이동통신 환경에서의 H324 프로토콜 지원을 위한 H.245 프로토콜의 구현

유제영° 이승익 이동만
한국정보통신대학교 공학부
{jayyu, silee, dlee}@icu.ac.kr

Implementation of H.245 Protocol to Support H.324 over the Mobile Environment

Je-young Yu°, Seungik Lee, Dongman Lee
School of Engineering, Information and Communications University

요 약

H.324는 일반 전화망을 이용한 영상 전화 및 회의용 터미널 시스템 규약이며, H.245는 H.324 기반 터미널 양단이 통신을 시작하거나 끝내기 위해 교환하는 제어 메시지를 처리하는 프로토콜로써, H.324 시스템의 제어 프로토콜의 역할을 담당한다. H.245 프로토콜은 H.324 뿐만 아니라 H.323 등에도 같은 역할을 담당한다. 그래서 멀티미디어 통신에서는 빠질 수 없는 프로토콜 규약으로써 이를 준수하는 프로토콜 모듈들이 다수 개발되어 왔으며 지금도 개발이 진행되고 있다. 그러나, 이를 H.324를 기반으로 하는 멀티미디어 이동 통신 환경에 그대로 적용하는 데 있어서는, 제한된 시스템 자원 등으로 인하여 여러 제약이 따른다. 따라서 기존에 구현된 H.245를 그대로 적용시키는 것은 무리가 있다. 이에 본 연구에서는 ITU-T 영상회의 표준인 H.245를 준수하는 이동 통신 환경의 인터넷 폰 및 휴대전화에서의 멀티미디어 통신에 최적화된 제어 프로토콜 모듈을 설계, 구현하였다.

1. 서 론

H.324[1]는 일반 전화망을 이용한 영상 전화 및 회의용 터미널 시스템 규약이며, H.323[5] 을 기반으로 한 멀티미디어 셋탑 박스의 제한된 이동성을 극복한, 이동 통신 환경을 위해 제안된 프로토콜이다. 그러나 H.324 시스템은 하나의 독립된 단위 개체로써 멀티미디어 통신에 대한 모든 규약 및 방법을 규정하는 것이 아니라 복수개의 프로토콜 스택으로 나뉘어진 하위 레벨의 프로토콜에 의존하여 통신에 필요한 정보를 교환하도록 설계되어 있다.

그 중에서 특히 H.245[2]는 H.324 기반 터미널 양단이 통신을 시작하거나 끝내기 위해 교환하는 제어 메시지를 처리하는 프로토콜로써, H.324 시스템의 제어 프로토콜의 역할을 담당한다. H.324 규약을 준수하는 양단의 통신 터미널은 각 상대의 터미널 상태 및 기능 정보에 대해 알지 못하기 때문에 이러한 제어 메시지를 교환하지 않고서는 통신을 시작 및 끝을 내거나 지속할 수 없게 된다. 이러한 역할로 인해 H.245 프로토콜은 H.324 시스템과 가장 밀접한 관계를 갖는다. H.245 프로토콜은 H.324 뿐만 아니라 H.323 등에도 같은 역할을 담당한다. 그래서 멀티미디어 통신에서는 빠질 수 없는 프로토콜 규약으로써 이를 준수하는 프로토콜 모듈들이 다수 개발되어 왔으며 지금도 개발이 진행되고 있다.[7]

그러나 이를 H.324를 기반으로 하는 멀티미디어 이동 통신 환경에 그대로 적용하기에는 많은 부담이 뒤따른다. 즉, 모든 기능이 포함된 H.245 프로토콜 모듈을, 시스템 자원이 제한된 이동 통신 터미널에 그대로 이용하게 되면, 시스템 자원을 비효율적으로 사용할 수밖에 없다. 게다가 효과적인 자원 이용을 위한 이동 통신 터미널 자체의 특성으로 인하여 대부분의 소프트웨어가 기존의 셋탑 박스 및 PC 플랫폼과는 달리 제한된 플랫폼에 그 사용이 국한된다.

따라서 이동 통신 환경에서 멀티미디어 서비스를 효과적으로 제공하기 위해서는 H.324 시스템의 이동 통신 환경에서의 기능에 최적화된 H.245 프로토콜 모듈의 개발이 필수적이라 할 수 있다. 이에 본 연구에서는 ITU-T 영상회의 표준인 H.245를 준수하는 이동 통신 환경의 인터넷 폰 및 휴대전화에서의 멀티미디어 통신에 최적화된 제어 프로토콜 모듈을 개발하였다.

2. H.245 표준 규약 분석

H.245는 H.324 터미널간에 멀티미디어 데이터를 주고받기 위해 각 터미널의 동작을 제어하거나 통신에 대한 정보를 전달할 수 있는 컨트롤 메시지 교환을 위한 하나의 제어 프로토콜이다.

그림 1에서 보는 바와 같이 H.245는 H.245프로토콜을 사용하는 프로토콜 유저와 인터페이스를 가지고 있으며, 이 인터페이스를 통해서 H.245와 H.324는 서로의 의미 단위 (semantic) 들을 주고 받게 된다. 이 때, 호환성을 위해서 부호화 (Marshalling) 과 복호화 (Unmarshalling) 의 필요성이 대두되는데, 이를 위하여서는 ASN.1 PER (Packed Encoding Rule) [4] 을 사용한다.

H.245는 시스템 컨트롤을 위한 제어 메시지를 가지고 있으며, 이것을 H.324는 자신의 시스템이나 상대편의 시스템을 제어하는데 이용한다. H.245 표준에서 기술하는 여러 기능들은 프로토콜 처리부 (*Protocol Entity*) 에 의해서 수행된다. 프로토콜 처리부는 여러 가지 신호 처리부 (Signaling Entity)로 구성되어 있으며. 이 중에는 필수적인 것들과 그렇지 않은 것들이 있다. 신호처리부는 다음과 같은 부분들로 이루어진다.

1. 주종 관계 결정 (Master/Slave Determination)
2. 능력 교환 (Capability Exchange)
3. 논리 채널 신호 전달 (Logical Channel Signaling)
4. 논리 채널 폐쇄 요청 (Close Logical Channels)

5. H.223[3] 다중 테이블 정보 교환 (H.223 Multiplex Table procedures)

6. 멀티미디어 데이터 형식 요청 (Mode request procedures)

7. 왕복 지연 정보 교환 (Round Trip delay procedures)

이 밖에도 관리 순환 신호 전달 (Maintenance loops), 명령 및 표시 (Commands and Indications) 신호 처리부가 존재한다.

신호처리부에 대한 상세한 내용은 ITU-T 영상회의 표준 문서인 H.245[2]에 기술되어 있다.

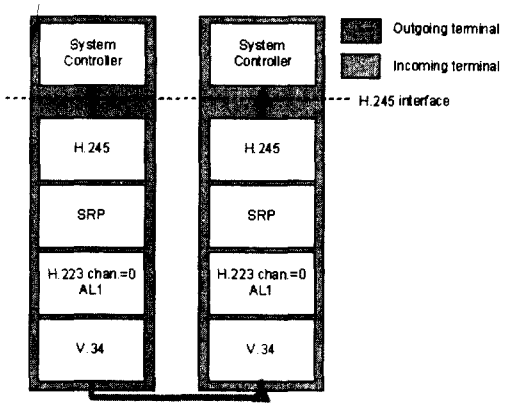

그림 1 H.324 컨트롤 스택에서의 H.245 프로토콜

3. H.245 모듈 설계

이동통신 환경에서의 H.324 프로토콜 지원을 위한 H.245 프로토콜 모듈을 설계 및 구현하는데 있어서 몇가지 문제들을 발견할 수 있다. 이 문제들은 이동통신 터미널의 플랫폼의 제약으로부터 발생하는 것과 구현 상의 제약으로부터 발생하는 것으로 나눌 수 있다.

우선 이동통신 터미널의 플랫폼으로부터 발생하는 제약사항은 OS의 지원을 받지 못함으로 인하여 멀티쓰레드나 멀티프로세스를 사용할 수 없고, 동적 메모리 사용에 제약이 있다는 것과 메모리나 타이머 등의 시스템 자원이 한정되어 있다는 것을 들 수 있다. 구현 상의 제약으로부터 발생하는 것으로는 C++ 기반으로 개발되는 H.245를 사용하는 다른 프로토콜-H.324 프로토콜-에서는 C++가 아닌 C등의 다른 언어를 사용할 수 있어야 한다는 것이다.

H.245 프로토콜 구현시 C++ 언어를 사용하는 이유는 첫째, 앞장에서 기술한 H.245 프로토콜의 특성상 객체 지향적 방법으로 설계 및 구현하는 것이 적당하며, 사용하고자 하는 ASN.1 컴파일러[7] 역시 C++ 형태의 PER 복/부호화 모듈을 생성하기 때문이다.

이동 통신 터미널의 플랫폼으로부터 발생하는 제약사항을 극복하기 위하여 H.245를 이동 통신 환경에 최적화 할 필요성이 있다. 최적화 방법으로는 이동 통신 환경에서 불필요한 기능들의 제거 및 프로토콜 메시지의 단순화, 시스템 자원의 재활용이 있으며, 하드웨어의 의존하는 기능, 예를 들어 타이머 관련 기능 등은 H.245에서 외부 모듈-시스템 컨트롤러-에 요청하여 해결한다. 또한 이동 통신 터미널의 특성상 풍부한 저장장치를 가지고 있지 않으므로, H.245 프로토콜 모듈의 크기를 최소화해야 한다.

구현 상의 제약으로부터 발생하는 문제를 해결하는 방법으로 H.245의 외부 인터페이스를 최대한 단순화시키고, 클래스 형태의 자료 구조뿐만 아니라, C 형태의 자료 구조도 제공하고, 각각의 자료 구조 형태로 자료를 자동으로 변환할 수 있도록 함으로써 H.245 프로토콜의 사용자가 어떤 언어로 구현하더라도 쉽게 H.245 프로토콜 모듈의 기능을 사용할 수 있도록 한다. 또한 H.245 프로토콜을 기능별로 3부분(신호전달 처리부, 메시지 변환 처리부, PER 스트림 복/부호화 처리부)으로 분리하여 각 부분들 사이의 상호의존성을 최소화하여, 각 모듈의 수정이나 변경으로 인한 다른 모듈의 추가 변경을 최소화 시킨다. 특히 PER 스트림 복/부호화 처리부는 ASN.1 컴파일러에 의해 자동 생성되는 부분으로써 추후 더 성능이 좋은 ASN.1 컴파일러를 사용할 경우를 대비하여 교체가 용이하도록 설계한다. 각 부분별 수행 기능들은 다음과 같다.

- 신호 전달 처리부 (Signaling entities): H.324 터미널 간의 통신을 위해 사용되는 해당 메시지들의 교환에 대한 처리 (negotiation) 을 담당한다.

- 메시지 변환 처리부 (PDU handlers): H.324 의 멀티미디어 통신을 제어하기 위한 컨트롤 메시지의 처리를 담당한다. 즉 각 메시지들의 의미 (semantic) 을 정의하여 이를 H.324 프로토콜 (시스템 컨트롤러) 로 전달하거나 반대로 H.324 프로토콜로부터 특정 신호를 전달받아 이를 H.245 컨트롤 메시지로 정의하는 기능을 담당한다.

- PER 스트림 복/부호화 처리부 (PER Stream encoder/decoder): 양단의 H.324 터미널은 이진 데이터 형식인 PER 비트 스트림 (bit stream) 의 데이터를 서로 주고받는데 이러한 PER 비트 스트림으로의 부호화 및 복호화를 처리한다.

H.245 프로토콜은 전체적인 프로토콜 스택에 따라 메시지 전달 경로가 정해져 있지만 본 연구에서는 H.245 프로토콜은 모든 통신을 H.324 시스템 컨트롤러를 거쳐서 이루어지도록 설계하였다. 즉, H.245 프로토콜이 직접 SRP(Simple Retransmission Protocol)[8]와 이진 데이터를 주고 받는게 아니라 타 계층 프로토콜과의 통신을 H.324 시스템 컨트롤러에 위임함으로써 프로토콜 통신 경로가 일원화된다. 이러한 H.245 인터페이스의 일원화는 전체적으로 H.324의 구조를 단순화시키며, H.245 프로토콜이 마치 시스템 컨트롤러의 일부인 것처럼 동작할 수 있게 함으로써 H.324가 H.245를 관리하기 쉽게 만들어 준다.

4. 구현 및 테스트

H.245 프로토콜 모듈의 사용자가 직접적으로 이용하는 클래스가 그림 2에 나타나있는 CH245Protocol 클래스이다.

H.324 시스템 컨트롤러는 SendToH245 멤버함수를 이용하여 서비스를 요청하며, H.245 프로토콜 모듈은 SendToH324SC 멤버함수를 이용하여 H.324 시스템 컨트롤러에게 각종 이벤트나 서비스 요청의 결과를 알려준다. 멤버함수인 ProcessPDU와 ProcessPERStream은 메시지 변환 처리와 PER 복/부호화 처리를 하는 부분으로, 메시지 변환 처리부인 CH324ControlPDU 클래스와 PER 스트림 복/부호화 처리부를 사용하여 작업을 수행한다.

각 신호처리부는 독립적인 클래스로 구현되어 있으며, CH245Protocol 클래스는 각각의 신호처리부를 멤버 변수로 가져 각각의 신호처리부에 대한 요청을 수행한다.

H.245 프로토콜 모듈의 다음과 같이 동작한다. H.245 프로토콜은 양단의 H.324 시스템을 컨트롤하는 프로토콜로써 해당 중추 기능들은 각 성격별로 세분화되어 각각 독립적으로 동작한다. 이들이 바로 신호 전달 처리부 (SE) 이다. 이들은 기본적으로 해당 신호 전달 처리부 개체 (SE instance) 의 상태 (state) 및 내부 매개변수값을 관리하면서 H.324 시스템 컨트롤러의 명령형 데이터 (primitive), 상대측 터미널의 컨트롤 메시지, 그리고 내부 타이머 등에 의해 동작이 시작된다.

CH245Protocol

Member variables

```
CH245_CapabilityExchange m_CE;
CH245_CloseLogicalChannel m_CLC;
CH245_LogicalChannel m_LC;
CH245_MaintenanceLoop m_ML;
CH245_MasterSlaveDetermination m_MSD;
CH245_ModeRequest m_MR;
CH245_MultiplexTable m_MT;
CH245_RequestMultiplexEntry m_RME;
CH245_RoundTripDelay m_RTD;
CH245_Command m_Cmd;
CH245_Indication m_Ind;
```

Member Functions

```
bool SendToH245 (EH245_Primitive_Tag tag,
                 void* msg,
                 int size);
bool SendToH324SC (EH245_Primitive_Tag tag,
                   void* msg,
                   int size);
bool ProcessPDU (H324ControlPDU& pdu);
bool ProcessPERStream (char *ptrByteArray,
                       int size);
```

그림 2. CH245Protocol 클래스

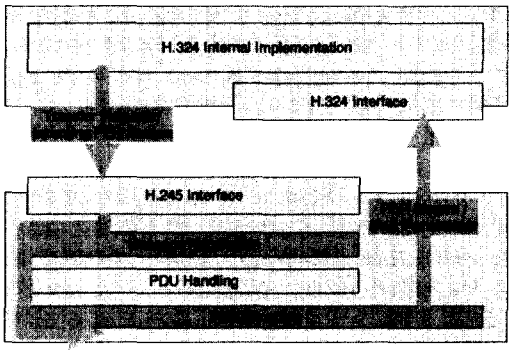

그림 3 터미널에서의 H.324와 H.245와의 상호통신

각각의 신호 전달 처리부의 관리를 일원화하기 위해 하나의 H.245 프로토콜 개체(CH245Protocol 클래스의 개체(instance)) 내에 모든 신호 전달 처리부를 위치시켜 H.324 시스템 컨트롤러는 하나의 H.245 프로토콜 개체와 통신한다.

H.245 프로토콜 개체는 H.324 시스템 컨트롤러로부터 명령형 데이터 (primitive) 를 전달 받는다. 이 명령형 데이터는 고급 언어형 메시지로 이루어져 있으며 H.245 프로토콜의 신호 처리부는 이 데이터를 해석 및 적용하여 알맞은 동작을 취하고, 그 과정 및 결과에 따라 적절한 메시지를 구성 및 부호화 (encoding) 하여 다음 프로토콜 스택인 SRP 에 전달하거나 동작 결과에 따른 알맞은 명령형 데이터를 H.324 시스템 컨트롤러에 전달하게 된다. 반대로 상대방 터미널로부터 메시지를 수신했을 경우, 해당 메시지는 H.223 및 SRP 를 거쳐서 이진 데이터 형식으로 H.245 프로토콜에 전달되며 H.245 프로토콜은 이 이진 데이터 형식을 디코딩 (decoding) 하여 고급 언어형 메시지로 변환한 다음 이를 신호 처리부 (SE) 에서 처리하도록 한다. 그리고 신호 처리부의 처리 결과에 따라 다시 새로운 메시지가 생성되어 상대측 터미널에 송신되거나 H.324 시스템 컨트롤러에 명령형 데이터 가 전달된다.

구현된 H.245 프로토콜 모듈의 테스트는 H.245 프로토콜의 자체적인 모듈 테스트 및 PER 복/부호화 테스트, 그리고 H.245 프로토콜 모듈간의 통신 테스트로 나뉘어서 진행되었다. H.245 프로토콜 모듈간의 통신 테스트는 이 페이퍼가 작성된 시점까지 H.324 시스템 컨트롤러의 구현이 끝나지 않은 관계로 H.245 모듈을 Linux에서 컴파일한 후, 간단한 테스트 프로그램을 작성하여 두 H.245 모듈이 서로 정상적으로 컨트롤 메시지를 주고 받으며, 정상적인 동작을 하는 것을 확인하였다.

5. 결론 및 향후 연구 방향

본 연구의 중점적인 연구 사항은 H.245 프로토콜 표준 규약을 충실히 준수하면서 H.324 터미널의 이동성으로 인해 생기는 제약사항 등에 따라 H.245 프로토콜 모듈을 최적화하는 것이다.

본 연구에서는 H.245 프로토콜 모듈의 최적화를 위해 다음과 같은 3단계의 구조로 모듈을 세분화 했다.

1. 신호 전달 처리부 (Signaling entities)
2. 메시지 변환 처리부 (PDU Handlers)
3. PER 스트림 복/부호화 처리부 (PER stream encoder/decoder)

이는 H.245 프로토콜에서 H.324 시스템 컨트롤러와 통신하는 데 사용되는 메시지, 그리고 H.324 터미널 간에 사용하는 메시지를 올바르게 처리하기 위해서 각 처리부를 계층화한 것이다. 본 연구에서 이동성 환경을 위한 H.324 시스템을 개발하는데 필수적인 요소인 컨트롤 프로토콜, H.245 프로토콜 모듈이 개발되었다. 아직 추가 연구 사항이 산재한 상태이지만 본 연구를 통해 H.324 시스템의 구현 방향이 설정되고 최적화 방안이 논의되어 그 해결책이 제시되었다는 점에서 소기의 목적을 달성했다고 볼 수 있을 것이다.

본 연구의 향후 연구 방향으로는 타 H.245 모듈간 호환성 테스트와 성능 평가 및 최적화 등이 가능하다. 본 연구에서 수행한 H.245 프로토콜 모듈 테스트는 구현된 모듈간 테스트에 국한된 것으로써 타 구현 모듈과의 호환성을 검증하기에는 부족하다. H.245 프로토콜 모듈의 성능에 대해서는 본 연구에서 검증된 바 없다. 다만 H.245 표준 권고안을 충실히 따르는데 그치며 H.324 시스템과의 통신 및 다른 터미널과의 실제 통신 상황에서 어떤 성능을 보여줄 수 있는지에 대한 검증이 부족한 상태이다. 이에 향후 타 H.245 프로토콜과의 호환성 테스트 및, 성능평가, 최적화 등이 필요하다.

5. 참고 문헌

[1] ITU-T Recommendation H.324 (1997), Terminal for low bit-rate multimedia communication.

[2] ITU-T Recommendation H.245 (1997), Control protocol for multimedia communication.

[3] ITU-T Recommendation H.223 (1996), Multiplexing protocol for low bit-rate multimedia communication.

[4] ITU-T Recommendation X.691 (1997), Information technology ASN.1 Encoding Rules Specification of Packed Encoding Rules (PER).

[5] ITU-T Recommendation H.323 (1998), Packet-based multimedia communication systems.

[6] ITU-T Recommendation X.680 (1994), | ISO/IEC 8824-1:1995, Information technology Abstract Syntax Notation One (ASN.1): Specification of basic notation.

[7] Open H.323, http://www.openh323.org/

[8] ITU-T Recommendation H.324 (1997), Terminal for low bit-rate multimedia communication. Annex A.