임베디드 시스템을 위한 VoIP 모듈 구현

김정원*

Implementation of VoIP Module for Embedded System

Jeong-Won Kim*

요 약

본 논문에서는 임베디드 시스템을 위한 VoIP 모듈을 설계하고 구현한다. 기존 데스크탑용 VoIP(Voice of Internet Protocol) 모듈의 구현을 활발하지만 임베디드 시스템을 위한 VoIP 구현은 많지 않다. 구현된 VoIP 모듈은 OpenH.323 표준을 이용하여 설계되었으며 WinCE 가 탑재된 PDA상에서 구동된다. 구현된 시스템은 혼잡 제어 기법으로 TCP-friendly adaptation 알고리즘을 사용한다. 실험결과 약간의 잡음이 있지만 음성 통신은 원활히 이루어졌으며 향후 연구과제는 잡음의 최소화이다.

Abstract

This paper designs and implements a VoIP module for embedded system. Many other VoIP modules for desktop environment has widely implemented but the one for embedded system is not prevalent yet. The implemented VoIP module uses the OpenH.323 and is operated on the WinCE PDA. We use the TCP-friendly adaptation for congestion control. The experiment results show smooth communication but a little noise. Future works include the minimization of noise.

키워드

VoIP, Embedded system, H.323

I. 서 론

H.323을 이용한 전화시스템이 폭넓게 개발되면서 사용자의 요구가 더욱 증대되고, 그로 인한 서비스의 품질 보장이 필수적인 요소가 되었다. 인터넷 전화는 저렴하고 다른 서비스와의 통합 측면에서 장점이 있으나 통화 품질이 떨어지는 문제가 있다. 또한 기존의 인터넷 사용에서부터 전화서비스와 멀티미디어 서비스와 같은 실시간 서비스까지 모든 사용자의 요구를 수용하기 위해서는 제한된 자원의 효율적인 관리가 필요하다[1,2,3].

H.323 시스템의 구성요소 중 게이트키퍼(Gatekeeper)는 가장 중요한 요소로 터미널의 등록과 인증, 대역폭 관리를 주 기능으로 하며 중앙 집중적인 관리가 가능하게 하는 장치이다. 이러한 게이트키퍼의 대역폭 관리나 인증 제어에 적절한 QoS 보장 스케줄링을 적용함으로서 제한된 대역폭으로 보다 많은 사용자의 요구를 서비스할 수 있으며, 대역폭의 낭비를 막을 수 있다. 따라서 게이트키퍼에 QoS 보장 등을 하기위해서는 게이트키퍼 대한 원천 기술(소스)를 확보하는 것이 중요한다. 게이트키퍼의 개발 도구 구입에만 약 10만$ 정도가 필요하므로 게이트키퍼에 대한 기술 확보는 VoIP 사업과 VoIP를 이용한 부가 사업에서 필수적

*신라대학교 컴퓨터정보공학부

접수일자 : 2004. 8. 5

으로 확보해야 하는 기술이다[4,5,6].

최초 H.323의 버전1이 출시된 이래 현재까지 보안기능에서 출발해 서비스 제어기능, 프로토콜 채택방식, 서비스 품질(QoS) 수준을 높인 버전5까지 발표돼 전 세계적으로 대부분의 VoIP서비스업체나 장비 업체들이 기반기술로 채택하고 있다. 그러나 H.323 표준은 기본적으로 LAN환경에서 멀티미디어 통신을 지원하기 위해 개발된 기술 방식으로 광대역 네트워크를 지원하고 대규모 사용자를 지원하는 데는 기본적으로 한계점을 노출하고 있다. 이러한 구조적인 문제점을 해결하기 위한 대안으로 제시된 표준안이 바로 SIP다.

기존 데스크 탑상에서 VoIP 모듈의 구현은 활발하고 실제 상용화가 되고 있으나 임베디드 시스템상에서 VoIP 은 구현은 아직 상용화가 미미한 수준이다. 따라서, 본 연구에서는 WinCE를 운영체제로 탑재하고 있는 내장형 시스템상에서 VoIP 모듈을 구현하고 실험한다. 2장에서는 VoIP 모듈 설계 및 구현 과정을 소개하고 3장에서는 구현된 혼잡제어 기법에 대해 설명한다. 그리고, 4장에서는 PDA 상에서 구현한 VoIP 모듈의 성능 측정 결과를 소개하고 4장에서는 결론을 맺는다.

II. VoIP 모듈 설계 및 구현

2.1 개발 환경

내장형 WinCE에서 VoIP 모듈은 OpenH.323 공개 소프트웨어를 이용하여 개발한다. 아래의 그림 1 은 OpenH.323 모듈로 개발할 프로토타입 시스템의 구조이다. 타겟보드에서는 개발할 VoIP 모듈의 테스트 버전을 개발한 후 실제 구동은 윈 CE가 탑재된 PDA에서 이루어진다. 리눅스 타겟 보드에서의 개발은 타겟 보드의 마이크와 사운드 인터페이스에 문제가 있어 포켓 PC 가 탑재된 PDA 환경에서의 개발이 용이할 것으로 판단된다. 구현된 환경은 Peer-to-Peer 형태로 게이트웨이 없이 직접적으로 상대방 PDA로 연결될 예정이고 향후 게이트웨이로 연결할 예정이다.

표 1. PDA에서 VoIP 개발 환경
Table. 1 VoIP development parameters

항목	설명
개발언어	Embedded Visual C++
GUI	Windows Embedded
Library	OpenH.323
OS	포켓 PC 2002

기존 데스크탑의 인텔 펜티엄 CPU가 아닌 StringARM CPU 오브젝트 코드를 생산해야하므로 크로스개발 환경에서 OpenH.323 모듈을 컴파일해야 한다. [표 1]은 개발환경을 요약한 것이다.

2.2 시스템 모듈 구조

프로토타입 시스템의 주요 모듈들은 호를 설정 및 제어하는 호 제어(Call Controller) 모듈, 음성과 영상 데이터를 수집하는 그래버(Grabber) 모듈, 음성을 출력하는 재생(Displayer) 모듈, 음성 데이터를 압축하고 해제하는 코덱(codec) 모듈, 압축된 데이터를 인터넷으로 전송하고 수신을 담당하는 RTP 모듈, 그리고 네트웍의 상태에 따라서 전송률을 조절하는 TFBA(TCP-friendly Bandwidth Adaptation) 모듈 등이 있다. 사용자가 전화 걸기 요청을 하였을 때 호 제어기(call controller) 모듈은 상대편과 연결을 설정한다. 연결이 완료됐을 때 오디오는 샘플링이 시작된다. 오디오 데이터는 G.723.1로 압축된 후 RTP 프로토콜로 네트웍을 통하여 전송되어진다. 수신자는 전송된 데이터에 대한 패킷 손실률을 포함하는 피드백 정보를 송신자에게 보내고 TFBA는 이 정보를 이용하여 전송률을 조절한다.

다음은 구현된 시스템의 각 모듈에 대한 설명이다.

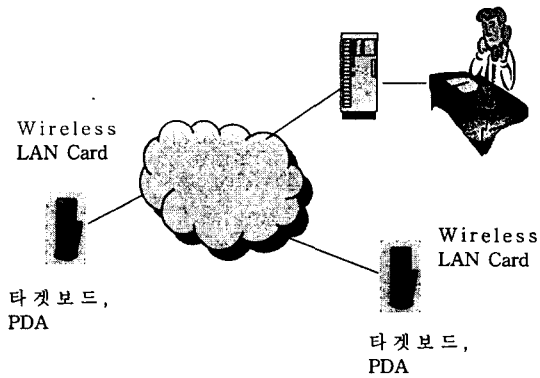

그림 1. 구현 환경
Fig. 1 Implemetation environment

•호 제어 모듈(Call controller)

H.323 기반의 인터넷 전화에서 호(call)의 연결 및 관리를 수행하기 위해 H.225와 H.245프로토콜을 사용한다. 사용자가 전화 걸기 요청을 하게 되면 H.225는 전화 걸기를 수행하고 연결을 설정한다. 이러한 작업들이 모두 완료된 후에 실제로 데이터를 주고받을 논리적 채널을 생성한다.

• 그래버(Grabber)와 재생(Displayer) 모듈

Grabber는 전송할 오디오 데이터를 캡쳐하는 모듈이다. 오디오 데이터의 경우 마이크로부터 샘플링된다. Windows 시스템에서는 오디오 데이터 처리를 위해서 WAVE 형식의 데이터를 처리하기 위한 API를 제공한다.

• 코덱(Codec) 모듈

오디오 데이터를 압축하기 위해서 PCM, GSM, G.711, G.723.1 등의 코덱을 구현하였다. H.723.1은 Windows 시스템의 ACM(Audio Compression Manager) 함수들을 사용한다. 디코딩 모듈은 버퍼의 데이터를 가져와서 디코딩하는 쓰레드와 비디오 데이터를 재생하는 쓰레드로 구성된다.

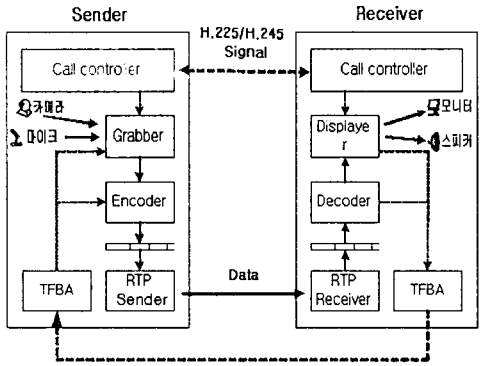

그림 2. 프로토타입 시스템의 구조
Fig. 2 Structure of prototype system

2.3 실시간 음성 전송 시스템의 구현

그림 2 는 프로토타입 시스템의 주요 모듈 구조를 나타낸다. 모듈은 호를 설정 및 제어하는 호 제어(Call Controller) 모듈, 음성 데이터를 수집하는 그래버(Grabber) 모듈, 음성을 출력하는 재생(Displayer) 모듈, 음성 데이터를 압축하고 해제하는 코덱(codec) 모듈, 압축된 데이터를 인터넷으로 전송하고 수신을 담당하는 RTP 모듈, 그리고 네트워의 상태에 따라서 전송률을 조절하는 TFBA (TCP-friendly Bandwidth Adaptation) 모듈 등이 있다[7,8]. 사용자가 전화 걸기 요청을 하였을 때 호 제어기(call controller) 모듈은 상대편과 연결을 설정한다. 연결이 완료됐을 때 오디오는 샘플링이 시작된다. 오디오 데이터는 G.723.1로 압축되고 RTP 프로토콜로 네트워을 통하여 전송되어진다. 수신자는 전송된 데이터에 대한 패킷 손실률을 포함하는 피드백 정보를 송신자에게 보내고 TFBA는 이 정보를 이용하여 전송률을 조절한다[9,10].

음성 데이터의 전송을 위해서 가장 압축률이 높은 G.723.1 송수신 모듈을 구현하였다. 기본적으로 사운드카드로부터 PCM 데이타를 샘플링하고 WinCE에서 제공하는 ACM(Audio Compression Manager)를 사용하여 G.723.1의 압축 모듈을 구현하였다. 그림 3 은 오디오 송수신 시스템의 구조를 보여준다. 그림에서 볼 수 있듯이 사운드 장치는 아날로그 신호를 디지털 신호로 변환해 주며 애플리케이션은 Windows API를 사용하여 사운드 장치로부터 WAVE 형식의 데이터를 얻을 수 있다.

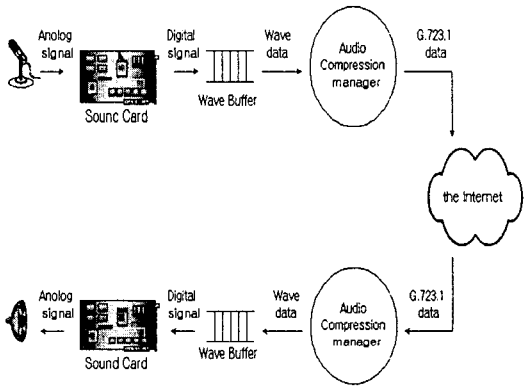

그림 3. G.723.1 송수신 시스템의 구조
Fig. 3 Structure of G.723.1

III. TCP-friendly 흐름제어 기법의 구현

VoIP에서 TCP는 네트웍에 혼잡이 일어날 경우 스스로 대역폭을 줄이는 방법을 사용하는데, 이것이 UDP 패킷과 같이 혼잡 제어를 하지 않는 데이터 플로우를 만나게 되면, TCP의 대역폭만 계속 줄어드는 현상이 일어나게 된다. 이러한 불공평한 상황을 극복하기 위한 혼잡 제어 알고리즘이 TCP-friendly bandwidth adaptation 기법으로, UDP 데이터 전송에 TCP의 혼잡 제어 방식과 같은 알고리즘을 적용하여 전송률을 조절한다. 본 논문에서 구현한 VoIP 시스템에서는 TCP-friendly 알고리즘을 구현하여 전송률을 적응적으로 조절한다.

TCP-friendly adaption과 관련하여 이 알고리즘을 제안 또는 적용한 기존의 연구들을 몇 가지 살펴보면 다음과 같다. [11]은 멀티미디어 응용 프로그램의 전송율을 네트웍상태에 따라 적응적으로 조절하기 위한 메커니즘을 제시한다. [12] 역시 현재 사용 가능한 네트웍 대역폭에 따라서 응용 프로

그램의 비디오 출력 대역폭을 조절하는 전송률 조절(rate control) 메커니즘을 제안한다. 이 알고리즘은 네트웍이 혼잡될 때 비디오 대역폭을 multiplicative하게 줄여주고, 혼잡 상태가 아닐 때에는 additive하게 대역폭을 늘려주는 방식을 사용한다. 그러나 이 논문에서는 얼마만큼의 대역폭을 조절해 줄 것인지에 대해서는 언급하지 않고 있다. [13]은 네트워의 패킷 손실 정도에 따라 전송율을 조절해 준다. 이 논문에서는 조절 파라메터로 RTCP의 피드백 정보로부터 계산할 수 있는 bottleneck bandwidth를 이용한다. [14]는 손실율과 round trip time 그리고 timeout 값을 이용하여 계산한 throughput으로 전송률을 결정하는 TCP-friendly adaptation 알고리즘을 제안한다.

TCP-friendly adaptation은 TCP의 혼잡 avoidance 알고리즘과 유사한 방식을 혼잡 control을 하지 않는 멀티미디어 응용에 사용하여, 네트웍 혼잡 시에 TCP 데이터에 불공평하지 않은 방법으로 해결할 수 있도록 하는 알고리즘이다. 즉, 네트웍 상태에 따라 멀티미디어 응용의 전송률을 TCP-friendly 방식으로 조절하여 네트웍 혼잡을 피하는 방법이라고 할 수 있다.

TCP-friendly adaptation 알고리즘을 사용하여 전송률을 조절하는 메커니즘은 다음과 같다. 먼저 응용 프로그램은 항상 RTCP의 피드백 정보를 이용하여 네트웍 상태를 관찰한다. 그리고 네트웍 상태에 따라 계속 전송률을 조절해 주게 된다. 네트웍 대역폭이 충분할 경우에는 전송률을 증가시키고(additive), 네트웍에서 혼잡이 발견되면 전송률을 감소시켜(Multiplicative) 주는 방법을 사용하게 되는데, 이러한 메커니즘을 AIMD (Additive Increase Multiplicative Decrease)라고 한다. 네트웍 대역폭을 늘려줄 경우에는 혼잡이 일어나는 것을 피하기 위해 전송률을 늘려주고, 대역폭을 줄여야 하는 혼잡의 상태에는 빠른 시간 내에 네트웍을 정상화시키기 위해 대역폭을 감소시켜 주는 것이다. 위와 같은 방법에 따르면 TCP-friendly adaptation 알고리즘은 그림 4와 같이 나타낼 수 있다.

여기서 r_i는 현재 응용 프로그램, 즉, VoIP 시스템의 출력 대역폭을 의미하고, B는 현재 네트웍 상태를 나타내는 파라미터로 본 논문에서는 네트웍의 대역폭 값을 사용한다. 그리고 INC와 GAIN은 각각 대역폭을 조정해주기 위한 Additive increase 값과 Multiplicative decrease 값을 나타낸다. INC값은 average-packet-size / RTT 으로 계산하고, GAIN은 TCP의 AIMD에 따라 0.5로 정의한다. RTT는 round-trip time을 의미하는데, RTCP 피드백 정보를 이용하여 구할 수 있다.

$$\text{if } (r_i \leq B)$$
$$r_i = r_{i-1} + INC;$$
$$else$$
$$r_i = r_{i-1}/GAIN;$$

그림 4. AIMD 알고리즘
Fig. 4 AIMD algorithm

TCP-friendly adaptation 알고리즘을 적용하기 위해 가장 먼저 고려해야 할 것은 어떻게 네트웍의 혼잡 상태를 결정할 것인가 하는 것이다. 앞에서도 언급한 바와 같이 네트웍의 사용 가능한 대역폭 값을 계산하여 그 값을 현재의 출력 대역폭과 비교한다. (식1)은 네트웍 대역폭(B)을 계산하는 식이다.

$$B = \frac{1.3 * MTU}{RTT * \sqrt{loss}} \quad \text{(식 1)}$$

(식1)에서 MTU는 최대 패킷 크기를 나타내고 2048 bytes 값을 사용한다. 그리고 RTT는 round trip time, 그리고 loss는 loss rate를 나타낸다. 이를 통해 계산된 대역폭 값을 응용 프로그램의 출력 대역폭과 비교했을 때 출력 대역폭이 네트웍의 대역폭 보다 적을 경우에는 non-혼잡 상태로 보고 전송률을 늘려주고, 그 반대의 경우에는 혼잡 상태로 보고 전송률을 줄여주게 된다.

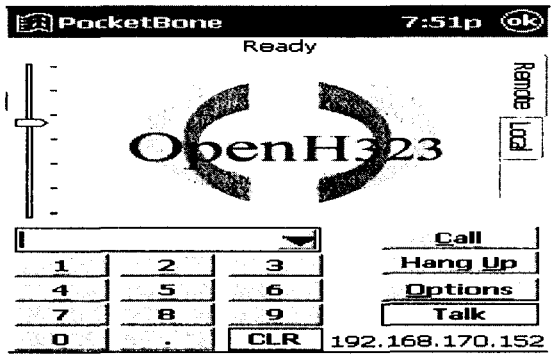

그림 5. 포켓 본의 인터페이스
Fig. 5 Interface of Pocketbone

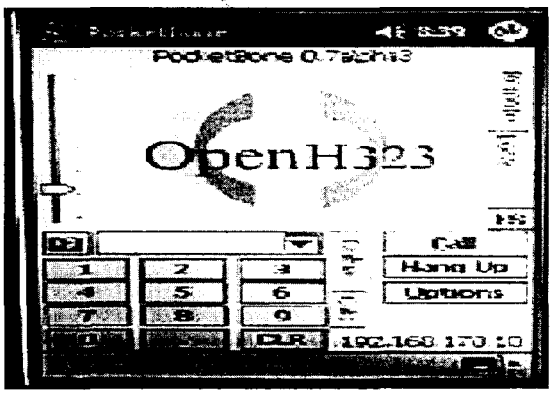

그림 6. 실행 장면
Fig. 6 Running image

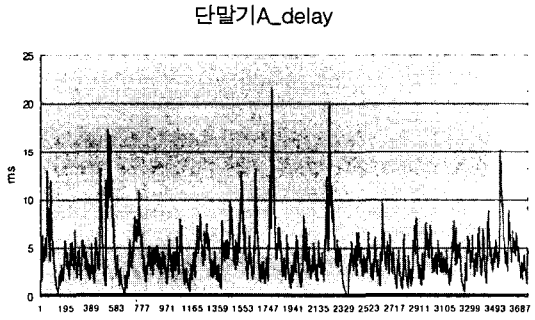

그림 7. Delay 측정
Fig. 7 Delay

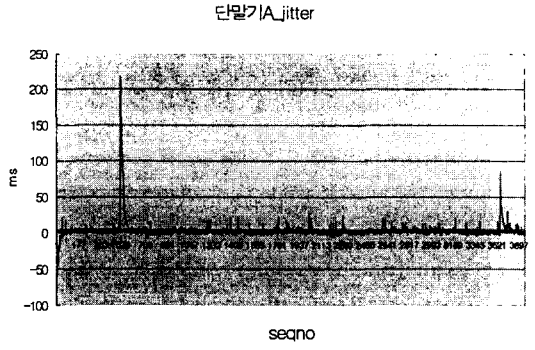

그림 8. 지터 측정
Fig. 8 Jitter

IV. 실험결과

그림 5 는 본 연구에서 구현한 VoIP 모듈의 실행 화면이다. 그림 5 는 준비상태의 인터페이스로서 주소입력 창에서는 상대측의 IP 번호를 입력할 수 있도록 텍스트 박스가 있고 입력을 하고나서는 상대측과 연결하기 위해서는 call 버튼을 누르면 된다. 그림 6 은 상대측 IP 번호를 입력하고 난 뒤 콜이 설정된 후의 상태를 보여주고 있다. 콜이 연결되면 인터페이스 하단에 상대측의 IP 번호가 디스플레이된다. 제공되는 코텍의 종류에는 G.7111-uLaw-64k, G.711-ALaw-64k, GSM-06.10, MS-GSM, LPC-10 등이 있다.

측정은 한 개의 무선 라우터(Access point)를 통과하였을 때 두 단말기 간의 네트웍 지연과 지터(Jitter)를 측정하였다. 그림 7, 8은 측정 결과를 보여준다. 그림 7은 단말기 사이의 지연 시간을 측정한 것이다. x축은 패킷의 시퀀스 번호이고 y축은 지연시간이다. 대부분의 패킷들은 평균 5ms 이내의 지연시간을 보이고 있고 간혹 보이는 버스트는 프로세스 스케줄링이나 인코딩 및 디코딩의 불규칙한 오버헤드에 기인한 것으로 보인다. 그림 8은 지터를 계산한 것인데 대부분 지연시간의 변이(variation)가 0에 수렴하는 것으로 보이고 있는데 이것은 음성 통신이 정상적으로 수행될 수 있음을 나타낸다.

V. 결론 및 향후 연구 방향

본 논문에서는 임베디드 시스템을 위한 VoIP 모듈을 설계하고 구현하였다. PDA 등과 같은 내장형 시스템의 경우에 탑재되는 VoIP 모듈로는 OpenH.323 프로젝트에서 제공하는 라이브러리를 이용하여 내장형 시스템에 맞게 재구성하거나 데스탑에서 Netmeeting SDK를 이용하여 VoIP 모듈을 구현할 수 있듯이 내장형 윈도우 CE상 에서도 OpenH.323 모듈을 이용하여 VoIP 애플리케이션을 구축할 수 있었다. 또한 음성 전송을 위해서 G.273.1 송수신 모듈과 영상 전송을 위해서 H.263 송수신 모듈을 구현하였다. 그리고 다른 TCP 연결에 악영향을 미치지 않고 네트웍 상태에 따라서 전송률을 조절할 수 있는 TCP-friendly 흐름 제어 모듈을 구현하였다. 잡음의 원인은 PDA 의 하드웨어 성능이 음성 통신의 요구사항을 만족하지 못하는 것으로 보이며 향후 연구로는 이 잡음의 제거에 있다.

참고문헌

[1] J-C. Bolot, A.V Garcia, "Control mechanishms for packet audio in the Internet", Proc. IEEE INFOCOM'96, April 1996.

[2] C. Padhye, K.J. Chrstensen and W. Moreno, "A new adaptive FEC loss control algorithm for voice over IP applications", IEEE IPCCC'00, 2000.

[3] Dorgham Sisalem, "End-to-end quality of service control using adaptive applications", IFIP, 1997.

[4] J-C. Bolot, T. Turletti, "Experience with control mechanisms for packet video in the internet", INRIA, France, 1998.

[5] Dorghan Sisalem, Henning Schulzrinne, "The loss-dely based adjustment algorithm: A TCP-friendly adaptation scheme", GMD-Fokus, Berlin, 1998.

[6] Jitendra Padhye, Jim Kurose, Don Towsley, Rajeev Koodli, A Model Based TCP-Friendly Rate Control Protocol, Dept. of Computer Science, UMass, 1999.

[7] Dorgham Sisalem, Adam Wolisz, Towards TCP-Friendly Adaptive Multimedia Applications Based on RTP , IEEE Symposium on Computers and Communications, 1998.

[8] Thierry Turletti, Christian Huitema, Videoconferencing on the Internet, IEEE/ACM Transactions on Networking, Vol. 4, No. 3, June 1996.

[9] JongWon Kim, Young-Gook Kim, TCP-Friendly Internet Video Streaming Employing Variable Frame-Rate Encoding and Interpolation , IEEE, 2000.

[10] Jamshid Mahdavi, Sally Floyd, TCP-Friendly Unicast Rate-Based Flow Control , Pittsburgh Supercomputing Center (PSC), Carnegie Mellon, January 1997.

[11] Dorgham Sisalem, "End-to-end quality of service control using adaptive applications", IFIP, 1997.

[12] J-C. Bolot, T. Turletti, "Experience with control mechanisms for packet video in the internet", INRIA, France, 1998.

[13] Dorghan Sisalem, Henning Schulzrinne, "The loss-dely based adjustment algorithm: A TCP-friendly adaptation scheme", GMD-Fokus, Berlin, 1998.

[14] Jitendra Padhye, Jim Kurose, Don Towsley, Rajeev Koodli, A Model Based TCP-Friendly Rate Control Protocol , Dept. of Computer Science, UMass, 1999.

저자소개

김정원(Jeong-Won Kim)

1995년 부산대학교 전자계산학과(학사)
1997년 부산대학교 대학원 전자계산학과(석사)
2000년 부산대학교 대학원 전자계산학과(박사)
2000년~2001년 기술신용보증기금 기술평가역(차장)
2002년~현재 신라대학교 컴퓨터정보공학부 전임강사
※관심분야 : 내장형시스템, 멀티미디어, 운영체제