서는 2번째 방식의 GetMovieBlocks 인터페이스를 수정하여 (그림 13)과 같이 미디어 블록을 직접 BeeHive 서버 시스템에서 얻어오는 방식으로 구조를 변경하고, 속도를 향상 시켰다



(그림 13) ISSA와 BeeHive 사이의 스트리밍 인터페이스 (방법 3)

컨텐츠 관리자 내부의 BeeHive 커넥터는 MovieBlock의 구조를 얻어오고, 실제 미디어 블록은 미디어 관리자에 소속되어 있는 MediaIO 클래스를 계승한 MediaIOBeeHive 클래스를 통하여 BeeHive 서버 시스템으로

부터 직접 얻어진다. 이것은 BeeHive 시스템에 저장되어 있는 미디어 객체의 추출을 단순화하여 실제 저장되어 있는 데이터를 가장 빠르게 얻어오는 방식이다. 여기서는 GetMovieBlock 인터페이스를 삭제하고, MediaSource에서 BeeHive Server의 표준 기본 메소드인 retrieveObject 메소드를 이용해서 직접 미디어 데이터를 획득함으로써 기존의 2번에 걸친 RPC 인터페이스를 한번의 RPC 인터페이스로 단축하였음을 보여준다.

6. 결 론

본 논문에서는 이기종 환경에서 다른 시스템과의 연동이 용이한 분산 스트리밍 시스템의 주요 모듈인 컨텐츠 관리자의 설계와 구현에 대한 개발 경험을 소개하였다. 제안된 컨텐츠 관리자는 서버가 제공하는 미디어 컨텐츠들에 대해 체계적인 관리 메커니즘을 제공하며 파일, 스케줄된 미디어 컨텐츠, 데이터베이스, 라이브 미디어 등 다양한 미디어 컨텐츠들을 관리할 수 있도록 하고 있다. 또한, 미디어에 대한 정보를 관리하며 자동으로 메타파일이나 URL을 생성함으로써 사용자에게 미디어에 대한 손쉬운 접근을 제공한다. 현재 컨텐츠 관리자는 데이터베이스로 BeeHive 만을 지원하고 있다. 따라서, 다양한 데이터베이스와의 연결을 위해 데이터베이스 커넥터를 동적으로 관리할 수 있는 메커니즘이 필요하다. 향후 표준 데이터베이스 연결 방법을 통하여 연결한다면 컨텐츠 관리자는 모든 유형의 미디어 컨텐츠를 효율적으로 관리할 수 있는 메커니즘이 될 것이다.

참 고 문 헌

[1] Chan-Gyun Jeong, Hyung-Ill Kim, Young-Rae Hong, Eak-Jin Lim, Sungyoung Lee, Jongwon Lee, Byeong-Soo Jeong, Doug-Young Suh, Kyoung-Don Kang, John A. Stankovic, and Sang H. Son, "Design for an Integrated Streaming Framework," Department of Computer Science, University of Virginia Technical Report, CS-99-30, November, 1999.

[2] 김형일, 이승룡, "멀티미디어 QoS를 위한 미디어 객체 구조의 설계", '98 한국정보과학회 춘계 학술발표 논문집, 1998년 4월, pp.699-701.

[3] 정찬균, 김형일, 홍영래, 임익진, 이승재, 이승룡, 심병수, "분산 스트리밍 시스템 설계", '99 한국멀티미디어학회 추계 학술발표 논문집, 1999년 11월, pp.338-343

[4] J. Stankovic, S. Son and J. Liebeherr, "BeeHive: Global Multimedia Database Support for Dependable, Real-Time Applications," In Proc. of Second Workshop on Active Real-Time Databases, Lake Como, Italy, September 1997.

[5] J. Stankovic and S. H. Son, "An Architecture and Object Model for Distributed Object-Oriented Real-Time Databases," *Journal on Computer Systems Science and Engineering*, Special Issue on Object-Oriented Real-Time Distributed Systems, Vol.14, No.4, pp.251-259, July 1999.

[6] H. Schulzrinne, A. Rao, and R. Lanphier, Real-Time Streaming Protocol (RTSP), IETF RFC 2326, April 1998.

[7] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, RTP: A Transport Protocol for Real-Time Applications, IETF RFC 1889, January 1996.

[8] Object Management Group, Control and Management of A/V Streams specification, OMG Document telecom/97-05-07 ed., October 1997.

[9] S. Mungee, N. Surendran, and D. C. Schmidt, "The Design and Performance of a CORBA Audio/Video Streaming Service," In Proc. of the 32st Hawaii International Conference on System Systems (HICSS), Hawaii, January, 1999.

[10] Cisco, IP/TV Content Manager : User Guide, http://www.cisco.com/univercd/cc/td/doc/product/softwarc/iptv30.

[11] Sonera, Sonera Live Content Manager, http://www.live.sonera.com/eng/tools/production_software.html.

[12] Banta Integrated Media, B・Media Content Manager, http://centrus.com/pr_media_body_2.html

[13] C. Aurrecoechea, A. T. Campbell, and L. Hauw, "A Survey of QoS Architectures," ACM/Springer Verlag Multimedia Systems Journal, Special Issue on QoS Architecture, Vol.6, No.3, pp.138-151, May 1998.

[14] S. N. Bhatti and G. Knight, "Enabling QoS adaptation decisions for Internet applications," Journal of Computer Networks, Vol.31, No.7, pp.669-692, March 1999.

[15] Microsoft Corps, Introducuon to DirectShow, http://www.microsoft.com/directx/dxm/help/ds/default.htm.

홍 영 래

e-mail : yrhong@oslab.kyunghee.ac.kr

1996년 경희대학교 전자계산공학과 졸업(학사)

1998년 경희대학교 대학원 전자계산공학과 졸업(공학석사)

1998년~현재 경희대학교 대학원 전자계산공학과 박사과정

관심분야 : 실시간 시스템, CORBA, 임베디드 시스템, 실시간 통신, QoS.

김 형 일

e-mail : hkim@oslab.kyunghee.ac.kr

1994년 경희대학교 물리학과 졸업(학사)

1996년 경희대학교 대학원 전자계산공학과 졸업(공학석사)

1996년~현재 경희대학교 대학원 전자계산공학과 박사과정

관심분야 : 실시간 시스템, 멀티미디어 시스템, 자바

이 승 룡

e-mail : sylee@oslab.kyunghee.ac.kr

1978년 고려대학교 재료공학과 졸업(학사)

1986년 Illinois Institute of Technology 전산학과(석사)

1991년 Illinois Institute of Technology 전산학과(박사)

1992년~1993년 Governors State University 조교수

1993년~현재 경희대학교 전자정보학부 부교수

관심분야 : 실시간 시스템, 실시간 고장허용시스템, 멀티미디어 시스템

정 병 수

e-mail : jeong@nms.kyunghee.ac.kr

1983년 서울대학교 전자계산기공학과 졸업(공학사)

1985년 한국과학기술원 전산학과(석사)

1995년 Georgia Institute of Technology, College of Computing(박사)

1985년~1989년 한국데이타통신(주) 경보통신연구소 선임연구원

1996년~현재 경희대학교 전자정보학부 조교수

관심분야 : 병렬 데이타베이스, 실시간 데이터베이스

윤 석 환

e-mail : voonsh@mail.iita.re.kr

1982년 아주대학교 산업공학과 졸업(공학사)

1984년 건국대학교 산업공학과 (공학석사)

1992년 품질관리 기술사 자격획득

1996년 아주대학교 산업공학과(박사)

1986년~1997년 한국전자통신연구원 책임연구원

1998년~현재 정보통신연구진흥원 책임연구원 (융자 1팀장)

1998년~현재 한국정보처리학회 이사 겸 학회지 편집위원장

관심분야 : 분산처리, 소프트웨어 공학, 품질보증, 개발체계, ERP

정 찬 근

e-mail : chan@software.or.kr

1979년 한국항공대학교 정자공학과 (학사)

1981년 서울대학교 대학원 전자공학과(석사)

1983년~1999년 한국전자통신연구원 근무

1999년~현재 한국SW진흥원 부장

관심분야 : 멀티미디어 데이타처리, 컴퓨터구조