멀티미디어 스트리밍 시스템을 위한 동기화 모델

정규수*, 유인호**, 김태형**, 김형진**
*엔코아컨설팅, **전북대학교
e-mail: kim@chonbuk.ac.kr

Synchronization Model for Multimedia Streaming System

Kyu-su Jung*, In-ho Ryu**, Tae-hyoung Kim**, Hyoung-jin Kim**
*Encore Consulting Ltd.
**Chonbuk National University

요 약

본 논문에서는 기존의 멀티미디어 스트리밍 시스템의 복잡하고 확장성이 떨어지는 문제점을 해결하기 위하여 새로운 컴포넌트를 생성하고 조합하여 작업을 동기화 할 수 있는 기법을 제안하였다. 이 기법은 복잡한 멀티미디어 처리 모듈을 단순한 선형적인 구조로 구성하여 버퍼링 모듈, 동기화 모듈, QoS 제어모듈, 비 호환 모듈간의 호환성을 보장하는 어댑터 모듈 등 효과적으로 확장 할 수 있게 설계함으로써 멀티미디어 스트리밍 시스템의 확장성과 이식성이 높아질 수 있도록 하고자 한다.

1. 서론

멀티미디어 통신, GIS(Geographical Information System), GPS(Global Positioning System), 무선 통신, 엔터테인먼트 등이 통합되면서 일반인들도 일상생활 속에서 필수품처럼 생각할 만큼 최신 기술들이 하나로 집적된 제품들을 쉽고 빠르게 만나 볼 수 있게 되었고, 유무선 통신 관련 시장이 활성화 되어 있다. 멀티미디어 관련 상품이나 서비스들이 인터넷을 통하여 쉽게 접근할 수 있는 특징 때문에 사용자들은 보다 많은 정보를 얻을 수 있고 편리해지면서 새로운 요구사항과 불만사항들이 빠르게 나타나며 이에 뒷받침할 수 있는 새로운 기술과 새로운 콘텐츠의 개발이 가속화되고 있고, 방송, 가전, DVD, VOD 등의 멀티미디어 서비스들이 인터넷으로의 통합 과정이 활발해지고 있다. 또한 멀티미디어 데이터의 압축 및 복원 기술, 멀티미디어 데이터 전송 및 에러 복원을 위한 채널 코딩 기술, 암호화, 대량 미디어 데이터의 효율적인 저장 및 검색 기술 등 방대한 기술들을 통합하기 위한 활동들이 활발히 이루어지고 있다. 멀티미디어 응용 분야의 핵심으로 디지털 비디오 압축 기술로 디지털방송, HDTV, DVD 등은 MPEG-2 비디오 압축 표준안을 사용하고 있고, MPEG-4는 현재 캠코더 폰의 비디오 압축 알고리즘 및 VOD 서비스에 이미 사용되고 있다. 이러한 과정에서 멀티미디어를 구성하는 데이터의 구조는 매우 복잡하고 다양해졌으며 각각의 미디어 간 상호작용을 효과적으로 지원하기 위한 좀 더 발전된 기술이 필요하게 되었다[1,2].

기존 연구들은 대부분 영상압축기술, 인코딩/디코딩기술, 실시간전송기술과 같은 멀티미디어 처리 기술에 초점을 두고 있으며, 이러한 시스템은 멀티미디어 데이터와 처리 기술 간의 결합도가 높고 미디어 동기화나 전송 모듈이 데이터 처리 기술과 종속성이 높기 때문에 시스템의 확장이나 유지 보수에 많은 어려움이 발생할 수 있는 문제점을 안고 있다[3,4,5].

이러한 문제점을 해결하기 위하여 본 논문은 복잡한 구조의 멀티미디어 스트리밍 시스템을 단순하고 기본적인 작업 구조를 설계하고, 이 구조를 바탕으로 확장성과 이식성을 높일 수 있는 시스템 설계에 목적을 두고 있다.

본 논문에서 제시하고자 하는 기본 모델은 단순한 기본 구조와 일관성 있는 처리과정을 정의하고 단순한 시스템을 구조에서 서비스의 목적과 요구사항에 따라 자유로운 구조 변경을 허용하여 시스템의 유지보수와 이식성이 높은 선형 태스크 집합(linear task set) 모델이다. 이 모델은 전체 시스템에서 처리해야 하는 모듈에 대하여 최소 작업 단위로 분할하고, 이를 연속된 작업 흐름으로 관리하도록 하였다. 또한 새로운 컴포넌트를 생성하고 조합하여 작업을 동기화 할 수 있는 기법을 제안하였다[6].

2. 컴포넌트 시스템 모델

다양한 멀티미디어 스트림을 지원하고자 할 경우, 다양한 포맷에 대한 모든 처리를 하는 것보다 하나의 목표 스트림으로 변환하고 그 스트림을 전송한 후 다시 스트림을 원하는 포맷으로 변환하는 트랜스코딩 방식과 MPEG-2에서 지원하는 공간해상도와 시간해상도의 제어 기법 두 가지를 적용할

수 있다.

스트리밍을 전송하는 서버에서 오디오와 오디오 소스에 대하여 다양한 포맷을 수용할 수 있도록 하기 위해 소스에 대한 포맷을 변환하는 트랜스코더(transcoder)는 MPEG-2 스트림으로 변환한다. 변환된 스트림은 디코더에 의해 DCT 양자화 계층을 통하여 공간 스캐일러빌러티(Spatial Scalability)를 지원한다.

QoS 맵퍼(QoS mapper)는 Combiner로 부터 프레임을 뽑아내고 패킷의 중요도와 사용자의 요구사항을 바탕으로 시간적 공간적 최적화된 QoS 파라미터 정보와 우선순위 정보를 설정한다. 맵퍼에서 사용자가 프레임 재생률이나 낮은 에러율을 선택할 수도 있으며, 이보다는 해상도에 더 중점을 줄 수도 있다. 이렇게 사용자의 요구사항과 패킷의 중요도에 따라 우선순위를 설정하고 재정렬 후 버퍼에 우선순위대로 저장한다. 또한, QoS 맵퍼는 디코더에 공간해상도와 시간해상도를 조정할 수 있는 파라미터를 제공하여 시공간 해상도를 조절하는 파라미터로 사용할 수 있게 한다.

전송컨트롤러는 전송할 패킷을 버퍼로부터 꺼내어 QoS 파라미터를 체크하고 요구 사항에 맞게 최적화된 전송한다. QoS에서 설정한 패킷에서 타임스탬프와 우선순위 정보를 바탕으로 우선순위 스트림 전송을 한다. 네트워크 상태가 지정된 시간 내에 패킷이 도착하지 못할 상태라고 판단될 때에 우선순위가 낮은 패킷에 대해 폐기한다.

클라이언트에서는 네트워크 컴포넌트에서 수신되는 이벤트를 모니터링 하기 위한 출력포트가 활성화 속성을 갖고 있다. 수신되는 패킷들은 타임스탬프 순서대로 패킷들을 정렬한 후 버퍼에 저장한다. 버퍼에 데이터가 시스템에서 정해진 상한과 하한치의 버퍼량에 도달하면 트랜스코딩을 시작하도록 이벤트를 통보한다. 버퍼는 상한치와 하한치를 모니터하면서 지속적으로 이벤트를 통보하여 다른 컴포넌트들이 참고하도록 한다. 트랜스코더와 디코더는 정렬된 패킷을 재생 버퍼로 푸시(push)하여 MPEG이나 이미지 포맷으로 패킷 데이터를 변경한다. 변경된 데이터는 복합기를 통하여 동기화기(synchronizer)에 전달한다. 동기화기는 오디오와 비디오 스트림의 타임스탬프를 이용하여 동기화 작업을 하며. 내부 버퍼가 비워지면 입력포트는 복합기에서 스트림이 존재하는지 확인하고 없으면 버퍼가 비었음을 시스템에 통보한다. 그림 1에서는 전체 시스템에 대한 구성 모델을 보여준다.

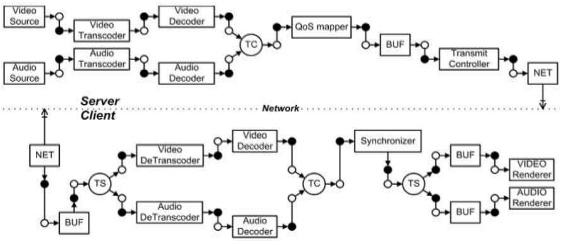

그림 1. 컴포넌트 시스템 구성 모델

3. 동기화 모델

3.1 활성화 포트 동기화 개념

본 논문에서는 동기화를 위하여 2장에서 제안한 시스템 모델을 기반으로 동기화 구조를 제안하고자 하는 것이다.

그림 2의 모델에서 active port를 나타내고 있으며 이 지점이 이 모델에서 중점을 두고 싶은 지점임을 알 수 있다. 먼저, 정보를 생산해 내는 SRC에서부터 보면, SRC(Source)와 TS(Tee Splitter)에서는 TS가 SRC로 데이터를 끌어오는 지점이 있고, TS는 출력에는 관심이 없다. Decoder에서는 TS에 데이터를 가져와서 디코딩 하며 비디오/오디오 컨트롤러(Video/Audio Controller)의 디코딩된 데이터를 요구를 받게 된다.

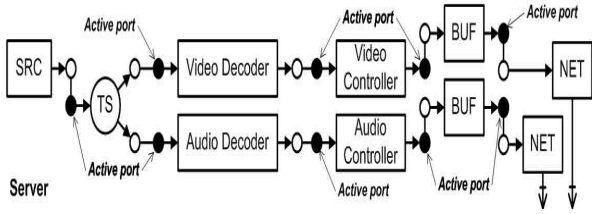

그림 2. 시스템 모델상의 active port

컨트롤러는 디코딩된 데이터를 요구하면서 버퍼에 데이터를 저장하는 요구를 한다. 입출포트 모두 활성화 포트이며 입력 포트에서 데이터를 끌어올리기 위하여 디코더의 출력포트에 풀(pull)모드 인터페이스를 호출할 것이다. 입력포트에서 끌어올린 데이터를 전송하기 위하여 QoS 정보나 프레임에 대한 타임스탬프, 우선순위 등의 정보를 캡슐화 하여 패킷을 구성한 후 출력포트에서 버퍼 컴포넌트의 입력포트의 푸시(push) 모드 인터페이스를 호출하여 데이터를 전달할 것이다. 즉, 스트리밍 시스템에서 전체 시스템의 동기화 클럭 기준이 입출력 포트 모두 활성화된 컴포넌트들에게 초점을 맞출 수밖에 없다. 왜냐하면 디코더가 임의로 소스 컴포넌트에서 스트림을 읽어 처리한 후 처리된 출력 스트림을 다음 컴포넌트로 보내려고 할때 다음 컴포넌트가 무엇을 하고 있는지 알 수가 없다. 물론 서로 버퍼상태나 작업 상태에 대하여 주고받을 수 있지만 디코더가 프레임을 처리할 때 마다 다음 컴포넌트의 상태를 항상 모니터링 하고 있어야 한다는 것이다. 따라서 한쪽만 활성화된 컴포넌트들은 양쪽이 활성화된 컴포넌트들의 서비스 요구에 따라 바로 처리해서 전달해주는 것이 더 효과적이고 요구 이벤트가 발생했을 때 바로 처리해서 전달해 주기 때문에 버퍼가 작아질 수 있는 장점도 있다.

입출력 모두 활성화 포트를 갖고 있는 컴포넌트를 활성화 컴포넌트라고 하고, 그 외의 컴포넌트를 비활성화 컴포넌트라고 하자. 시스템을 구성하는데 활성화 컴포넌트는 한 개 이상 존재할 수 있으며 이 활성화 컴포넌트들의 동기화 기준 클럭을 맞추게 되면 비활성화 컴포넌트들은 요구에 반응하는 컴포넌트들이기 때문에 전체 시스템의 효과적인 동기화를 기

대할 수 있다. 그림 1의 모델에서 활성화 컴포넌트가 한 개 이상 존재할 경우 이 컴포넌트 사이에는 반드시 비활성화 컴포넌트가 존재하며, 또한 입출력 포트 모두 비활성화 포트를 갖고 버퍼와 분해역할을 하는 TS(Tee Splitter) 컴포넌트가 존재하는 것을 볼 수 있다. 경우에 따라서 활성화 컴포넌트가 연결될 수도 있다. 이 경우는 서로 푸시(push), 풀(pull) 모드를 지원하는 인터페이스를 구현하고 있는 컴포넌트이어야 한다.

제어 인터페이스는 두 집합의 속성을 갖는다. 컴포넌트 자체의 속성과 컴포넌트를 통한 정보흐름의 속성을 갖는다. 구별하기 위하여 특정 네트워크에 접속된 환경에서 대역폭은 기본적인 네트워크의 속성이다. 따라서 대역폭이 변동될 경우 실제 데이터 흐름은 시스템의 요구에 따라 다양하게 변하게 된다. 각 활성화 포트들이 컴포넌트 동기를 위하여 그림 3과 같이 활성화 포트에서 동기화 처리 컴포넌트로 이벤트를 통보하는 것을 보여 주고 있다.

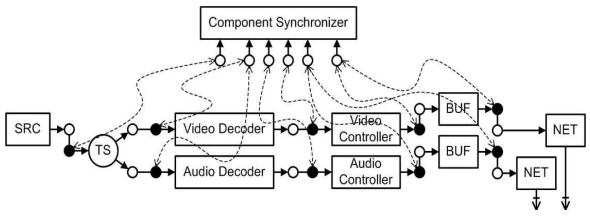

그림 3. active port의 이벤트 통보 흐름

3.2 이벤트 모델

컴포넌트와 컴포넌트는 서로 연결되어 상호 통신을 한다. 컴포넌트는 정보를 요구할 수도 있고 정보를 전달할 수도 있고 이웃하는 컴포넌트 간의 조절을 위한 활동도 필요하다. 하지만 이 작업은 컴포넌트간의 액션의 관계를 의미하며 전제 컴포넌트들의 상태에 따라 전체 시스템에 받는 영향에 대해서는 고려되지 않았다. 다시 말해서 컴포넌트들의 상호 활동은 자신의 상태에 따라 반응하며 발생하는 시점에 대해서는 정의된 것이 없었다. 자신이 처리할 작업이 지연되거나 빨리 처리 되었을 때에 서로에게 반응하는 시점이 명확해야 한다. 즉, 컴포넌트 상호 활동은 능동적으로 활동할 수 있는 처리가 있지만 많은 컴포넌트들에 대한 통보가 필요하다는 것이다.

이벤트는 전체 시스템에 대한 다양한 이벤트로부터 이벤트의 발생 여부를 확인하고 이벤트 핸들러로 다중 수신할 수 있어야 한다. 이벤트는 작업스케줄링, 메시지 큐 스케줄링과 같은 모듈에 절대 시간을 제공하여 서로 다른 컴포넌트들 간의 동기화 기준을 설정하였다.

3.3 이벤트 맵 구조

이벤트 종류는 시스템의 목적과 범위에 따라 다양하고 전달되는 정보의 크기도 다양하다. 컴포넌트는 OOP(Object Oriented Programming) 개념을 반영하고 있고, OOP가 갖는 특성을 모두 갖고 있다. 작업 흐름과 컴포넌트의 작업 구조가 기본적으로 선형으로 구성되어 있지만 절대적으로 선형은 아니다. OOP 개념으로 비추어 볼 때 상속이 가능한 컴포넌트에서는 하나의 컴포넌트는 부모 컴포넌트의 기능을 활용할 수 있다.

컴포넌트 A, B, C가 연결되어 있는 상태이고 컴포넌트 A→B→C 순으로 작업이 진행된다면 컴포넌트 테이블에서의 컴포넌트들의 이벤트 메시지 맵에 대한 리스트를 관리하게 된다. 컴포넌트 A가 생성이 되어 컴포넌트 테이블에 등록이 되었다. A는 자신이 처리하거나 받아야할 이벤트에 대하여 등록하게 된다. 그림 4에서 컴포넌트 A가 EVENT_ITEM이라는 노드들의 리스트를 EVENT_MAP이 갖고 있게 된다. A에서는 자신의 이벤트들에 대하여 EVENT_ITEM에 저장하고 이 이벤트에 대한 핸들러가 존재한다면 그 핸들러의 참조값을 포함할 수 있다. EVENT_ITEM의 리스트는 헤더 역할을 하는 EVENT_MAP에 의해서 관리된다. 컴포넌트 B와 C도 자신의 이벤트에 대하여 EVENT_MAP을 관리하게 된다. 컴포넌트 A, B, C 가 시스템을 구성하기 위하여 등록이 되고 A→B→C 순으로 연결이 될 때 그림 4처럼 컴포넌트 B의 EVENT_MAP의 한 속성이 컴포넌트 A의 EVENT_MAP을 참조한다. 즉, 컴포넌트의 내부적인 이벤트는 EVENT_ITEM에 대한 정보는 EVENT_MAP에 의해서 관리되고, 전체 컴포넌트들의 이벤트는 EVENT_MAP을 노드로 하여 리스트가 형성이 된다. 이 EVENT_MAP의 리스트를 관리는 컴포넌트 테이블에서 관리된다.

컴포넌트는 자신의 이벤트만을 관리하지만 컴포넌트가 연결되는 시점에서 자신의 이벤트 맵을 등록함으로써 컴포넌트 간의 A←B←C 방향으로 참조하는 이벤트 맵을 구성할 수 있게 된다. 따라서 컴포넌트에 대한 이벤트는 컴포넌트 이벤트 맵에 의존하여 처리하게 되는 것이다. 이벤트에 반응하기 위하여 컴포넌트의 메서드 테이블을 검색하지 않을 것이며, 이벤트에 대한 핸들러가 존재한다면 바로 핸들러를 호출하여 처리할 수 있는 장점을 갖고 있다. 또한 컴포넌트가 많이 연결되어도 이벤트 맵은 모두 연결되어 있기 때문에 이 이벤트 맵을 등록하고 이벤트를 검색하여 해당 핸들러를 호출하는 작업을 하나의 컴포넌트가 집중적으로 처리할 수가 있으며 이 컴포넌트는 최상위 컴포넌트로 하고 이벤트를 받을 컴포넌트들은 이 최상위 컴포넌트를 상속 받기만 하면 된다. 이벤트 맵에는 플랫폼에서 지원하는 이벤트도 적용될 수 있고 컴포넌트와 시스템과의 통보를 목적으로 하는 이벤트도 적용될 수 있다

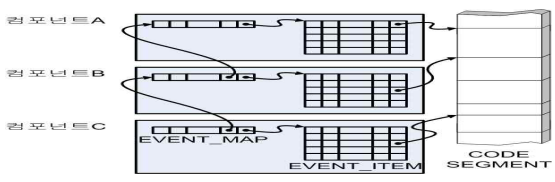

그림 4. 연결된 컴포넌트들의 이벤트 맵 구조

3.4 동기화 관리 모델

멀티미디어 스트리밍 시스템에서는 멀티미디어 데이터 처리 프로세싱과 데이터 전송 프로세싱에서 플랫폼의 성능을 최대한 활용해야하는 부분이 스레드 관리가 필수적이다. 스레드는 광범위한 운영체제 간의 기술적 차이점을 숨기려는 경향 때문에 운영체제에 따라 관리 기법이 다르기 때문에 커널상의 스레드 관리와 라이브러리 기반에서 스레드를 관리하는 방식에 따라 의존적일 수밖에 없다.

멀티스레드 모델은 시스템의 CPU를 최적으로 사용할 수 있게 함으로써 서비스의 속도향상을 가져올 수 있는 모델로서 다음과 같은 근본적인 문제를 안고 있다.

① 컨텍스트 스위칭(Context Switching)이나 멀티프로세서 시스템의 경우 CPU사이에 데이터 이동, 동기화 이유로 효과가 떨어질 수 있다.
② 복잡성, 동시성 제어를 요구한다. 스레드 간 협업이나 스레드 스케줄링과 같은 복잡한 문제에 대한 제어와 효율적인 정책을 요구하게 된다.
③ 멀티스레드를 지원하지 않는 운영체제가 있거나 모든 운영체제가 동일한 이식성을 제공하지 않는다. 임베디드 시스템(embedded system)의 경우 멀티스레드를 제공하지 않는 시스템이 종종 있으며 그렇지 않다 하더라도 시스템에서 제공하는 스레드 모델이 서로 다르므로 이로 인한 기능 및 성능차이로 어려움을 겪을 수 있다.
④ 가용자원에 접근하는 것에 대한 최적화를 요구한다. 여기에서 가용한 자원이란 CPU나 디스크립터 테이블과 같은 시스템 자원을 말한다. 이것은 클라이언트의 요청을 처리하는 것에 대한 고민을 하는 비용보다 더 큰 개발 비용이 요구될 수 있다.

성능이 치명적이지 않은 응용 프로그램이나 동기화 문제가 감당하기 힘들 정도로 공유 자원을 너무 많이 요구하는 응용 프로그램이나 요청이 자주 유입되지 않는 응용 프로그램과 같은 경우엔 싱글 스레드로 구성하는 것이 더 바람직하다. 컴포넌트들 사이의 연결이 동일해야 한다. 컴포넌트 자체의 연결 비용이 무의미할 정도로 작아야하며 단일 프로시저 호출이나 낮은 비용을 갖는 메서드 호출로 취급되어야 한다. 원격 프로시저 호출이나 블록킹(blocking)될 수 있는 메서드로 처리할 경우 잠재적인 지연이나 대기시간으로 큰 부하를 가질 수 있다.

스레드는 그림 5와 같은 상태 변환을 갖는다. 스레드는 실행 준비가 되면 준비상태가 된다. 준비상태가 된 스레드는 자신의 우선순위가 가장 높아지기 전까지는 실행되지 않는다. 우선순위가 높아지면 해당 스레드를 실행하고 그 스레드는 실행상태가 된다. 우선순위가 더 높은 스레드가 준비되면 실행 중이던 스레드는 다시 준비상태로 돌아오고 준비, 실행 상태간의 전환은 스레드가 선점할 때마다 일어난다.

특정 시점에 단 1개의 스레드만 실행되는 상태이다. 이는 실행 상태의 스레드가 실제 프로세서를 선점하여 제어권을 갖은 상태를 의미한다. 스레드가 일시중지 상태이면 실행 대상이 아니다. 이는 디스크 I/O나 네트워크상에서 데이터를 수신을 기다리거나 메시지 큐, 세마모어(semaphore), 뮤텍스(mutax), 이벤트, 메모리 할당 대기 등 명시적인 일시정지 상태이다. 일시 중지 상태가 풀려나면 스레드는 다시 준비 상태로 돌아가 스케줄링 대상이 된다.

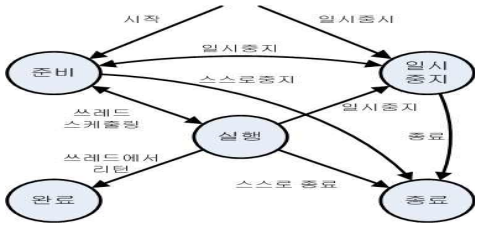

그림 5. 스레드 상태도

스레드가 완료된 것은 자신이 할 일을 다 마치고 처리하는 구역을 벗어나 리턴 되었다는 의미이다. 처리 규칙은 스레드가 생성하는 동안 명시된 코드 구역이며 완결 상태의 스레드는 다시 실행될 수 없다. 스케줄러는 준비대상 리스트에서 우선순위가 가장 높고 가장 오래 기다리고 있던 스레드를 깨우고 실행중인 스레드가 임의의 이유로 인터럽트 되면 해당 스레드의 콘텍스트(context)정보를 보관하고 꺼낸 스레드는 실행 상태로 가며 인터럽트 당한 스레드는 대기 목록에 들어간다. 중시 스레드 목록에 있는 스레드 실행 대상이 아니며 대부분 자원 할당을 위해 대기하거나 정기적인 수면 상태이거나 명시적인 일시중지 되었기 때문이다.

4. 결론

본 논문에서 제안한 컴포넌트 모델은 복잡하고 모듈간의 결합도가 높아질 수 있는 문제를 빠르게 찾아내고 재구성할 수 있는 장점을 갖고 있다. 이 컴포넌트 모델은 시스템을 상세히 묘사할 수 있으며 반대로 컴포넌트들 결합하여 거시적인 시점에서 시스템을 볼 수 있었다. 이는 멀티미디어 시스템에 대한 효과적인 요구 분석과 설계된 모델들을 서로 쉽게 비교 및 분석이 가능하여 시스템의 동기화 처리나 자원 관리 관점에 중점적으로 관리해야할 지점을 찾아내는데 효과적이었다. 향후 성능평가를 통해 제안기법이 우수함을 보이고자 한다.

참고문헌

[1] Steven Bogaerts, David Leake, “IUCBRF : A Framework For Rapid and Modular Case-Based Reasoning System Development”, IUCBRF, TECHNICAL REPORT 617, Aug. 2004
[2] A. P. Black, J. Huang, R. Koster, J. Walpole, C. Pu, “Infopipes : An Abstraction for Multimedia Streaming”, Multimedia System Vol. 8, pp. 406-419,

Jan. 2002.

[3] Feng Wu, Honghui Sun, Guobin Shen, Shipeng Li, Ya-Qin Zhang, Bruce Lin, Ming-Chieh Li, “SMART: An Efficient, Scalable and Robust Streaming Video System”, EURASIP Journal on Applied Signal Processing, Special issue on Multimedia over IP and Wireless Networks, Vol. 2, pp 192-206, Jan. 2004.

[4] ISO/IEC, “Text of Propesed Draft Techinical Report(PDTR)2”, ISO/IEC JTC1SC29/WG11, MPEG01/N3939, Jan. 2001.

[5] Bobby Woolf, Richard Turner, Gregor Hohpe, “Enterprise Integration Patterns : Designing, Building, and Deploying Messaging Solutions”, Addison-Wesley Professional, Oct. 2003.

[6] 정규수, “DirectShow Framework을 기반으로 한 분산 멀티미디어 스트리밍 시스템 설계 및 구현”, 군산대학교 박사학위논문, 2007