

論文2003-40TC-12-3

SIP 기반 VoIP 시스템에서 QoS 제어기능 구현

(Implementation of QoS Control Function in SIP based VoIP System)

羅晶煥*, 尹德鎬*, 金永翰*, 金恩淑**, 姜信角**

(Jeonghwan Na, Dukho Youn, Younghan Kim, Eunsook Kim, and Shingak Kang)

요약

본 논문에서는 SIP(Session Initiation Protocol) 기반의 VoIP(Voice over Internet Protocol) 시스템에서 종단간 QoS(Quality of Service)를 보장하기 위한 시스템 구조와 제어 방법을 설계하고 구현하였다. VoIP 시스템의 하부망은 Intserv over Diffserv 구조를 적용하였고 망 차원의 QoS 보장을 위해 수락제어 기법을 이용하여 자원을 관리하도록 하였다. UA(User Agent)는 호 설정을 위하여 QoS 절차가 포함된 SIP를 사용하고 하부망 QoS 구조에 독립적으로 동작할 수 있도록 모듈화 하였다. 구현된 시스템은 실험을 통하여 QoS 제어된 VoIP 성능을 검증하였다.

Abstract

In this paper, we design and implement a QoS control function in the SIP-based VoIP system. As a network infrastructure for VoIP service, we select the Intserv over Diffserv architecture where the network resources are managed by a call admission control mechanism. The SIP protocol extended to support QoS signaling procedure is modularized to operate independently with the infrastructure. The performance of the QoS-enabled VoIP system is verified by experiments.

Keywords :SIP, QoS, VoIP

I. 서 론

VoIP 서비스를 제공하기 위해서는 여러 QoS 요소들이 고려되어야 한다. 본 논문에서는 VoIP 시스템에 적용 가능한 QoS 제어방법을 설계하고 구현하였다. 먼저 QoS 요소들을 하부망과 UA의 두 측면으로 나누어 고려하였다. 하부망에서 고려되어야 하는 QoS 요소로는 망 구조^[1]와 망 자원 관리를 위한 기법 등이 있다. 그리

고 UA 측면에서 고려할 부분은 하부망의 QoS 구조에서 사용되는 자원예약 프로토콜의 종류^[2]와 호 설정을 위해 사용하는 시그널링 프로토콜^[3, 4] 등의 요소가 있다. 이러한 두 가지 측면을 고려하여 시스템을 설계하고 구현하였으며 이를 실험을 통하여 설계된 내용이 타당함을 증명하였다. 특히 본 논문에서는 설계한 UA는 모듈 구조를 취하고 있어 기능 추가 및 수정이 용이하다. 이 중에서 자원예약 절차를 담당하도록 설계한 모듈은 여러 하부망 QoS 구조를 인식하여 그에 맞는 자원예약 절차를 지원하도록 설계하였다. 이는 UA가 하부망 QoS 구조에 독립적으로 동작하도록 하여 확장성이 용이하다.

* 正會員, 崇實大學校 情報通信電子工學部
(School of Electronics Engineering, Soongsil University)
** 正會員, 韓國電子通信研究院 通信프로토콜標準研究팀
(Electronics and Telecommunications Research Institute)
接受日字:2003年11月8日, 수정완료일:2003年12月3日

VoIP 시스템에서는 하부망 구조가 중요한 비중을 차지 한다. 본 논문에서는 IETF에서 제안된 QoS 하부망 모델 중 Intserv over Diffserv 모델을 채택하였다^[1]. 이는 Intserv의 확장성 문제와 Diffserv에서 종단간 QoS를 제공하지 못하는 문제를 해결하였다. 그리고 망의 자원을 효율적으로 관리하기 위한 기법으로 파라미터 기반의 수락제어를 사용하였다^[5]. 설계한 VoIP 시스템에서 설정에 사용하는 시그널링 프로토콜로서 SIP에 자원 예약 절차를 지원할 수 있도록 확장된 방식^[3]을 사용하였다.

UA는 기능 추가 및 확장을 위하여 기능별로 모듈화하여 설계하였다. 설계한 모듈들은 계층적인 구조를 갖으며 각기 독립적으로 실행된다. 각 모듈은 사전에 정의된 API 함수를 통해 통신하며 내부 동작에 대해서는 서로 영향을 받지 않는다. 따라서 사전에 정의된 절차만 지키게 구성하면 해당 기능이 동작한다. 따라서 각 모듈의 기능 추가나 수정이 용이하게 되는 것이다. 특히 자원 예약을 담당하는 기능을 별도의 모듈에서 관리 하도록 구현하여 UA에서 하부망에 의존적이던 부분을 제거하였다. 즉, 하부망이 Intserv 망이면 RSVP를 사용하여 자원 예약을 하고 Diffserv 망이면 DSCP 값을 설정하게 되는 과정 등을 UA는 관여하지 않아도 된다. UA는 단지 자원 예약 담당 모듈에 API 함수를 통해 QoS 요청을 하며 자원 예약 성공 여부에 대한 결과만 받아서 처리한다. 따라서 UA 입장에서는 하부망 QoS 구조까지 고려할 필요가 없다. 이는 VoIP 시스템에서 UA 부분의 요구사항이 줄어든다는 측면에서 큰 이득이 있다.

서론에 이어서 II장에서는 VoIP 시스템의 QoS 요소에 대해서 살펴보고 III장에서는 이를 구현한 것에 대하여 설명하며 IV장에서는 구현된 시스템의 동작 결과를 보이고 V장에서 결론을 맺는다.

II. VoIP 시스템 QoS 요소 설계

1. 하부망의 QoS 기반구조

Intserv over Diffserv 모델은 종단간 QoS를 실현하기 위해서 Intserv와 Diffserv의 단점을 보완하여 제안되었으며 종합서비스의 문제점이었던 확장성 문제를 차등화 서비스 망을 이용하여 해결하였다. 또한 차등화 서비스에서 지원하지 못했던 end-to-end QoS나 수락제어를 종합서비스의 RSVP를 사용하여 가능하게 한 모델이다. 또 다른 장점으로는 자원기반 수락제어, 정책기반 수락

제어, 트래픽의 분류 및 조절 등이 가능하다^[1-2, 6].

두 망의 연동을 위해서 [6]에서 소개된 서비스 맵핑 중 기본 맵핑을 사용하였다. 기본 맵핑은 Intserv의 GS(Guaranteed Service) 트래픽을 Diffserv의 EF(Expedited Forwarding) 트래픽으로 CL(Controlled-load Service)을 AF(Assured Forwarding)로 맵핑하는 것이다.

또한 종단간 QoS를 구현하기 위해서 Diffserv 구간의 지역조건을 만족 시킬 수 있는 통계적인 방법을 적용하여 파라미터 기반의 수락제어에 이용하였다^[7].

2. 파라미터 기반 수락제어

Intserv over Diffserv 망은 여러 종류의 수락제어를 가능하게 한다. 수락제어는 보편적으로 이용되는 QoS 보장기법이다. 본 논문에서는 이러한 파라미터 기반의 수락제어를 수행하도록 구성하였다. 종단간 QoS 보장을 위해서 Diffserv 구간에서 허용 가능한 한계치를 수식 (1)의 계산식을 이용하여 라우터가 허용할 수 있는 호의 개수를 정하게 된다. 허용 가능한 자원이 남아있는 경우에만 수락하고 그렇지 않은 경우에는 해당 요청을 거부하게 되어 기존의 QoS 서비스를 받고 있는 플로우에 대해서 품질을 보장한다.

수식 (1)은 WFQ 스케줄러를 사용하는 Diffserv 구간의 지역특성을 모델링 한 것이다^[7]. 이 수식을 통해 종단간 요구되는 지역 값을 만족시키기 위하여 수락제어 시 호를 허용할 수준을 결정하게 된다.

$$\hat{D}_{\text{req}} \leq \hat{D}_{N^* M/G/1} + \sum_{i=1}^N \left(\frac{M}{R_i} + \frac{MTU_i}{C_{link}} \right) \quad (1)$$

여기서 R_i 는 i 번째 노드에 EF 클래스에 할당된 대역폭이며, M 은 EF로 계약한 VoIP 트래픽의 최대 패킷의 크기이다. C_{link} 는 i 번째 노드에 대한 출력 링크 대역폭이 되며, MTU는 패킷의 최대 크기가 된다. $\hat{D}_{N^* M/G/1}$ 은 N 차 $M/G/1$ 큐에서 통계적으로 유효한 종단간의 지역 값이며 수식 (2)와 같은 값을 갖는다.

$$\hat{D}_{N^* M/G/1} = \mu_N + \alpha(P) \cdot \sigma_N \quad (2)$$

수식 (1)과 수식 (2)를 [7]의 방법에 따라 고정된 지역 조건 하에서 VoIP 클래스에 예약된 대역폭과 해당 큐의 부하 관계의 수식으로 전개 가능하다. <그림 1>은 Diffserv 구간의 총 흡의 개수를 3개로 가정하고($N=3$), 모든 흡에서의 지역특성은 동일하며 ($R=R_1=R_2$

$= R_3$). MTU = 500bytes, VoIP 패킷크기를 100bytes로 설정하였고 C_{link} 는 10Mbps로 고정하였을 경우 VoIP 클래스에 할당된 WFQ의 가중치와 최대부하의 관계를 보여준다. Diffserv 구간의 지연 조건을 5msec에서 25msec 까지 5msec 단위로 설정하였다.

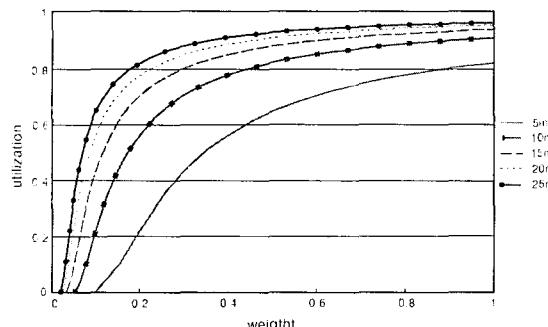


그림 1. 지연 및 부하곡선

Fig. 1. Maximum load as a function of weight(N=3).

즉 <그림 1>에서 종단간 QoS 보장을 위해 Diffserv 구간에서 10msec의 지연조건을 만족시키고자 할 때 VoIP 클래스가 전체 링크의 30%를 차지하는 경우 서비스를 보장을 위해서는 VoIP 클래스의 70%까지만 호를 허용해야 한다.

3. UA(User Agent) 구조 설계

일반적인 UA의 기능은 호를 설정하고 음성 데이터를 송수신하며 사용하는 코덱에 따라 인코딩과 디코딩을 수행한다. 본 논문에서는 SIP 확장^[3]을 사용하는 UA를 설계한다. 설계한 UA는 기능에 따라 모듈화 하였고 효율적인 관리를 위하여 계층적인 구조를 갖는다. 각 모듈들은 서로 독립적으로 동작하며 상호간에 정의된 API

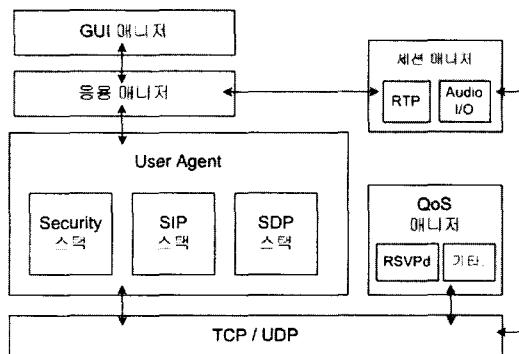


그림 2. UA 모듈 구성도

Fig. 2. UA structure.

함수를 사용하여 필요한 기능을 수행한다.

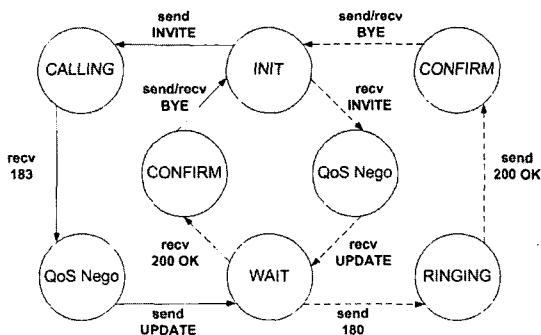
<그림 2>에서는 UA의 모듈의 위치와 각 모듈들이 호출하는 API 함수의 관계를 보여준다. 주요 모듈로서 GUI 매니저, 응용 매니저, 세션 매니저, QoS 매니저가 있고 핵심 UA 모듈이 있다. 각 모듈이 하는 역할을 다음과 같다.

- GUI 매니저 : 사용자 인터페이스 역할을 수행한다. 사용자가 설정한 내용을 저장하고 콜에 대한 명령을 응용 매니저에게 전달하며 UA 상태를 화면에 출력하는 기능을 수행한다.
- 응용 매니저 : GUI 매니저의 명령을 해석하며 이에 따라 UA를 이용하여 SIP 메시지를 전송하고, SIP 메시지를 받았을 경우 사용자의 확인이 필요한 경우 GUI에 표시하는 기능을 수행한다. 또한 세션이 연결될 때 SIP 시그널링의 결과를 이용하여 미디어 전송을 시작할 수 있도록 세션 관리자의 API 함수를 호출한다.
- 세션 매니저 : 응용 매니저를 통해 호가 설정되면 음성 통화를 시작하게 된다.
- UA : 호 상태에 따른 SIP/SDP 스택 및 SIP 절차적 상태를 관리한다.
- QoS 매니저 : UA의 자원예약 요청을 해석하여 하부망 QoS 구조에 따라 자원예약을 수행한다. 자원예약 상태를 관리하며 수행된 결과를 UA에게 알려주게 된다. 실질적으로 자원예약 절차에 관한 모든 과정을 담당하여 UA가 하부망에 독립적으로 QoS 서비스를 받을 수 있게 된다.

호 설정에 사용되는 SIP 확장^[3]은 기본 SIP 메소드에 QoS를 위한 단계를 추가하여 통화 개시 이전에 자원예약이 이루어지도록 확장된 것이다. 통화에 참여하는 UA들 간에 QoS에 조건들이 만족될 경우에만 호 설정이 완료되도록 설계 되었다. QoS 조건은 SDP 메시지의 'a' 파라미터에 담겨서 교환되며, UPDATE 메소드를 사용하여 조건이 만족되었음을 상대편 UA에게 알려주어 호 설정을 완료하게 된다.

[3]의 SIP 확장을 적용하는 UA를 구현하기 위해서는 내부적인 상태를 관리하여야 한다. SIP 기본적인 상태와 더불어 자원예약 상태에 대한 것도 고려하여서 새롭게 정의한 상태를 <그림 3>에서 보여준다.

<그림 3>에서 실선은 송신단의 상태이며 점선으로



연결된 부분은 수신단으로 동작할 때의 상태 다이어그램을 나타낸다. UA는 초기 INIT 상태로부터 시작한다. 호를 연결하고자 하는 송신단은 INVITE를 전송한 뒤 CALLING 상태로 가며 수신단에서 183 메시지를 수신한 뒤 QoS Nego 상태로 이동한다. 또한 INVITE를 수신한 수신단은 바로 QoS Nego 상태로 가게 된다. QoS Nego 상태에 있다는 의미는 자원예약을 처리하는 상태에 있다는 것을 의미한다. 자원예약이 성공적으로 이루어지면 UPDATE 메시지를 주고받는데 이때 송신단과 수신단은 WAIT 상태로 가게 되며 그 이후로는 일반적인 SIP 상태와 같게 된다.

(1) GUI 매니저 설계

GUI 매니저는 사용자의 요청을 해석하고 통화 요청과 종료 요청, 호의 진행 상태에 대한 정보를 알려준다. 이외에 사용자는 GUI를 통하여 선택할 수 있는 부가적인 SIP 기능은 다음과 같다. 통화요청은 사용자가 상대방의 SIP URL을 입력하고 호 요청 버튼을 눌렀을 때 응용 매니저에게 이를 알려 통화를 요청하는 기능이고, 등록 기능은 SIP Registrar 서버에 등록하기 위한 메뉴를 제공하는 것이고, 능력조회 기능은 통화하고자 하는 시스템의 능력 정보를 알아오기 위해서 사용하는 기능이다. 메뉴 기능은 서버주소 설정이나, 벨 사운드 선택, 코덱 정보, 통화에 요구되는 QoS 정보 등을 화면에 제공하는 것이다. 사용자가 설정 가능한 QoS 옵션은 [3]에 따라 SDP 메시지 안에 포함되는 내용에 부합하여 QoS 강도, 방향성 등을 선택할 수 있도록 설계하여 SIP 시그널링시 별도의 해석이 필요하지 않도록 설계하였다. 또한 GUI 매니저는 응용 매니저와 API 함수를 통하여 적절한 조치를 취하도록 한다. <표 1>에서는 응용 매니저와 통신에 사용하는 API 함수의 종류를 보여준다.

표 1. GUI 매니저와 응용 매니저간 API 함수
Table 1. APIs between GUI manager and App manager.

API 함수명	설명
EstablishConnection	사용자의 접속 요청을 응용 매니저로 전달
AcceptConnection	상대측 호연결 요청에 대해 사용자가 응답할 때 사용
RejectConnection	상대측 호연결 요청에 대해 사용자가 거부할 때 사용
ReleaseConnection	통화 종료시 호출
Registration	SIP 서버에 등록시 호출
GetCapability	상대 시스템의 서비스 능력 범위를 알아낼 때 호출

(2) 응용 매니저 설계

응용 관리자는 UA가 구동될 때 사용자 설정 파일로부터 각종 초기화에 필요한 동작을 수행하며, GUI 매니저로부터 호출된 API 함수를 처리한다. 응용 매니저는 UA와 세션 관리자, GUI 관리자와의 통신 기능이 있다. 응용 매니저는 전체적인 호의 상태를 관리한다. 사용자 설정과 관련된 기능은 GUI 매니저와 통신하고, 미디어 전송과 관련하여서는 세션 관리자와 통신하며, 호의 진행 상태와 관련하여서는 UA와 통신하는 기능을 갖추고 있다. 각 모듈과 정의된 API 함수를 사용하여 UA의 상태에 따라 가장 적절한 API 함수가 호출 될 수 있도록 하며 다른 모듈에서 호출된 API 처리가 가능하도록 설계하였다. 응용 관리자는 UA의 상태를 보고 현재의 상

표 2. 응용 매니저와 UA 간 API 함수

Table 2. APIs between App manager and UA.

API 함수명	함수의 기능	호출방향
CallSetupReq	INVITE 전송	App ↓ UA
CallSetupRsp	요청에 대한 응답 메시지 전송	
CallReleaseReq	통화 종료시 호출	
CallCancelReq	취소	
RegisterReq	REGISTER 전송	
CapabilitiesReq	OPTIONS 전송	
InformationInd	INVITE에 대한 200OK를 제외한 응답 수신시 UPDATE에 대한 200OK 수신	App ↑ UA
CallSetupInd	INVITE 수신	
CallSetupCnf	INVITE에 대한 200OK 수신	
CallReleaseInd	BYE 수신	
CallCancelInd	CANCEL 수신	
RegisterCnf	REGISTER에 대한 응답 수신	
CapabilitiesCnf	OPTIONS에 대한 응답 수신	

태가 어떤지에 따라서 사용자의 요구를 처리한다. 응용 매니저의 주요 기능은 호의 상태 관리이다. <표 2>에서 는 호와 관련하여 응용 매니저와 UA 간에 호출되는 API 함수를 보여준다.

(3) QoS 매니저 설계

QoS 매니저는 자원예약 절차를 담당한다. UA의 자원 예약 요청에 따라서 하부망에 적절한 자원예약을 시도 하며 수행 결과를 UA에 전달한다. 이는 UA에서 하부망에 의존적이던 부분을 분리한 것으로서 UA가 하부망 QoS 구조에 독립적으로 동작하도록 한다. 기존 QoS 기능이 포함된 UA에서는 UA가 직접 QoS 시그널링 및 자원 상태에 관여하여 관리를 해야만 했다. 이는 UA의 구조가 복잡해지고 같은 기능의 UA라고 하더라도 하부망에 QoS 종류에 따라서 내부 기능들이 수정되어야 함을 의미한다. 본 논문에서 제시하는 QoS 매니저는 이러한 부분을 분리시켜 UA는 [3]의 절차에 따라서 자원예약이 시작되는 시점과 자원예약의 성공 여부의 결과만을 처리도록 하며 QoS 매니저에서 하부망 QoS 구조가 Intserv이면 RSVP를 사용하여 자원을 예약하게 되며 Diffserv이면 알맞은 DSCP 코드값을 마킹하고 이외의 QoS 구조이면 그에 맞는 동작을 추가하여 QoS 기능을 담당하도록 설계하였다. 따라서 UA는 QoS 매니저를 사용함으로서 하부망에 독립적으로 동작할 수 있다. 이때 사용하는 API 함수는 <표 3>과 같이 QoS 매니저를 초기화 하는 것과 자원예약의 요청을 하는 것과 결과를 알려주는 것이 있다. 먼저 초기화 함수는 QoS 매니저를 동작시킨다. QoS 매니저는 호 설정 이전부터 동작하고 있어야 하기 때문에 초기화 부분이 필요하다. 따라서 쓰레드에 대한 처리와 자원예약에 필요한 정보들을 초기화 하는 루틴으로 구성된다. 그리고 UA가 실질적으로 자원예약이 필요한 경우 호출하는 함수와 결과를 알려줄 때 사용하는 함수를 정의하였고 자원예약의 시작과 완료 시점에서 호출되도록 설계하였다.

UA는 자신이 원하는 QoS 조건의 결과를 통보받아

표 3. UA-QoS 매니저간 API 함수
Table 3. APIs between UA and QoS manager.

API 함수명	함수의 기능
init_qosmanager	QoS 매니저 초기화
QoSReq	QoS 절차 요청시 호출
QoSResult	QoS 결과 알림을 알림줌

호 설정을 계속 진행할지 중단할지에 대한 결정을 한다.

QoS 매니저는 추가 기능의 확장과 기존의 UA에서도 사용이 가능한 뛰어난 이식성을 가지고 있다. 이는 라이브러리 형태로 제공되기 때문에 라이브러리 업데이트를 통해서 새로운 QoS 기능의 추가와 확장이 가능하다. 또한 UA가 [3]의 절차에 따라서 3개의 초기화, 자원예약 요청 및 결과 처리의 API 함수만을 추가시키는 과정을 통하여 비교적 간단하게 QoS 기능이 추가될 수 있다. 따라서 기존 UA의 재사용이 가능하며 UA 응용프로그램 개발시 많은 이득이 될 수 있다.

III. SIP 기반 VoIP 시스템 구현

<그림 4>는 Intserv over Diffserv 구조의 VoIP 시스템을 나타낸다. Ingress/Egress 라우터는 Intserv의 에지 라우터와 Diffserv의 보더 라우터의 기능을 동시에 수행하며 망 차원의 수락제어를 수행한다. 코어 라우터는 단순한 Diffserv 라우터의 기능만 수행한다.

UA1은 QoS 서비스를 요청하는 송신단으로 수신단인 UA2로 연결을 희망한다. QoS 보장 여부를 확인하기 위해서 배경 트래픽을 발생시키는 노드를 이용하여 서비스 보장을 확인하도록 구성하였다.

<그림 5>에서는 Intserv over Diffserv 망의 Ingress

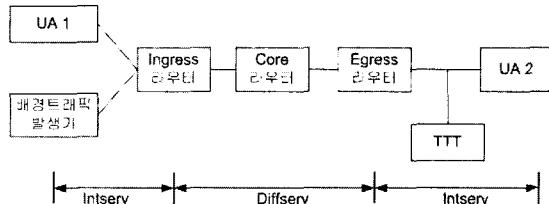


그림 4. VoIP 시스템 구현모델

Fig. 4. A VoIP system model.

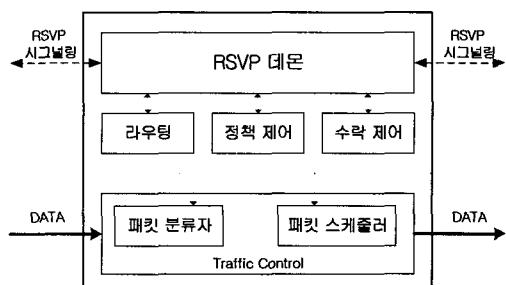


그림 5. Ingress/Egress 라우터 구조

Fig. 5. The egde router structure.

라우터와 Egress 라우터의 내부 구조를 보여준다. 양 보더 라우터에서는 서비스 맵핑이 이루어지며 이를 위해 RSVP 대본과 Diffserv 트래픽 제어 블록과 연동을 통하여 수락제어, 정책제어 등과 같은 기능을 수행한다.

호 연결 절차는 다음과 같다. UA1은 SIP 확장의 시나리오에 따라 호 연결을 시작하여 QoS 설정에 따라 자원예약 절차를 수행한다. Ingress 라우터에서는 Intserv 망으로부터 들어온 GS 트래픽을 EF 트래픽으로 맵핑하게 되고 이때 파라미터 기반의 수락제어가 이루어진다. 이는 Diffserv 구간에 자원이 남아있는 경우에만 EF로 표시하여 패킷을 전달하는 방식을 취하게 된다. 만일 망에 자원이 충분치 않을 때는 BE 트래픽으로 처리하여 해당 트래픽이 포워딩 되도록 설정하였다.

1. UA

UA는 [3][4][8]에 따라 SIP/SDP 스택을 포함하며 앞

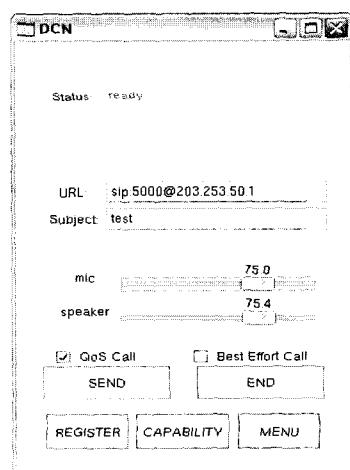


그림 6. 통화설정 화면

Fig. 6. User interface for call setup.

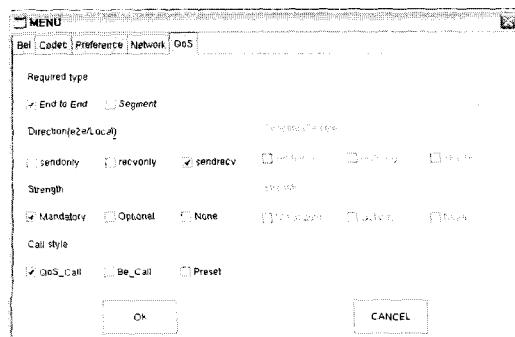


그림 7. QoS 설정화면

Fig. 7. User interface for QoS configuration.

장에서 설계한 GUI 매니저, 응용 매니저, QoS 매니저와 각 모듈간 정의한 API 함수들의 사용이 가능하도록 구현되었다.

<그림 6>과 <그림 7>은 UA 동작시 GUI 매니저가 사용자에게 보여주는 화면이다. 사용자는 자신이 연결하고자 하는 주소와 함께 통화시 희망하는 QoS 설정을 입력하게 된다.

UA는 SIP 절차에 따라 상대편 UA와 시그널링 메시지를 교환한다. UA에서 수행하는 SIP 확장의 플로우와 QoS 매니저가 수행하는 자원예약 절차를 <그림 8>에서 보여준다^[2]. A가 B로 통화를 연결하고 싶다면 INVITE를 전송한다. B는 이에 대하여 183응답을 하는 것과 동시에 QoSReq API 함수를 호출한다.

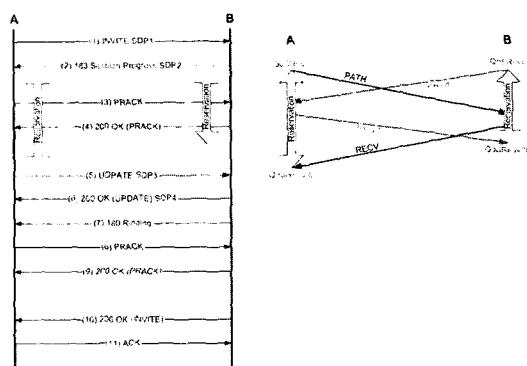


그림 8. UA 시그널링 및 QoS 매니저 시그널링 절차
Fig. 8. UA and QoS manager signaling.

QoS 매니저는 자원예약 요청에 따라 하부망에 알맞는 자원예약 절차를 수행하게 된다. UA가 접속하고 있는 하부망이 Intserv 구조이므로 QoS 매니저는 RSVP를 이용하여 자원을 예약한다. 자원이 성공적으로 예약되었으면 QoSResult API 함수를 호출하여 UA에게 결과를 알려주게 된다. QoS 매니저는 자원예약이 필요한 시점과 끝난 시점에서만 UA와 API 함수로 통신할 뿐 자원예약에 있어서는 UA와 독립적으로 동작한다.

IV. 실험 결과 및 고찰

<그림 9>와 <그림 10>에서는 본 논문에서 구현한 VoIP 시스템의 동작결과를 보여주고 있다. 차등화 서비스 망의 전체 대역폭은 10Mbps 이고 EF 클래스에 대해서 3Mbps가 할당되었으며 총 3개의 노드로 구성되어 설정하였다. 이때 차등화 서비스 망에서 만족 시켜야

하는 지역조건은 10msec로 가정 하였고 <그림 1>의 결과를 이용하여 EF 클래스 대역폭의 70%를 허용 한계치로 두어 수락제어가 이루어지도록 구성하였다. 즉 차동화 망에서는 2.1Mbps에 해당하는 호 까지 QoS 서비스를 받을 수 있도록 수락제어가 일어난다. 이를 초과하는 VoIP 트래픽들은 최선형 트래픽으로 처리된다.

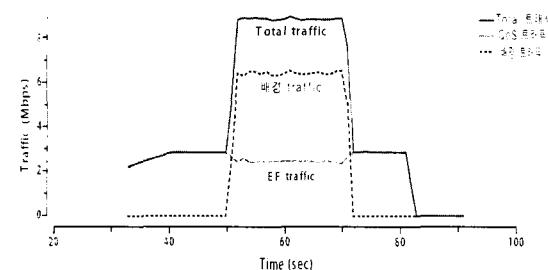


그림 9. VoIP 시스템 동작실험
Fig. 9. QoS performance of VoIP system.

실험에서는 트래픽 발생기인 MGGEN을 사용하여 UA1에서 G.711을 모델링한 QoS 트래픽을 1초 간격으로 30개 까지 발생시키고 50초에 10Mbps의 배경 트래픽이 발생되도록 구성하였다. 각 패킷은 CBR(Constant Bit Rate)의 패턴을 갖고 발생된다. 최선형 트래픽을 발생시키기 전 50초 까지는 망의 부하가 없으므로 30개의 음성 트래픽 전부가 손실 없이 수신단까지 도착하게 된다. 하지만 2.1Mbps 범위의 22개의 호까지만 자원이 예약되고 이를 초과하는 나머지 8개의 트래픽은 최선형 서비스를 받게 된다. 이는 배경 트래픽이 망으로 유입되는 시점이후에 확인할 수 있다. 2.1Mbps 이상으로 도착된 8개의 플로우는 서비스 품질을 보장받지 못하고 최선형 서비스와 동등하게 처리된다. 하지만 앞선 22개의 플로우는 자원이 예약된 상태에서 통신을 하므로 자신들이 원하는 서비스 품질을 받게 된다. 실험결과 최선형 서비스와 QoS 보장을 받지 못하는 트래픽들은 36.7%의 패킷 손실률을 보여주었으며 QoS 서비스를 받는 22개의 플로우는 0%의 손실률을 보여주었다. 이는 패킷에 포함된 번호를 통하여 확인 하였으며 이에 대한 결과는 <그림 9>를 통하여 확인 가능하다. 이론상 최선형 서비스의 손실률은 27.5%이나 실제 링크의 대역폭이 10Mbps 보다 적기 때문에 더 많은 36.7%의 손실률이 측정된 것이다. 또한 [7]에 따라서 Diffserv 구간의 QoS 플로우의 손실률은 최대 1% 정도까지 나타나게 되는데 패킷 손실이 전혀 일어나지 않은 것은 최악의 상황을 가정한

상황에서 최대 1% 까지 패킷 손실을 허용 한 것이므로 실험에서 그러한 극한의 환경이 발생하지 않았기 때문이다.

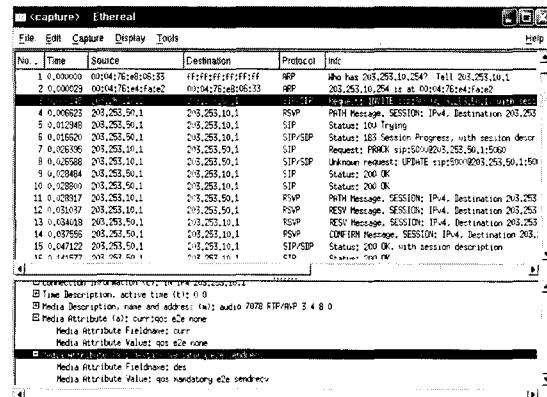


그림 10. UA간 시그널링 메시지 교환
Fig. 10. A snapshot of inter-UA signaling messages.

또 다른 실험은 UA의 동작을 확인하기 위해서 실시되었다. 구현한 UA가 설계한 내용에 따라서 SIP 확장^[3] 시나리오에 따라 호 설정이 이루어지는 것을 확인한다. 이를 패킷캡쳐 프로그램인 Ethereal을 사용하여 호 설정 시 교환되는 메시지를 확인하였다. 이를 <그림 10>에서 보여준다. UA1 (203.253.10.1)이 UA2 (203.253.50.1)에게 INVITE를 전송하여 통화를 요청하고 UA2는 이에 대하여 183 응답과 자원예약을 동시에 시작한다. 자원예약 메시지의 시작과 SIP 메시지의 전송상태에 따라서 정의한 API 함수가 적절하게 동작함을 확인할 수 있었으며 UA의 기타의 동작역시 설계한 내용대로 수행됨을 확인하였다. 최종 실험에서는 망의 부하가 있는 상태에서 음질을 측정하였다. 배경트래픽이 전혀 없는 상태와 15Mbps로 과도하게 발생시킨 상태에서 음질을 측정하였다. 배경 트래픽이 과도한 상태에서도 망의 부하가 없는 경우와 동등한 음질을 제공함을 확인하였다. 이때의 QoS 플로우의 패킷 손실률은 앞서 실험한 결과 마찬가지로 0%로 측정되었다.

실험에서 구현한 UA는 리눅스용으로 공개 배포중인 Linphone의 소스코드를 수정하여 사용하였다. 여기에 QoS 매니저 라이브러리와 API 함수를 추가하여 기존 응용 프로그램의 구조변경 없이 비교적 간단히 QoS 기능을 추가하였다. 이는 기존의 QoS 기법들이 전개하는데 있어서 문제가 되었던 확장성과 이식성 문제를 해결 할 수 있는 하나의 참조 모델이 될 수 있다.

V. 결 론

본 논문에서는 SIP를 사용하는 VoIP 시스템과 QoS의 제어 구조를 설계하고 구현하였다. QoS 하부망 구조로서는 IETF의 Intserv over Diffserv를 적용하였고 이 두 망간 연동부분에서 파라미터 기반의 수락제어를 수행하였다. 요청하는 대역폭보다 남아있는 자원이 부족하면 그 수락을 거절하는 방법으로 QoS 제어기능을 구현하였고 UA에서는 SIP 확장을 사용하여 자원예약이 성공하지 못하면 그 설정이 진행되지 않도록 하여 QoS를 제어하였다. 또한 SIP 확장과 QoS가 기존 UA에서 적용되는데 보다 유연하게 적용될 수 있도록 QoS 매니저의 개념을 도입하여 UA가 세부적인 자원예약 절차에 관여하지 않고 단지 QoS 매니저에게 자원예약 요청을 하고 그에 따른 결과를 수신하는 간단한 과정만 수행하도록 설계하였고 이를 실험을 통하여 타당성을 입증하였다. 본 논문에서는 현재까지 제안된 QoS 모델을 바탕으로 향후 실현될 VoIP 시스템 구조에서 QoS 기능이 보다 확장성 있고 유연하게 적용될 수 있도록 QoS 매니저를 통하여 해결 할 수 있음을 실험을 통하여 증명하였다.

참 고 문 헌

- [1] Y. Bernet, et al., "A Framework for Integrated Services Operation over Diffserv Networks", RFC 2998, Nov. 2000.
- [2] R. Braden, Ed, et al., "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification," RFC 2205, Sep. 1997.
- [3] G. Camarillo, et al., "Integration of Resource Management and SIP" RFC 3312, Internet Engineering Task Force, Oct. 2002.
- [4] J. Rosenberg, et al., "SIP: Session initiation protocol," RFC 2543, Feb. 2002.
- [5] E. Knightly and N. Shroff, "Admission control for statistical QoS: Theory and practice," IEEE Network, vol. 13, no. 2, pp. 20-29, Mar. 1999
- [6] Y. Bernet, et al., "A Framework for Integrated Services Operation over Diffserv Networks," RFC 2998, Nov. 2000.
- [7] Maarten Böhli, et al, "Resource Allocation and Management in DiffServ Networks for IP Telephony", NOSSDAV 2001, 33-39, June 2001.
- [8] M. Handley and V. Jacobson, "SDP: session description protocol," RFC 2327, Apr. 1998.

저 자 소 개



羅晶煥(正會員)

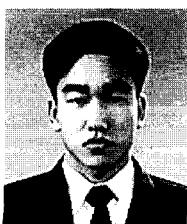
2002년 2월 : 숭실대학교 정보통신 전자공학부(학사). 2002년 3월~현재 : 숭실대학교 정보통신공학과(석사과정). <주관심분야 : 인터넷 QoS 기술, 홈 네트워크, IPv6 등.>



金永翰(正會員)

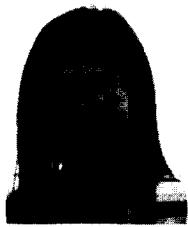
1984년 2월 : 서울대학교 전자공학과 졸업(공학사). 1986년 2월 : 한국과학기술원 전기 및 전자공학과 졸업(공학석사). 1990년 8월 : 한국과학기술원 전기 및 전자공학과 졸업(공학박사). 1987년 1월~1994년 8

월 : 디지콤정보통신연구소 데이터통신연구부장. 1994년 9월~현재 : 숭실대학교 정보통신전자공학부 부교수, 통신학회 인터넷연구회 위원장, VoIP포럼 차세대기술분과 위원장. <주관심분야 : 컴퓨터네트워크, 인터넷 네트워킹, 이동 데이터 통신망 등.>



尹德鎬(正會員)

2002년 2월 : 한국외국어대학교 정보통신과 졸업(학사). 2002년 3월~현재 : 숭실대학교 정보통신공학과(석사과정). <주관심분야 : 차세대 인터넷 기술, 인터넷 QoS 기술, SIP 등.>



金 恩 淑(正會員)

1996년 2월 : 숙명여대 전산학과(학사). 1998년 2월 : 숙명여대 전산학과(석사). 2001년 8월 : 숙명여대 컴퓨터과학과(박사). 2001년 3월 ~ 2001년 7월 : 한국전자통신연구원 표준연구센터 인턴연구원. 2001년 7월 ~ 현재 : 한국전자통신연구원 표준연구센터 선임연구원. <주관심분야 : 신뢰성을 보장하는 멀티캐스트 (Reliable Multicast Transport), 인터넷 멀티미디어 QoS 솔루션, VoIP 응용을 위한 종단간 QoS 시그널링, 이기종 컨퍼런스 간의 컨퍼런스 제어 메커니즘 등.>



姜 信 角(正會員)

1984년 2월 : 충남대 전자공학과(학사). 1998년 2월 : 충남대 전자공학과(박사). 1995년 12월 : 정보통신기술사. 1984년 ~ 현재 : 한국전자통신연구원 통신프로토콜표준연구팀 팀장/책임연구원. 1997년 ~ 현재 : ITU-T SG17 Q.8(End-to-end QoS Multicast) Rapporteur. 2000년 ~ 현재 : 인터넷텔레포니포럼 부의장/운영위원장. 2003년 ~ 현재 : ANF VoIP WG 의장. <주관심분야 : VoIP, 멀티미디어통신, 인터넷정보보호 등.>