# 실시간 멀티미디어 스트리밍 서비스를 위한 Time-triggered Message-triggered Object 기반의 프레임워크 및 동기화 메커니즘

조 은 환[†] · 김 문 회[††]


## 요 약

본 논문에서는 실시간 객체모델인 Time-triggered Message-triggered Object (TMO)를 사용하여 분산 실시간 멀티미디어 스트리밍 서비스를 효과적으로 개발 할 수 있는 새로운 소프트웨어 프레임워크와 스트림 동기화 메커니즘을 소개한다. 본 프레임워크의 목적은 개발자로 하여금 복잡한 실시간 멀티미디어 스트리밍 서비스를 쉽게 설계하고 적시 스트리밍 기능들을 제공하는데 있다. 이를 위해서 본 프레임워크는 Multimedia Streaming TMO, MMStream TMO Support Library 그리고 TMO Support Middleware로 구성된다. 특히, MMStream TMO와 동기화 기법이 제공하는 time-triggered 특성은 QoS 보장이 어려운 통신 채널과 시스템 환경에서 불규칙한 미디어 데이터 전달 및 처리하는 역할을 실시간으로 수행한다. 결론적으로 본 논문에서 제시한 프레임워크의 적시 서비스 능력은 향후 실시간 멀티미디어 스트리밍 서비스 개발에 기여할 것으로 기대된다.


**키워드 : TMO, 실시간 멀티미디어 스트리밍, 프레임워크, 동기화, MMStream TMO**

# A Framework and Synchronization Mechanism for Real-time Multimedia Streaming Services based on the Time-triggered Message-triggered Object

Eun Hwan Jo[†] · Moon Hae Kim[††]


## ABSTRACT

In this paper, we present a new framework and stream synchronization mechanism to effectively support developing real-time multimedia streaming services by using a real-time object model named TMO (Time-triggered Message-triggered Object). The purpose of the framework is twofold. Firstly, the framework helps developers to design complex distributed real-time multimedia streaming services. Secondly, it supports timely streaming facilities. In order to achieve these goals, our framework is consist of Multimedia Streaming TMO, MMStreaming TMO Support Library and TMO Support Middleware. The time-triggered spontaneous feature of the MMStream TMO and a global-time based synchronization scheme is used as a regulator against the irregular deliveries and processing of media units caused by QoS non-guaranteed systems and communication channels. In conclusion, timely service capability of our framework is expected to contributed to overall enhancement of the real-time multimedia streaming.


**Key Words : Time-triggered Message-triggered Object, Real-time Multimedia Streaming, Framework, MMStream TMO**

## 1. 서 론

21세기 IT 기술은 디지털 기술의 발전과 초고속 네트워크의 확산으로 컴퓨터, 방송, 가전, 통신기술 등이 상호 결합 및 접속하여 인간에게 실시간 정보서비스를 제공할 수 있는 형태로 빠르게 진화하고 있다. 이 같은 추세는 다양한 분산 멀티미디어 서비스에 대한 수요를 가중시키고 있으며, 신뢰성 있는 서비스를 제공하기 위해서 실시간 멀티미디어 처리 기술을 필요로 하고 있다. 분산 멀티미디어 서비스가 궁극적으로 지향하는 것은 선명한 화질과 뚜렷한 음성을 유·무선 네트워크를 통해 사용자들에게 신뢰성 있게 전달하는 것이다[6, 7]. 이를 위해서 분산 멀티미디어 서비스는 디스크에 저장된 미디어 또는 실시간으로 디지타이징(digitizing)된 미디어를 연속적인 흐름상에서 처리 및 전달하는 스트리밍 기술을 사용한다[1]. 하지만, 기존의 멀티미디어 스트리밍 기술

† 준 회 원 : (주)동부정보기술 미래기술연구소 연구원
†† 정 회 원 : 건국대학교 정보통신대학 컴퓨터공학부 교수
논문접수 : 2006년 7월 19일, 심사완료 : 2006년 10월 9일

은 새로운 형태의 멀티미디어 서비스를 개발하기에 역부족이다. 특히, 다자간 영상회의(multipoint video conferencing), 원격진료(telemedicine), 그리고 텔레매틱스(telematics)와 같은 복잡한 분산 실시간 멀티미디어 서비스 개발은 더욱더 어렵다. 그 이유는 다음과 같다. 첫째, 대부분의 멀티미디어 스트리밍 기술은 "guarantee"서비스가 아닌 "best of quality" 서비스를 제공하기 때문에 QoS(Quality of Service) 및 동기화 보장이 힘들다. 즉, 실시간 멀티미디어 스트리밍 서비스 능력이 부족하다. 둘째, 멀티미디어 스트림에 대한 입력 및 출력 등의 기본적인 기능들로 구성된 단순 라이브러리는 리눅스에서 동작하는 분산 멀티미디어 서비스 개발을 어렵게 한다. 따라서 본 논문에서는 적시 서비스 능력(timely service capability)을 갖춘 멀티미디어 스트리밍 서비스를 효과적으로 개발할 수 있는 TMO(Time-triggered Message-triggered Object) 기반의 실시간 멀티미디어 스트리밍 지원 프레임워크와 동기화 메커니즘을 설계 및 구현하였다.

본 논문의 내용을 살펴보면 다음과 같다. 우선 2장에서는 TMO 스킴에 대해서 간단히 설명하고, 3장에서는 TMO 기반의 분산 실시간 멀티미디어 서비스를 위한 프레임워크 아키텍처를 소개한다. 4장과 5장에서는 글로벌 시간 기반의 동기화 메커니즘과 실험결과를 기술한다. 마지막으로 6장에서는 본 논문의 결론과 향후과제를 제시한다.

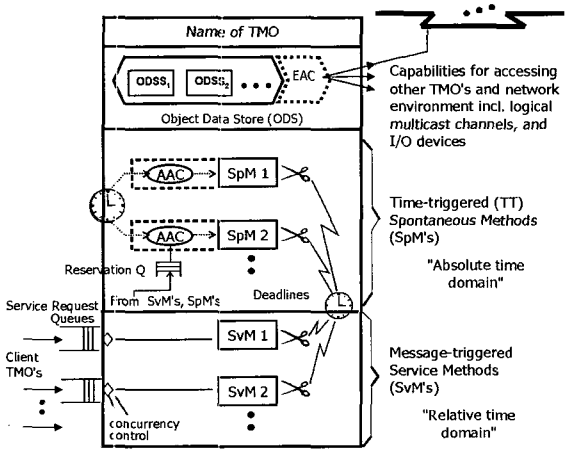
## 2. Time-triggered Message-triggered Object

TMO는 Time-triggered Message-triggered Object의 약자로써, Kane Kim 등에 의해서 개발된 Object Structuring Scheme이다[4,5]. TMO는 기존의 객체 모델을 의미적으로 확장한 모델로써, 실시간 시스템이 가지는 시간적인 특성과 행동을 쉽게 추상화 할 수 있는 구조를 가진다.

TMO 모델의 구조는 크게 다음 4개의 부분으로 구성된다.

- Object Data Store (ODS) : 실시간 데이터를 저장하기 위한 부분으로 Object Data Store Segment (ODSS) 단위로 관리된다. ODSS는 TMO 메소드인 SpM과 SvM에 의해서 상호배타적으로 접근 가능하다.
- Environment Access Capability (EAC) : 외부와의 논리적인 통신 채널, 그리고 I/O 디바이스 인터페이스 등에 대한 연결 창구다.
- Spontaneous method (SpM)
- Service method (SvM)

(그림 1)은 TMO 객체 모델의 기본적인 구조를 보여주고 있으며, 기존의 객체 모형과는 다른 다음과 같은 특징들을 가지고 있다




(그림 1) TMO 기본 구조

(1) 분산 컴퓨팅 컴포넌트

멀티노드의 TMO 객체들은 non-blocking 형태의 remote method call을 통하여 분산 처리를 수행한다.

(2) Spontaneous method (SpM)

Time-triggered method인 SpM은 클라이언트의 서비스 요청에 의해서 실행되는 SvM과는 달리 TMO 설계 시에 명세한 시간이나 주기가 되면 실시간 클록(clock)에 의해 자동으로 실행되는 메소드다. SpM의 시간 조건은 디자인 시에 AAC(Autonomous Activation Condition)에 상수로 명세 된다. 아래는 AAC 명세 방법에 대한 예시를 보여주고 있다.

"***for*** t = ***from*** 9:00am ***to*** 9:30am ***every*** 20min ***start-during*** (t, t+1min) ***finish-by*** t+5min"

(3) Basic concurrency constraint (BCC)

TMO들의 시간적인 서비스 능력을 보장하기 위한 제약 조건으로써, SpM과 SvM이 공유데이터 ODS를 동시에 접근하려고 할 때 발생할 수 있는 충돌을 방지하기 위한 수행 규칙이다. 이때 SpM이 SvM 보다 더 높은 우선순위를 갖는다.

(4) 종료시간과 데드라인 보장

디자이너가 메소드의 시작시간, 종료시간 그리고 데드라인을 명세함으로써 시스템의 적시 서비스 능력을 디자인 단계에서 보장할 수 있도록 지원 한다.

## 3. TMO 기반의 실시간 멀티미디어 스트리밍 지원 프레임워크

### 3.1 Time-triggered 멀티미디어 스트리밍

Time-based 미디어란 데이터 처리에 있어서 시간 요소가 중요한 의미를 갖는 미디어로써 오디오, 비디오, 미디(MIDI) 그리고 애니메이션과 같은 미디어가 여기에 포함된다. Time-based 미디어는 시간 종속적이고 연속적(continuous)인 특성을 가지고 있기 때문에 엄격한 시간제한 속에서 전달 및 처리될 수 있어야한다. 이때, 분산 네트워크 환경에서의 time-based 미디어 처리를 멀티미디어 스트리밍(multimedia streaming)이라 하고, 네트워크로 송수신 되는 패킷의 시퀀스(sequence)인 데이터 스트림(stream)을 연속적으로 처리하는 기술로 정의할 수 있다. 마이크로소프트, 리

얼네트웍스 그리고 썬마이크로시스템즈에서 제공하는 멀티미디어 스트리밍 기술들은 상위 수준의 라이브러리 또는 소프트웨어 패키지 형태로 제공하기 때문에 내부적인 time-based 미디어 처리 구조를 파악하기 어렵다. 반면에 코드가 공개되어 있는 리눅스 경우 대부분의 멀티미디어 스트리밍 응용프로그램은 시간이 아닌 이벤트 중심에서 time-based 미디어를 처리하고 있기 때문에 처리의 일관성이 부족하고 스트리밍 제어가 힘들다. 이것은 미디어 처리에서 발생하는 비동기 요소를 적절하게 제어할 수 없기 때문에 서비스 품질과 신뢰성에 부정적 영향을 미칠 수밖에 없다. 결국, 이와 같은 문제는 점점 더 복잡해지는 분산 멀티미디어 스트리밍 서비스 개발을 힘들게 할 것이다.

본 논문에서 제시한 Time-triggered 멀티미디어 스트리밍 또는 time-triggered 미디어 처리란 TMO 기반의 time-based 미디어 처리 기술로써, TMO의 time-triggered 특성에 따라 미디어 데이터의 입력, 필터링, 그리고 출력이 하나의 시간 축 상에서 제어 및 처리되는 실시간 스트리밍 기술이다. 본 스트리밍 기술을 실현하기 위해서 본 논문에서는 TMO를 멀티미디어 스트리밍에 맞도록 확장한 MMStream TMO(Multimedia Streaming TMO)를 정의하였으며, MMStream TMO의 객체 멤버인 SpM 메소드가 time-based 미디어를 처리하게 된다. 본 스트리밍 방식은 기존의 이벤트 중심의 미디어 처리와는 달리, 시간 중심으로 미디어를 처리하기 때문에 전체 스트리밍 시간 분석이 용이한 것이 가장 큰 특성이자 장점이다.

### 3.2 MMStream TMO 구조 및 특성

MMStream TMO는 실시간 멀티미디어 서비스 개발에 적합한 실시간 분산 객체로써, 분산 네트워크 환경에서 하나의 연속된 멀티미디어 스트림을 연결하고 서비스한다. MMStream TMO는 Source SpM과 Sink SpM을 기본적으로 포함해야 하고, 제어 메시지 채널과 데이터 전송 채널을 통해서 서비스 연결과 멀티미디어 전송을 가능하게 한다.

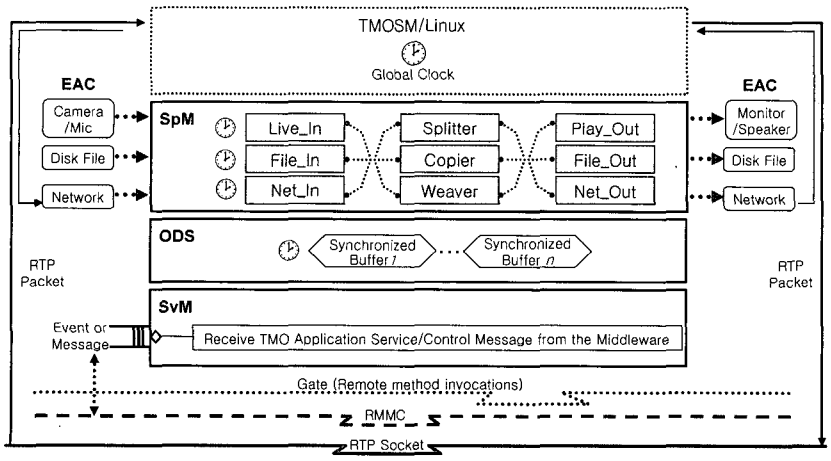
Time-triggered 멀티미디어 스트리밍 방식에서 하나의 SpM은 로컬 시스템 내의 시간 기반의 미디어 처리를 담당한다. 그리고 분산 멀티미디어 스트리밍 서비스를 위해서, 송신 측의 SpM과 수신 측의 SpM은 네트워크를 통해서 하나의 데이터 스트림을 송수신 및 처리한다. 이때의 연결을 스트림 채널이라 하며, 데이터 스트림의 입력 및 전송을 담당하는 SpM이 Source SpM이고 수신 및 재생을 담당하는 SpM이 Sink SpM이다.

SpM은 실시간 객체 모델인 TMO를 구성하는 하나의 실시간 멤버 함수이기 때문에 TMO 객체 내부에서 선언되고 정의됨으로써 사용이 가능해진다. 마찬가지로 time-triggered 멀티미디어 스트리밍을 실현하기 위해서는 Source SpM 또는 Sink SpM을 멤버로 가질 수 있는 MMStream TMO 객체가 필수적이다. 그리고 각 SpM을 포함하는 MMStream TMO 객체가 네트워크로 연결되었을 때 하나의 완전한 멀티미디어 스트리밍 채널이 완성된다.

MMStream TMO는 멀티미디어 스트림을 실시간으로 처리하도록 만들어진 공장과도 같아서, TMO 실행 엔진인 TMOSM/Linux 서비스를 지원 받아서 정해진 시간에 자동으로 멀티미디어 스트림을 처리한다.

(그림 2)는 MMStream TMO의 기본 아키텍처로써, 주요 구성요소는 다음과 같다.

- SpM : MMStream TMO의 멤버 메소드로 선언되는 SpM 들은 time-based 특성을 가진 미디어 처리를 담당하는 메소드다. 하나의 SpM은 멀티미디어 서비스 특성에 따라 미디어 데이터의 입력, 처리, 그리고 출력 등의 기본적인 기능을 가진 객체 라이브러리를 사용하여 구현되며, 한개 이상의 SpM이 MMStream TMO 내에 선언되어 사용될 수 있다. 그리고 각각의 SpM들은 자신의 미디어 처리에 맞는 AAC에 따라 spontaneously 하게 처리한다.
- ODSS : MMStream TMO 내에서 2개 이상의 스트리밍 채널이 필터링을 통해서 하나의 채널로 합쳐지거나, 하나의 채널이 서로 다른 미디어로 분리되어 처리될 때 발생하는 비동기를 해결하기 위한 버퍼 역할을 담당한다.




(그림 2) MMStream TMO

즉, 동기적인 미디어 처리를 위해서 timed queue 역할을 하는 미디어 데이터 저장소다

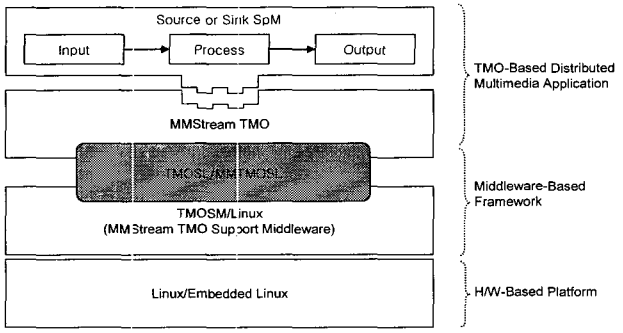
- SvM : 분산 노드에 위치한 MMStream TMO 객체로부터 스트리밍 서비스 요청과 제어 메시지를 수신 및 처리하는 메소드다. SvM은 분산 미들웨어인 TMOSM/Linux로부터 들어오는 원격 메소드 호출을 통해서 활성화되며, 프로그램 이벤트를 네트워크 메시지로 변환하여 원격 메소드 호출을 하기도 한다. 이때, 제어 메시지 채널로 Gate와 RMMC[5]를 사용한다.

### 3.3 MMStream TMO 응용 프레임워크

TMO 기반의 실시간 멀티미디어 스트리밍 서비스를 위한 프레임워크는 개발자가 복잡한 응용프로그램 구조에 대한 부담을 덜고 신뢰성 있는 서비스를 쉽게 설계 및 구현할 수 있도록 지원하는 소프트웨어를 말한다. 본 논문에서 추구하는 프레임워크는 리눅스 플랫폼에서 다양한 디바이스와 미디어를 지원하고 실시간 멀티미디어 응용 개발에 있어서 유연성(flexibility)과 적응성(adaptation)을 갖추고 있으며, 적시 서비스 능력을 제공할 수 있는 프레임워크다. 본 프레임워크의 주요 목적은 크게 두 가지다. 첫 번째는 복잡하고 실시간 시스템 성격이 강한 분산 멀티미디어 응용프로그램을 쉽게 설계 및 구현할 수 있도록 지원하는 것이다. 이를 위해서 본 프레임워크는 실시간 객체 모델인 TMO 모델을 멀티미디어 시스템 설계에 적당하도록 확장한 MMStream TMO 모델을 제공함으로써, 시스템 디자이너가 분산 멀티미디어 스트리밍 서비스를 정확하고 일관되게 설계할 수 있도록 지원한다. 또한, 다양한 미디어 처리 및 네트워킹에 대한 하부구조를 추상화하여 고수준의 API로 제공함으로써 응용프로그램 개발자가 쉽게 서비스를 개발할 수 있도록 하고 있다. 두 번째는 신뢰성 있는 멀티미디어 스트리밍 서비스를 제공할 수 있도록 보장하는 것이다. 이를 위해서 본 프레임워크는 MMStream TMO 모델을 사용하여 time-based 미디어의 실시간 처리를 설계 단계서부터 보장할 수 있도록 하였고, 정확한 실시간 처리를 지원하기 위해서 TMO 처리 엔진에 멀티미디어 스트리밍 지원 기능을 추가하였다.

(그림 3)은 프레임워크를 구성하는 각각의 구성요소들을 보여주고 있으며 그 특성과 역할을 다음과 같다.

- Source/Sink SpM : MMStream TMO에서 time-based 미디어 데이터를 처리하는 응용프로그램 모듈로써, 스트림의 생성, 변환, 그리고 소비를 담당하며, MMTMOSL (MMStrema TMO Support Library)를 사용하여 구현할 수 있다.
- MMStream TMO : MMStream TMO는 분산 실시간 멀티미디어 응용프로그램 설계 및 구현을 위해 제공되는 객체 모델인 동시에 time-triggered 미디어 처리 모듈의 연결방식을 제공한다. 연결된 SpM 모듈들은 AAC에 명세 된 시간 축 상에서 미디어 데이터를 처리한다.
- TMOSL/MMTMOSL API : TMOSL(TMO Support Library)는 TMO 기반의 실시간 응용프로그램을 개발할 수 있도록 지원되는 기본 인터페이스다. 그리고 MMTMOSL은 실시간 멀티미디어 스트리밍 서비스를 개발하기 위해서 필요한 API다. 결국, TMOSL 및 MMTMOSL을 TMOSM/Linux가 제공하는 실시간 스트리밍 지원 기능들을 제공받을 수 있다.
- TMOSM/Linux : TMOSM/Linux는 기본적으로 TMO 객체를 처리하는 실시간 처리 엔진으로써[8], MMStream TMO 객체를 지원하기 위해서 미디어 전송, 미디어 동기화, 자원관리, QoS 제어, 그리고 글로벌 시간 동기화 등과 기능들을 포함한다. 실시간 멀티미디어 스트리밍 서비스는 TMOSM/Linux의 지원 아래 적시 서비스가 가능하다.



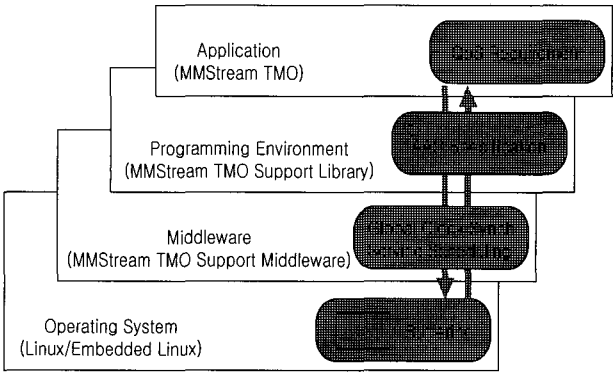

(그림 3) 프레임워크 구성 요소

## 4. Time-triggered 멀티미디어 스트리밍 동기화

### 4.1 동기화 문제점 및 솔루션

실시간 멀티미디어 서비스에서 신뢰성(guarantee)이라 함은 다양한 비동기 장애를 극복하고 요구된 QoS를 만족시킬 수 있도록 하는 시스템 능력을 의미하는데, 이를 위해서는 네트워크에 연결된 종단간(end-to-end) 지연 및 지터[2]를 고려한 스트리밍 동기화가 필수적이다[3]. 지연 및 지터와 함께 서비스 비동기를 유발하는 요소로는 정확하지 않은 시스템 설계 방법 및 동기화 시간 명세를 들 수 있다. 실시간 시스템인 분산 멀티미디어 스트리밍 서비스는 설계 단계에서부터 서비스의 품질인 QoS에 대한 고려가 필요하다. 다시 말하면, 설계 단계에서 QoS에 영향을 미치는 로컬자원, CPU 사용률, 네트워크 대역폭, 그리고 사용자 요구사항과 같은 실시간 특성을 정확히 분석 및 표시함으로써 적시서비스에 대한 틀을 마련해야 한다. 즉, 시스템의 특성 및 요구조건이 시스템 설계에 자세히 나타나 있어야 한다. 특히, 미디어내(intra-media) 동기와 및 미디어간(inter-media) 동기화를 정확히 명세 가능해야 한다.

하지만, 기존에 제안된 많은 동기화 기법들은 LDU (Logical Data Unit)에 대한 처리시간을 고려하지 않고 단순히 미디어들 간의 시간 관계에 대한 명세만 치중해 온 문제점을 가지고 있다. 반면에 본 논문에서 제안한 동기화 솔루션은 GPS로부터 수신된 표준 시간을 축으로 해서 분산 노드에 참여하는 모든 time-based 미디어를 하나의 연속된 흐




(그림 4) TMO 기반의 동기화 처리 모델

름과 시간상에서 표현 및 처리할 수 있도록 할 뿐 아니라, 각각의 LDU에 대한 처리 시간까지 고려하였다.

(그림 4)는 이와 같은 과정을 TMO 기반의 실시간 멀티미디어 스트리밍 지원 프레임워크의 각 부분에 적용한 그림을 보여주고 있다.

(1) 응용프로그램 수준에서는 사용자가 지정한 동기화 시간을 명세하기 위해서 MediaTime를 제공한다. MediaTime는 스트리밍의 시작시간과 정지시간을 간단히 명세할 수 있도록 추상화된 객체이다.

(2) 프로그래밍 인터페이스 환경에서는 응용프로그램에게 정적(static)인 스트리밍 동기화 시간을 명세할 수 있는 방법을 제공한다. 특히, 단순한 스트리밍 시작 및 종료 시간에 만족하지 않고 미디어내 동기화와 미디어간 동기화 시간까지도 명세할 수 있다. 이것은 TMO가 가지고 있는 AAC 명세 방법을 통해서 가능한데, 이러한 조건들은 스트리밍 객체 생성 시 미들웨어에게 전달된다. AAC는 관련연구에서 설명하였듯이 SpM의 시간조건을 명세 하는데 사용된다.

(3) 미들웨어 수준에서는 실시간 멀티미디어 스트리밍 객체들이 가지고 있는 AAC에 따라 정확히 동기화 될 수 있도록 지원한다. 여기에는 글로벌 시간 동기화와 서비스 스케줄링 등의 주요 기능과 부가적인 기능들이 제공된다. 특히, 분산 미들웨어인 TMOSM/Linux는 현재 진행 중인 서비스의 QoS 조건들을 모니터링 하여 동적(dynamic)으로 AAC 조건을 변경함으로써 그룹 동기화를 보장할 수 있도록 지원한다.

(4) 시스템 수준에서는 네트워크 전송에서 발생하는 지연 및 지터를 해결하기 위해서 버퍼링을 수행한다.

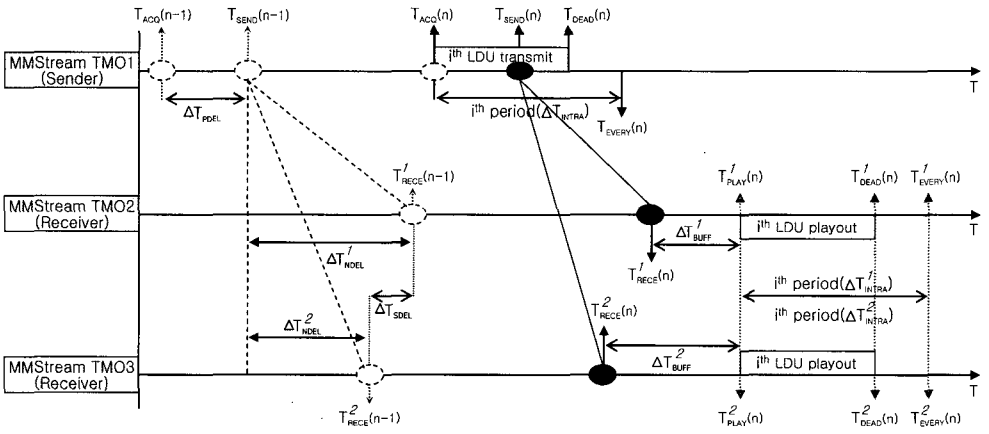
### 4.2 동기화 메커니즘

본 논문에서 제안한 동기화 메커니즘은 GPS에서 수신된 글로벌 시간정보와 TMO 스킴을 토대로 3단계에 걸쳐서 진행된다. 다음은 동기화 메커니즘을 단계별로 설명하고 있다.

- 글로벌 시간 동기화 구축 단계 : GPS로부터 수신된 표준 시간인 글로벌 클록을 사용하여 분산 멀티미디어 스

〈표 1〉 동기화 시간 결정을 위한 표기법

| | |
|---|---|
| $LDU(n)$: n번째 LDU | $T^{i}_{ACQ}(n)$: LDU 획득 시간 |
| $T^{i}_{SEND}(n)$: LDU 전송 시간 | $T^{i}_{RECE}(n)$: LDU 수신 시간 |
| $T^{i}_{PLAY}(n)$: LDU 재생 시간 | $T^{i}_{DEAD}(n)$: LDU 처리 마감 시간 |
| $T^{i}_{EVERY}(n)$: LDU 처리 주기 시간 | $\triangle T_{PDEL}(n)$: LDU의 전송 전 지연시간 |
| $\triangle T_{SDEL}(n)$: LDU의 동기화 지연시간 | $\triangle T^{i}_{NDEL}(n)$: LDU의 전송 지연시간 |
| $\triangle T^{i}_{BUFF}(n)$: LDU의 버퍼 대기시간 | i: 스트리밍 채널 번호, n: LDU 번호 |




(그림 5) MMStream TMO 기반의 LDU 처리과정

트리밍 서비스를 수행하는 각 노드의 시스템 시간을 동기화 하는 단계이다. 이때, 주파수 동기화(frequency synchronization) 를 하지 않았기 때문에 발생하는 클록 스큐(clock skew)는 시간 동기화를 주기적으로 적용함으로써 해결한다. 글로벌 시간을 동기화 하는데 걸리는 지연 시간을 측정한 결과, 평균 지연은 12.57ms, 최대 지연은25.155ms 그리고 최소 지연은 4.007ms이다. 이 시간 값은 멀티미디어 스트리밍 동기화 허용 스큐에 부정적으로 작용할 수 있으므로, 최적의 글로벌 시간 동기화 알고리즘이 필요시 된다.

- 미디어 동기화 시간 결정 단계 : 1단계 결과로 구축된 글로벌 상태에서 time-triggered 멀티미디어 스트리밍을 수행하기 위해서 필요한 동기화 시간을 결정하는 단계다. 본 단계에서의 스트리밍 동기화 시간이란 미디어내 동기화와 미디어간 동기화 시간이며, 결정된 시간은 MMStream TMO의 SpM AAC에 상수 값으로 명세 된다.

우선, 동기화 시간을 결정하기 전에 <표 1>과 같은 기본적인 표기법을 정의하였다.

(그림 5)는 <표 1>에 정의된 표기법을 사용하여 3개의 분산 노드에 위치한 MMStream TMO 객체들이 LDU를 처리하는 과정을 글로벌 시간 축 상에서 보여주고 있다.

이때, MMStream TMO2와 MMStream TMO3의 스트리밍 시간 조건 중 가장 중요한 $T^{i}_{PLAY}(n)$은 아래와 같은 동기화 시간 결정 알고리즘에 의해서 결정된다.

*Algorithm* :

```
DecideSynchTime(T) {
  DecideTripDelay(T) {
    maxDelay = 0;
    for i = 1 to j {
      do { send LDU(n-1) To MMStream_TMO(j);
           △T^i_NDEL(n-1) = T^i_RECE(n-1) - T^i_SEND(n-1);
           if (maxDelay < △T^i_NDEL(n-1))
           then maxDelay = △T^i_NDEL(n-1);
         }
    }
  }
  DecidePlayOut(T) {
    δ = △T_PDEL(n) + maxDelay;
    T^i_PLAY(n) = T^i_ACQ(n) + δ ;
  }
}
```

- 동기화 수행 및 흐름 제어 단계 :본 단계에서는 스트림 동기화와 흐름제어를 수행하는 단계로써, 결정된 스트리밍 시간을 원격 TMOSM/Linux에게 전송 한 후에 로컬 및 원격 MMStream TMO는 time-triggered 멀티미디어 스트리밍을 시작한다. 서버의 흐름 제어는 네트워크 부하에 동적을 대처하고 지속적으로 동기화를 보장하기 위해서 클라이언트가 피드백한 버퍼량 및 패킷 손실 정보를 토대로 스트리밍 시간을 재결정하여 스트리밍을 계속한다.

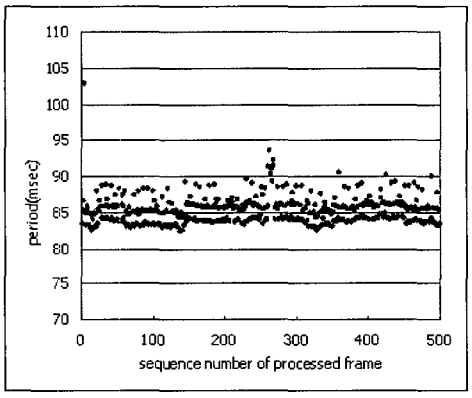
## 5. 구현 및 실험

본 장에서는 time-triggered 멀티미디어 스트리밍을 지원하기 위해서 개발된 프레임워크에서 적시 서비스 및 동기화 능력을 향상시킬 수 있는지 여부를 알아보기 위해서 다음과 같은 비교실험을 하였다.

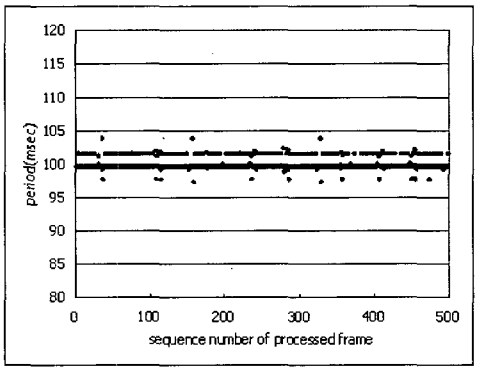
### 5.1 Intra-media 동기화 능력에 대한 실험

본 논문에서 제안한 동기화 기법이 intra-media 동기화 능력을 향상시킬 수 있는지 여부를 알아보기 위해서 다음과 같은 방법으로 실험을 하였다. 우선, 리눅스 상에서 웹 카메라인 webcam-eye을 사용하기 위해 video4linux API 기반의 cpia2 드라이버를 설치한 다음, time-triggered 멀티미디어 스트리밍 지원 프레임워크가 제공하는 MMStream TMO, LiveVideoSrc SpM, LiveVideoSink SpM을 사용하여 비디오 스트림을 재생하는 경우와, 리눅스 오픈 소스인 mview 1.0가 재생하는 경우를 비교 실험하였다.

(그림 6)의 (가)는 MMStream TMO를 사용하지 않았을




(가) mview1.0 기반에서의
비디오 프레임 재생 간격




(나) MMStream TMO 기반에서의
비디오 프레임 재생 간격

(그림 6) Intra-media 동기화 스큐 비교결과

경우 단일 H.261 비디오 스트림의 재생 주기를 보여주고 있으며, 반면에 (나)는 MMStream TMO를 사용하여 구현된 경우를 보여주고 있다. 실험 결과, (그림 6)에서처럼 MMStream TMO를 사용한 경우에는 각 프레임의 재생 주기 편차인 intra-media 동기화 스큐가 상대적으로 작아진 것을 확인할 수 있다. 즉, (가)의 경우 각 비디오 프레임간의 최대 동기화 스큐는 21msec인 반면, (나)의 경우는 6msec 이내인 것을 알 수 있다. 따라서, 본 실험 결과를 토대로 lip 동기화와 inter-media 동기화 능력도 MMStream TMO를 사용하였을 경우에 더 향상될 수 있음을 추정할 수 있다. 그리고 이는 각 비디오 프레임 간 재생 주기 차이가 작고 일정하기 때문에 분산 실시간 멀티미디어 서비스의 신뢰성과 품질을 향상시킬 수 있음을 의미한다.

### 5.2 Group 동기화 능력에 대한 실험

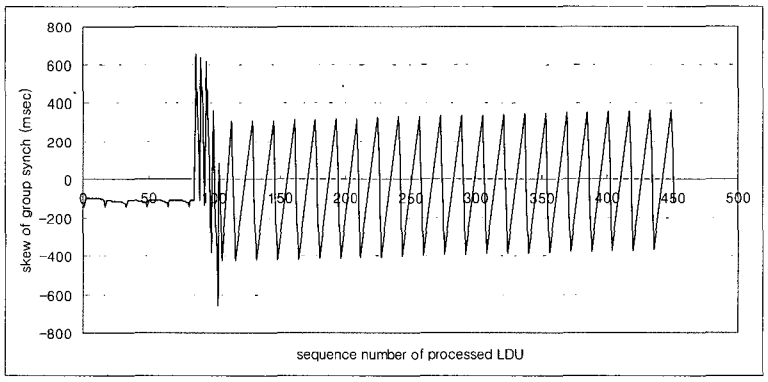
본 실험은 분산 노드간의 서비스 동기화 조건인 group 동기화 보장 여부를 측정하기 위한 실험으로써, 리눅스 스트리밍 서버에 2개의 리눅스 클라이언트를 연결한 후에 LDU 크기가 2kbye인 PCM 오디오를 스트리밍하면서 발생하는 그룹 동기화 스큐를 측정하였다.

(그림 7)은 본 논문에서 개발한 동기화 메커니즘을 적용하지 않았을 경우 client1과 client2 간에 발생하는 group 동기화 스큐 측정결과이다. (그림 8)은 동기화 메커니즘을 적용한 경우에 발생하는 group 동기화 스큐 측정결과이다. 이때, 사용된 SpM의 주기는 600msec이다. 분석 결과, time-triggered 멀티미디어 스트리밍과 실시간 동기화 메커니즘을 적용했을 경우에는 적용하지 않았을 경우보다 client1과 client2간에 발생하는 그룹 동기화 오차인 스큐 값이 현저히 작았고 일정한 패턴을 보였다. 즉, 본 논문에서 제시한 time-triggered 멀티미디어 스트리밍 기술과 실시간 동기화 기술을 적용하지 않은 경우에는 인터럽트에 대한 처리가 원활하고 일정 성능을 유지할 수는 있으나 스큐 값이 일정하지 않고 허용 오차 범위를 크게 초과했다. 물론, 허용 오차 범위에 가깝게 스큐를 줄이기 것이 가능은 하겠지만, MMStream TMO 기반으로 서비스를 개발하는 것이 더욱 효과적이고 쉽게 신뢰성 있는 서비스를 개발할 수 있음을 알 수 있었다.

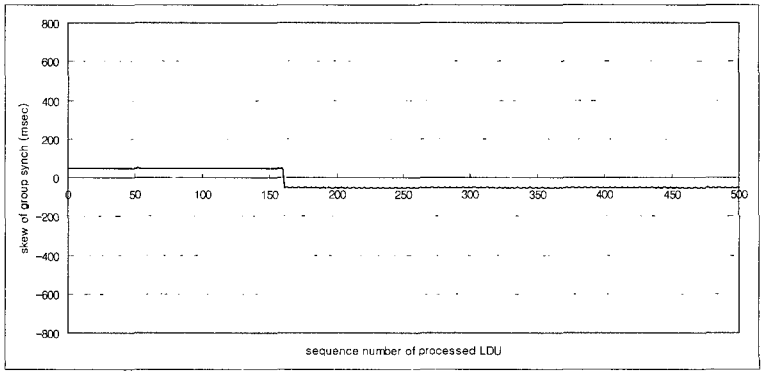
## 6. 결론 및 향후과제

본 논문에서는 신뢰성 있는 멀티미디어 스트리밍 서비스를 효과적으로 개발하기 위한 TMO 기반의 time-triggered 멀티미디어 스트리밍 방식과 실시간 동기화 메커니즘, 그리고 이를 지원하는 소프트웨어 프레임워크를 제시하였다.

Time-triggered 멀티미디어 스트리밍은 TMO 기반의 time-based 미디어 처리 기술로써 주기적 메소드인 SpM을 사용해서 time-based 미디어를 처리하는 실시간 스트리밍 처리 기술이다. 그리고 MMStream TMO는 time-triggered




(그림 7) Group 동기화 스큐 측정결과1




(그림 8) Group 동기화 스큐 측정결과2

멀티미디어 스트리밍 기술을 실현하기 위해서 개발된 실시간 멀티미디어 스트리밍 객체 모델로써, 분산 멀티미디어 스트리밍 서비스 객체 연결과 제어를 가능하게 한다. 특히, MMStream TMO는 SpM의 AAC에 따라 자동으로 스트림 데이터를 처리하는 시간 중심의 미디어 처리 방식이기 때문에 SpM의 AAC 명세를 변경함으로써 스트리밍 흐름을 효과적으로 제어할 수 있고, 서비스 시간 분석이 용이하기 때문에 신뢰성 있는 멀티미디어 스트리밍 서비스 개발이 가능해진다.

Time-triggered 멀티미디어 스트리밍 기술이 적용된 멀티미디어 스트리밍 서비스는 본 논문에서 제안한 소프트웨어 프레임워크를 통해서 효과적으로 개발할 수 있다. 본 프레임워크의 목적은 개발자가 복잡한 응용프로그램 구조에 대한 부담을 덜고 신뢰성 있는 멀티미디어 스트리밍 서비스를 개발할 수 있도록 지원하는 것이다. 이를 위해서 본 프레임워크는 time-based 미디어 처리를 담당하고 있는 SpM을 time-triggered 미디어 처리 모듈로 제공하고 있으며, MMStream TMO가 실시간 처리를 지원받을 수 있도록 미들웨어 형태의 처리 엔진인 TMOSM/Linux를 제공한다. 또한 개발자에게는 C++ 형태의 MMTMOSL API를 제공함으로써, 객체지향 소프트웨어 개발을 할 수 있도록 지원한다.

본 논문에서는 time-triggered 멀티미디어 스트리밍 기술과 함께 서비스의 적시 서비스 능력을 향상시키기 위해서 글로벌 시간 기반의 실시간 멀티미디어 동기화 메커니즘을 개발하였다. 본 동기화 메커니즘은 글로벌 시간 구축 단계, 동기화 시간 결정 단계 그리고 동기화 수행 및 흐름제어 단계로 구성되었으며, 분산 네트워크에서 발생하는 지연 및 지터를 극복하고 미디어내, 미디어간 그리고 그룹 동기화를 보장할 수 있는 메커니즘을 제공한다.

결론적으로 본 논문에서 제안한 time-triggered 멀티미디어 스트리밍 기술과 프레임워크, 그리고 실시간 동기화 메커니즘은 향후 실시간 멀티미디어 스트리밍 서비스 개발에 효과적으로 적용될 수 있음을 동기화 실험을 통해서 검증하였다.

## 참 고 문 헌


[1] Sitaram, D. and Dan, A., "Multimedia Servers: Application, Environments, and Design," Morgan Kaufmann Publishers, San Francisco, 2000.

[2] R. Steinmetz, "Human perception of jitter and media synchronization," IEEE Journal on Selected Areas in Communications, Vol.14, pp.61-72, January 1996.

[3] Y. Xie, C. Liu, M. J. Lee, and T. N. Saadawi, "Adaptive multimedia synchronization in a teleconference system," Journal of Multimedia Systems, Vol.7, pp.326-337, 1999.

[4] Kim, K.H., "Real-Time Object-Oriented Distributed Software Engineering and the TMO Scheme," Int'l Jour. of Software Engineering & Knowledge Engineering, Vol. No.2, pp.251-276, April, 1999.

[5] Kim, K.H., Ishida, M., and Liu, J., "An Efficient Middleware Architecture Supporting Time-Triggered Message-Triggered Objects and an NT-based Implementation," Proc. ISORC'99 (IEEE CS 2nd Int'l Symp. on Object-oriented Real-time distributed Computing), pp.54-63, May, 1999.

[6] Kim, D.H, "An Extended Object Composition Model for Distributed Multimedia Services," Proc. WORDS2002 (IEEE Workshop in Object-oriented Real-time Dependable Systems), San Diego, Jan., 2002.

[7] Kim, D.H, "A TMO-based Software Architecture for Distributed Real-time Multimedia Processing," Proc, WORDS2003 (IEEE Workshop in Object-oriented Real-time Dependable Systems), Guadalajara, Mexico, Jan., 2003.

[8] Kim, H.J., Park, S.H., Kim, J.G., and Kim, M.H., "TMO-Linux: A Linux-based Real-time Operating System Supporting Execution of TMOs," Proc. ISORC '02 (5th IEEE CS Int'l Symp. on Object-Oriented Real-time Distributed Computing), Crystal City, VA, April, 2002.



**조 은 환**

e-mail : joeh@dongbu.com

1997년 건국대학교 전자계산학과(학사)

1999년 건국대학교 대학원 컴퓨터정보통신학과(공학석사)

2005년 건국대학교 대학원 컴퓨터정보통신학과(공학박사)

2003년~2006년 건국대학교 컴퓨터공학부 강의교수

2006년~현재 (주)동부정보기술 미래기술연구소 연구원

관심분야 : 유비쿼터스 컴퓨팅, 실시간 임베디드 시스템, 소프트웨어공학



**김 문 회**

e-mail : mhkim@konkuk.ac.kr

1979년 서울대학교 전기공학과 학사

1981년 서울대학교 석사

1985년 University of South Florida, Computer Science and Engineering, MSCS

1991년 University of California, Berkely, Computer Science Ph.D

1991년~2006년 9월 건국대학교 컴퓨터공학부 교수

관심분야 : 객체지향 분산실시간 시스템, 임베디드 시스템, 결함허용시스템, 소프트웨어공학