

차별화 서비스 망을 위한 정책 기반 서비스 품질 관리 모델

차 시 호[†]·강 영 만^{††}·조 국 현^{†††}

요 약

차별화 서비스(DiffServ)는 효율적이며 규모성을 갖는 방법으로 서비스 품질(QoS)을 제공하는 기술이다. 그러나, 현재의 차별화 서비스 규격들은 완전한 QoS 관리 프레임워크와 구현 모델의 제공에는 취약한 실정이다. 본 논문은 차별화 서비스 망의 QoS를 관리하기 위하여 차별화 서비스 정책을 지원하는 정책 기반 차별화 서비스 QoS 관리 모델을 제안한다. 이 모델은 모델-뷰-컨트롤러(MVC) 구조를 따르며, EJB(Enterprise JavaBeans) 기술에 기반을 두고 있다. 우리의 모델에서 상위 레벨 차별화 서비스 QoS 정책들은 XML 스키마를 갖는 유효한 XML 문서들로 표현되고 EJB 기반 정책 서버 내에 하위 레벨 EJB 정책 빈들로 변환된다. QoS 정책을 정의하기 위해 필요한 라우팅 형상과 라우터의 역할 정보 검출은 SNMP MIB-II를 사용하고, QoS 정책 분배와 모니터링은 SNMP DiffServ MIB를 사용한다.

A Policy Based Management Model of Quality of Service for Differentiated Services Networks

Si-Ho Cha[†]·Young-Man Kang^{††}·Kuk-Hyun Cho^{†††}

ABSTRACT

Differentiated Services (DiffServ) is a technique to provide Quality of Service (QoS) in an efficient and scalable way. However, current DiffServ specifications have limitations in providing the complete QoS management framework and its implementation model. This paper proposes a policy-based QoS management model that supports DiffServ policies for managing QoS of DiffServ networks. The management model conforms to Model-View-Controller (MVC) architecture, and is based on Enterprise JavaBeans (EJBs) technologies. In our model, high-level DiffServ QoS policies are represented as valid XML documents with an XML Schema and are translated to low-level EJB policy beans in the EJB-based policy server. The routing topology and role information required to define QoS policies is discovered by using SNMP MIB-II, and the QoS policy distribution and monitoring is accomplished by using SNMP DiffServ MIB.

키워드 : 차별화 서비스(DiffServ), QoS 관리(QoS Management), 정책 기반 관리(Policy-based Management), 서비스 관리(Service Management), 망 관리(Network Management)

1. 서 론

현재의 IP 망은 기존의 최선형(best-effort) 서비스 모델에서 QoS(Quality of Service)를 지원하기 위하여 다양한 서비스 레벨을 제공할 수 있는 새로운 서비스 모델로 변화되고 있다. 최선형 서비스 모델에서의 트래픽은 가능한 한 빠르게 처리되지만 QoS는 보장되지 않는다. 이러한 문제를 해결하기 위하여 IETF(Internet Engineering Task Force)는 통합 서비스(IntServ) 모델[1]과 차별화 서비스(DiffServ) 모델[2]을 제시하였다. 통합 서비스 모델은 자원 예약 프로토콜(RSVP)[3]을 사용하여 플로우마다 자원 예약과 접근 관리를 수행한다. 따라서, 통합 서비스 모델에서는 플로우마다

상태 정보가 필수적으로 요구되고 코어 IP 망 내에서의 QoS 처리로 인하여 심각한 규모의 문제가 발생한다[4]. 그러나, 차별화 서비스 모델은 개별적인 플로우들 보다는 집합적인 트래픽 클래스들을 지원하기 때문에, IP 망 내의 다양한 패킷 클래스들에 대하여 차별화된 QoS를 제공할 수 있다. 그럼에도, 현재의 차별화 서비스 모델의 규격들은 완전한 QoS 관리 프레임워크와 구현 모델을 제공하는 데는 취약한 실정이다. 현재의 차별화 서비스 관련 RFC들과 초안(drafts)들은 주로 QoS 제공과 망 구성을 다루고 있으며, 완전한 QoS 관리 프레임워크와 구현 모델은 아직 세부적으로 기술되지 않고 있다[5]. 따라서, 차별화 서비스 망에서의 QoS를 제공하기 위해서는 망 및 서비스 관리의 지원이 필수적이다. 이런 지원이 없이는 완전한 QoS를 제공하는 것이 불가능할 것이다.

따라서, 본 논문은 차별화 서비스 망의 QoS를 관리하기

† 준 회 원 : 광운대학교 대학원 컴퓨터과학과
 †† 정 회 원 : 여수대학교 정보기술학부 교수
 ††† 정 회 원 : 광운대학교 컴퓨터공학부 교수
 논문접수 : 2002년 11월 20일, 심사완료 : 2003년 1월 24일

위하여 QoS 정책에 기반한 관리 모델을 제시한다. 정책 기반 QoS 관리란 상위 레벨의 비즈니스 규칙들이나 명령들을 사용하여 망의 서비스 품질을 관리하는 것을 의미한다. 본 논문의 구현 모델은 모델-뷰-제어 구조를 준수하며, EJB 프레임워크[6]에 기반을 두고 있다. 상위 레벨 차별화 서비스 QoS 정책들은 정책의 유효성 검증을 위하여 XML 스키마를 갖는 유효한 XML 문서들로 작성되고, 유효성이 검증된 상위 레벨 QoS 정책들은 하위 레벨 EJB 정책 빈들로 변환된다. QoS 정책을 정의하기 위해 필요한 라우팅 형상과 라우터의 역할 정보의 검출은 SNMP MIB(Management Information Base)-II를 사용하고, QoS 정책 분배와 모니터링은 SNMP DiffServ MIB[7]를 사용한다.

본 논문의 2장에서는 관련 기술에 대하여 기술하고, 3장에서는 차별화 서비스 망의 QoS 관리를 위한 정책 기반 QoS 관리 모델에 대하여 자세히 기술한다. 4장에서는 리눅스 기반 차별화 서비스 라우터와 3장에서 제시한 관리 모델의 구현에 대하여 살펴본다. 마지막 5장에서는 본 논문의 요약 및 향후의 연구 과제에 대하여 기술한다.

2. 관련 연구

차별화 서비스 망은 차등 서비스 코드 포인트(DSCP)의 값에 따라 다양한 PHB(Per Hop Behavior)들을 수행함으로써 집합적인 IP 패킷 스트림들에 차별화된 서비스를 제공한다. PHB란 각 라우터에서 수행할 차별화 서비스의 행위 즉, 각 차별화 서비스 라우터들에서 패킷들에 행해지는 처리를 말한다. DSCP란 단순히 IP 헤더 내의 차등 서비스(DS) 필드의 6비트 값을 의미하며, 이 값은 해당 패킷에 제공되어야 할 특정한 PHB를 나타낸다. 현재의 차별화 서비스 망에는 기본 서비스(best-effort), 클래스 선택자(CLS), 신속 전송(EF), 보장 전송(AF)의 4가지의 PHB가 사용되고 있다.

차별화 서비스 망에서 QoS 보장 서비스를 제공하기 위해서는 QoS 관리가 필수적이며, 정책 기반 관리 기법이 망 및 서비스 관리 분야에서 가장 주목을 받고 있다. 정책 기반 관리란 상위 레벨의 비즈니스 규칙들과 명령들을 사용하여 망의 장비들 및 망의 형상과 가능한 한 독립적인 방법으로 망의 행위를 관리하는 것을 의미한다. 이러한 정책 기반 관리의 개념은 Morris Sloman[8]에 의하여 처음으로 제시되었으며, 차별화 서비스 망에서 QoS를 관리하기 위한 정책 규칙들을 기술하기 위하여 Ponder라는 언어를 개발하였다. S. Boros[9]는 정책 기반 구성 관리를 위하여 SNMP를 사용하는 방법을 제시하였고, Yasusi Kanada[10, 11]는 그의 여러 논문에서 차별화 서비스 정책들의 병합과 분리에 대한 많은 방법들을 제시하고 있다. Paris Flegkas 등[12]은 차별화 서비스 망에서의 자원 관리 특성들에 대한 기능을 제시하고 이를 구현하기 위하여 IETF Script MIB[13]를 사용하는 구조를 제안하였으며, 이 방법은 포괄적인 QoS 관리

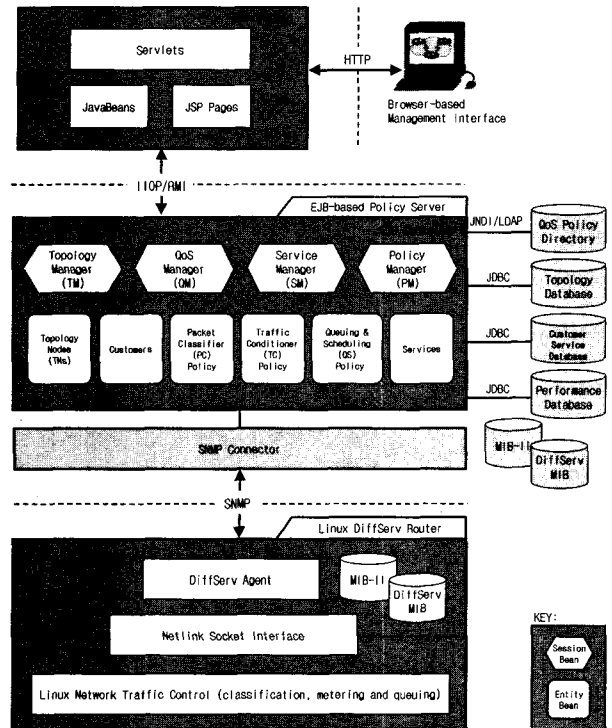
를 위한 미시적인 해결책이다. QoS 모니터링은 Y. Jiang 등[14]과 포항공대 홍원기 교수등[5]에 의해 수행되었으며, 이들은 전통적인 망 관리와 QoS 모니터링을 통합하는 솔루션에 대해 연구하고 있지만, 정책 기반 망 관리의 개념은 포함하지 않고 있다.

최근에 우리는 서비스 객체를 컴포넌트화 함으로써 새로운 QoS 보장 서비스들을 신속하고 안전하게 구축할 수 있는 컴포넌트 기반 운영 지원 시스템 플랫폼인 COSS 플랫폼[15]을 제안하였다. COSS 플랫폼은 EJB 프레임워크에 기반을 두고 있기 때문에 EJB가 제공하는 다양한 이점들을 얻을 수 있다. 본 논문에서 제안한 차별화 서비스 망을 위한 정책 기반 서비스 품질 관리 모델은 이 COSS 플랫폼 상에서 설계되고 구현된다.

3. 관리 모델의 설계

3.1 구조

차별화 서비스 망을 위한 정책 기반 QoS 관리 모델의 전체적인 구조는 (그림 1)과 같다. 본 관리 모델은 모델-뷰-컨트롤러 구조[15, 16]와 세션 패사드(Session Facade) 패턴[17]을 준수하여 설계되었기 때문에 고도의 규모성을 가지며, 관리 서비스에 포함된 객체들의 명확한 분산 전략을 제공한다.



(그림 1) 차별화 서비스 망을 위한 정책 기반 QoS 관리 모델

본 관리 모델은 크게 웹 서버와 EJB 기반 정책 서버로 구성된다. 웹 서버는 본 관리 모델의 표현을 담당하고 EJB 기반 정책 서버는 본 관리 모델의 비즈니스 로직을 담당한다.

다. 본 관리 모델에서는 차별화 서비스 망 상에서 수행될 상위 레벨의 QoS 정책들을 생성하고 관리하기 위한 사용자 인터페이스로 웹 브라우저를 사용한다.

본 관리 모델의 EJB 기반 정책 서버를 구성하는 구성 요소들은 크게 형상과 역할 검출, 하위 레벨 QoS 정책의 설정과 분배, 고객 서비스 관리, 그리고 정책 서버의 인터페이스 등으로 구성된다. 이들 중에서 차별화 서비스 망의 형상과 라우터들의 역할 정보 검출과 관련된 요소들은 다음과 같다.

- 형상 노드(TN : Topology Node) 빈 : 망의 라우팅 형상과 라우터의 역할 정보를 포함하는 엔티티 빈으로, 이들에 대한 정보는 SNMP MIB-II를 사용하여 검출된다.
- 형상 데이터베이스(TD : Topology Database) : 망으로부터 검출된 형상과 역할 정보를 저장한다
- 형상 관리자(TM : Topology Manager) 빈 : 형상 정보와 역할 정보 등의 검출과 이들의 정보를 형상 데이터베이스에 저장하고, 형상 노드 엔티티빈에 설정하는 일련의 행위를 처리하고 관리하는 세션 빈이다.

하위 레벨 QoS 정책의 생성과 분배를 수행하기 위하여 필요한 요소들은 다음과 같다.

- 패킷 분류자(PC : Packet Classifier) 정책 빈 : 패킷 흐름들 내의 패킷들을 분류하고 클래스 식별자(CID)라 불리는 레이블을 패킷들에 할당하는 엔티티 빈이다. PC 빈의 행위는 에지 라우터들로 분배되어 차별화 서비스 망으로 입력되는 트래픽을 제어하게 된다.
- 트래픽 조절자(TC : Traffic Conditioner) 정책 빈 : 패킷 분류자에서 선별된 패킷들이 트래픽 프로파일의 적합성 여부를 측정하고 이 결과에 따라 해당 패킷을 마킹하고, 폐기하거나 셰이핑하는 엔티티 빈이다. TC 빈의 행위는 에지 라우터들로 분배되어 에지 라우터로부터 출력되는 트래픽을 제어하게 된다.
- 큐잉과 스케줄링(QS : Queuing & Scheduling) 정책 빈 : 패킷들을 큐잉하고, 스케줄링하거나 폐기하는 행위를 수행하는 엔티티 빈이다. QS 빈의 행위는 코어 라우터들로 분배되어 코어 라우터로부터 출력되는 트래픽을 제어하게 된다.
- QoS 정책 디렉토리(QPD : QoS Policy Directory) : 하위 레벨 QoS 정책 빈들을 저장하는 디렉토리이다.
- 정책 관리자(PM : Policy Manager) 빈 : 유효한 XML 문서로 생성된 상위 레벨 QoS 정책들을 하위 레벨 QoS 정책들로의 변환을 수행하고, QoS 정책에 따라 차별화 서비스 라우터들 상의 DiffServ MIB 변수 값들을 적절히 설정함으로써 QoS 정책들을 차별화 서비스 라우터들로 분배하는 세션 빈이다.
- QoS 관리자(QM : QoS Manager) 빈 : 실제 정책 실행

자인 차별화 서비스 라우터들로부터 DiffServ MIB의 값을 검출함으로써 QoS 모니터링을 수행하고, 분배된 QoS 정책들이 제대로 반영되었는지를 감시하는 세션 빈이다. 만일 QoS 위배가 발견되면 QoS 관리자는 적절한 처리가 운영자에 의해 수행될 수 있도록 운영자에게 그 정보를 알려주어야 한다.

차별화 서비스 망의 고객들을 위한 서비스 관리를 수행하는 서비스 관리 요소들은 다음과 같다.

- 고객 빈 : 하나의 고객 빈은 한 명의 고객에 대한 뷰(view)를 표현하는 엔티티 빈이다. 각각의 고객 빈은 각각의 고객을 위한 질의를 위해 고객 서비스 데이터베이스(Customer Service Database)의 테이블들에 대한 데이터 접근을 수행한다.
- 서비스 빈 : 하나의 서비스와 관련된 서비스 유형을 나타내는 엔티티 빈이다. 서비스 빈의 하나의 인스턴스가 고객을 위해 제공되는 하나의 서비스를 나타내게 된다.
- 서비스 관리자(SM : Service Manager) 빈 : 서비스 엔티티 빈, 고객 엔티티 빈들과 함께 이 서비스 관리자 세션 빈은 서비스 관리 시스템을 위해 필요한 EJB 빈이다. 서비스 관리자 세션 빈은 서비스 관리와 관련된 다른 빈들에 대한 접근을 제공하고 서비스 관리 업무를 처리한다.

EJB 기반 정책 서버에서 다음과 같은 인터페이스들과 커넥터가 사용된다.

- JDBC(Java Database Connectivity) : JDBC 드라이버는 EJB 컨테이너를 형상 데이터베이스, 고객 서비스 데이터베이스, 그리고 성능 데이터베이스로 연결시키는 역할을 수행한다.
- JNDI(Java Naming and Directory Interface)/LDAP(Lightweight Directory Access Protocol) : JNDIAPI와 LDAP은 QoS 정책 디렉토리를 위한 룩업(lookup) 서비스를 제공한다.
- SNMP 커넥터 : SNMP 커넥터는 망 형상과 라우터 역할 검출, QoS 정책 분배와 모니터링, 그리고 서비스 관리 등의 행위를 수행하기 위한 SNMP 프로토콜을 지원한다.

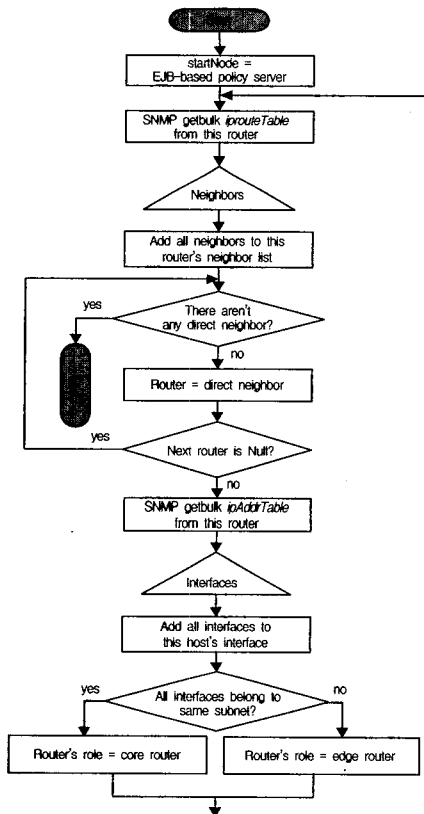
그리고, 제안한 관리 모델의 실험을 위하여 리눅스 기반의 차별화 서비스 라우터들을 구축하고, 각각의 라우터들에 DiffServ MIB[7]와 MIB-II를 지원하는 SNMP 에이전트가 추가된다.

3.2 라우팅 형상과 라우터의 역할 검출

차별화 서비스 망의 QoS를 관리하기 위해서는 우선적으로 모든 차별화 서비스 망의 라우팅 형상 정보와 망 내의

라우터들의 역할 정보가 필요하다. TM 세션 빈은 SNMP를 사용하여 라우팅 형상 정보와 라우터의 역할 정보를 검출한다. 검출된 라우팅 형상 정보와 라우터의 역할 정보는 TD에 저장되며 EJB 기반 정책 서버의 EJB 컨테이너 내에 TN 엔티티 빈들로 표현된다. 이러한 일련의 작업들은 TM 세션 빈에 의해 관리된다. 라우팅 형상 정보와 라우터의 역할 정보는 두 개의 SNMP MIB-2 테이블 즉, *ipAddrTable* 과 *ipRouteTable*을 사용함으로써 검출된다. 라우팅 형상 정보는 *ipRouteTable*을 사용함으로써 얻을 수 있고, 라우터의 역할 정보는 *ipAddrTable*을 검색함으로써 얻을 수 있다. *ipAddrTable*은 라우터 내의 모든 망 인터페이스의 IP 주소를 포함하며, *ipRouteTable*은 목적지 IP 주소들을 위한 다음으로 연결된 호스트와 망 인터페이스를 갖는 IP 라우팅 테이블을 포함한다.

(그림 2)는 라우팅 형상과 라우터의 역할을 검출하기 위한 알고리즘 흐름도이다. 라우팅 형상과 라우터 역할 검출 알고리즘은 EJB 기반 정책 서버로부터 시작된다. 따라서, EJB 기반 정책 서버는 자신의 시스템에 SNMP 에이전트를 가져야만 하고, 차별화 서비스 망의 최소한 하나의 에지 라우터와 직접적으로 연결이 설정되어 있어야만 한다. 이 알고리즘은 우선 *ipRouteTable*을 *getbulk* 함으로써 자신과 라우팅을 수행 중인 모든 이웃 라우터들의 리스트를 얻을 수 있



(그림 2) 라우팅 형상과 라우터의 역할 검출을 위한 알고리즘 흐름도

다. 그리고, *ipAddrTable*을 *getbulk* 함으로써 자신이 가지고 있는 모든 인터페이스들에 대한 정보를 얻을 수 있다. 자신의 모든 인터페이스가 같은 서브넷에 속하면 자신의 역할은 코어 라우터가 되고, 그렇지 않으면 에지 라우터가 된다. 이러한 일련의 절차가 모든 이웃 라우터들에 반복적으로 적용됨으로써, 차별화 서비스 망의 모든 라우팅 형상 정보와 각 라우터들의 역할을 검출할 수 있게 된다.

3.3 QoS 정책 관리

QoS 정책 관리 절차는 정책 생성, 정책 유효성 검증, 정책 변환, 정책의 분배, 그리고 QoS 모니터링으로 구성된다.

3.3.1 정책의 생성, 유효성 검증 및 변환

가장 기본적인 정책 검증은 정책 구문에 대한 유효성을 검증하는 것이다. 본 논문의 관리 모델에서는 XML 문서로 표현된 상위 레벨 QoS 정책들의 유효성을 검증하기 위하여 XML 스키마를 정의하였다(<표 2> 참조). 웹 서버 내의 자바 서블릿은 관리자로부터 정책 데이터를 입력으로 받아 들여 유효한 XML 문서를 생성한 후 XML 스키마를 이용하여 유효성 검증을 수행한다. 자바 서블릿이 정책의 유효성 검증을 완료한 후에는 PM 세션 빈에 하위 레벨 QoS 정책 엔티티 빈들의 인스턴스를 생성할 것을 요청하게 된다. PM 세션 빈은 앞에서 설명한 세 가지의 QoS 정책 엔티티 빈들의 변수 속성 값들을 적절히 설정함으로써 정책 변환을 제공한다. 에지 라우터와 관련된 상위 레벨 정책들은 PC 정책 엔티티 빈들과 TC 정책 엔티티 빈들에 적용되며, 코어 라우터와 관련된 상위 레벨 정책들은 QS 정책 엔티티 빈들에 적용된다. 다음의 *policy1*은 가장 간단한 형태의 정책의 한 예를 보인 것이고 <표 1>은 이 정책을 정의한 유효한 XML 문서를 간략히 보인 것이다. 그리고, <표 2>는 모든 정책 XML 문서의 유효성 검증을 위하여 본 관리 모델에서 정의한 XML 스키마를 간략히 보인 것이다.

policy1 : IF (Source_IP_Address is 192.168.1.2)
THEN COS is Gold ;

<표 1> *policy1*을 위한 유효한 XML 정책 문서

```

<policyList >
  <policy id = "1">
    <sourceGroup >
      <source > 192.168.1.2 </source >
    </sourceGroup >
    <destinationGroup >
      <destination > any </destination >
    </destinationGroup >
    <serviceGroup >
      <service > any </service >
    </serviceGroup >
    ....
    <cos> Gold </cos >
  </policy >
</policyList >
    
```

〈표 2〉 모든 XML 정책 문서들의 유효성 검증을 위한 XML 스키마

```

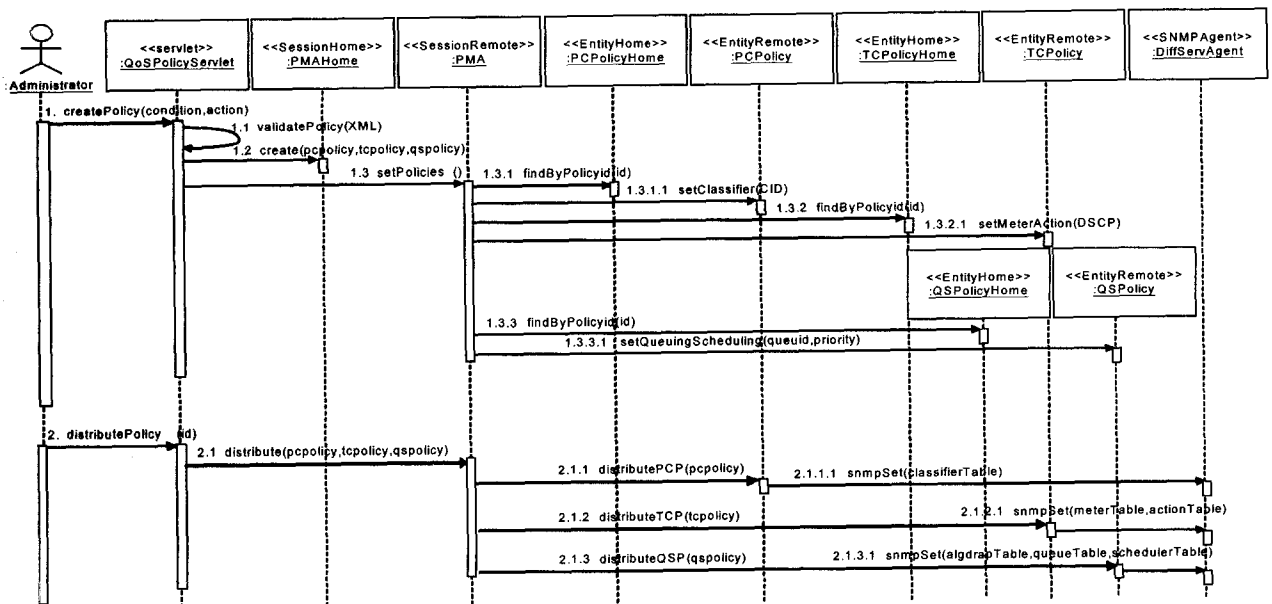
<?xml version = "1.0"?>
<xsd:schema xmlns:xsd = "http://www.w3.org/2000/10/
XMLSchema">
<xsd:element name = "policyList" type = "policyList" />
<xsd:complexType name = "policyList">
<xsd:sequence>
<xsd:element name = "policy" type = "policy"
minOccurs = "1" maxOccurs = "unbounded" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name = "policy">
<xsd:sequence>
<xsd:element name = "sourceGroup"
type = "sourceGroup" />
<xsd:element name = "destinationGroup"
type = "destinationGroup" />
<xsd:element name = "serviceGroup"
type = "serviceGroup" />
<xsd:element name = "timeGroup"
type = "timeGroup" />
<xsd:element name = "cos" type = "cosEnum" />
</xsd:sequence>
<xsd:attribute name = "id" type = "xsd:int" />
</xsd:complexType>
<xsd:complexType name = "sourceGroup">
<xsd:sequence>
<xsd:element name = "source" type = "xsd:string"
minOccurs = "1" maxOccurs = "unbounded" />
</xsd:sequence>
</xsd:complexType>
....
<xsd:simpleType name = "cosEnum">
<xsd:restriction based = "xsd:string">
<xsd:enumeration value = "Premium" />
<xsd:enumeration value = "Gold" />
<xsd:enumeration value = "Silver" />
<xsd:enumeration value = "Bronze" />
</xsd:restriction>
</xsd:simpleType>
</xsd:schema>
    
```

3.3.2 정책의 분배

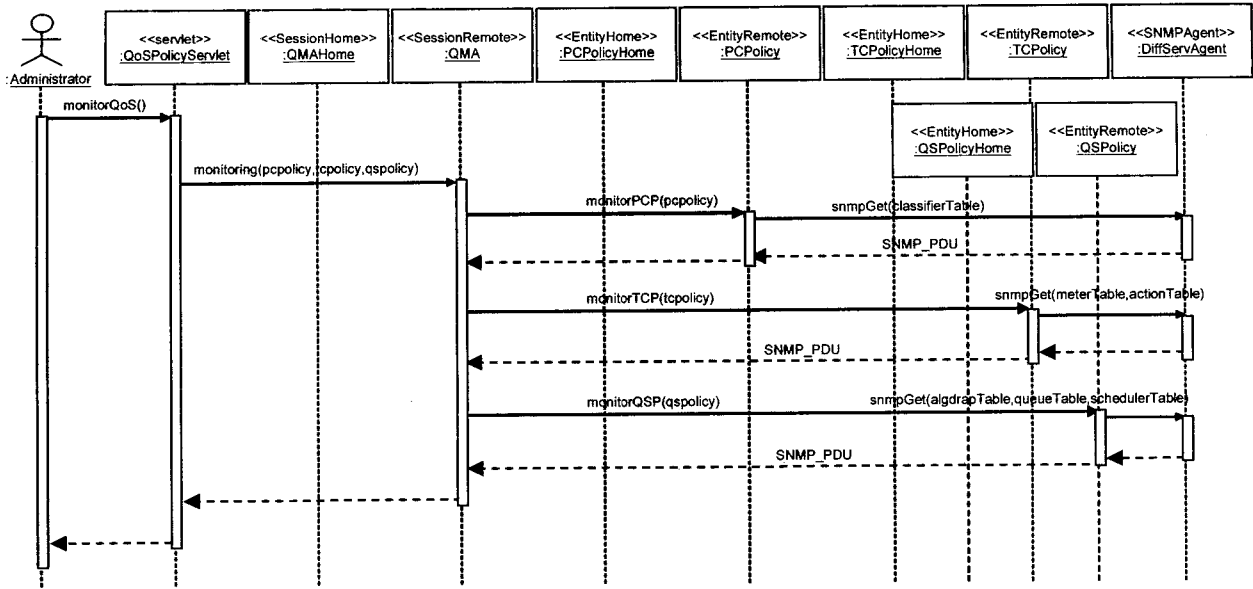
정책의 분배는 PM 세션 빈과 하위 레벨 QoS 정책 엔티티 빈들인 PC 빈, TC 빈, 그리고 QS 빈을 사용하여 이루어진다. (그림 3)은 일련의 정책 분배의 절차를 보여주는 UML 순차 다이어그램이다. PC 정책 빈은 패킷의 흐름들을 분류하여 그들에 분류 식별자를 할당하고, TC 정책 빈은 패킷들을 측정하여 그들에 마킹, 카운팅, 그리고 폐기 등의 행위를 수행한다. QS 정책 빈은 패킷들을 큐잉하고, 스케줄링하거나 폐기하는 행위를 수행한다. PM 세션 빈은 이러한 하위 레벨 QoS 정책 엔티티 빈들을 사용함으로써 정책 변환을 수행하고, 정책 분배를 수행하게 된다. 하위 레벨 QoS 정책 빈들은 SNMP 오퍼레이션을 사용하여 DiffServ MIB[7]에 대한 적절한 값의 설정과 모니터링을 수행해야 한다. (그림 3)에서 보인 것처럼, PM 세션 빈은 PC 정책 빈과 TC 정책 빈 그리고 QS 정책 빈으로 하여금 SNMP를 사용하여 정책 분배를 수행하도록 한다. DiffServ MIB는 차별화 서비스 노드의 구성과 관리 특성들을 기술하고 있다. 이 MIB는 다양한 테이블들을 사용하는 데이터 패스(datapath)의 기능적인 요소들을 포함한다. 이 DiffServ MIB는 RowPointer들이 하나의 데이터 패스로 다양한 기능적인 요소들을 결합하기 위하여 사용된다. 이 DiffServ MIB 테이블들은 분류자(classifier), 측정자(meter), 수행(action) 및 큐(queue)와 같은 차별화 서비스의 구조적인 요소들로 분류된다. 본 관리 모델에서 정책 분배를 위해 SNMP를 사용하는 주된 이유는 정책 분배가 모든 라우터들에 표준적인 방법으로 이루어질 수 있다는 것이다.

3.3.3 QoS 모니터링

분배된 정책들은 정책 정의에 기술된 내용에 맞게 동작하



(그림 3) 정책 분배를 위한 UML 순차 다이어그램



(그림 4) QoS 모니터링을 위한 UML 순차 다이어그램

지 않을 수 있다. 즉, 정책 분배로부터 발생된 QoS의 결과가 실제 정책과 다를 수 있다는 것이다. 따라서, 관리자는 정책 분배로부터 발생된 QoS를 모니터링 해야만 한다. QoS 모니터링 또한 SNMP DiffServ MIB를 사용한다. QM 세션 빈은 PC 정책 빈과 TC 정책 빈 그리고 QS 정책 빈에 정의된 정책들에 접근하여 실제 망으로부터 모니터링된 결과를 비교하게 된다. 만일 정책에 정의된 만큼의 QoS를 제공하고 있지 못하면, QM 세션 빈은 관리자에게 해당 메시지를 발생시키고 성능 데이터베이스를 갱신시키게 된다. (그림 4)는 QoS 모니터링의 절차를 보여주는 UML 순차 다이어그램이다.

4. 구현 및 실험

4.1 리눅스 기반 차별화 서비스 망

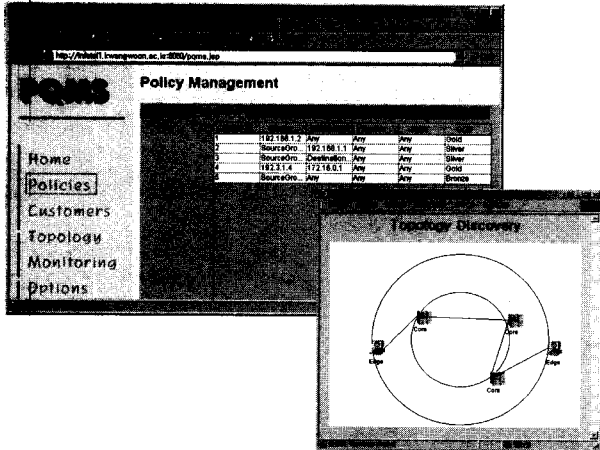
차별화 서비스 라우터는 리눅스 시스템을 사용하여 설정하였으며 SNMP MIB-2와 DiffServ MIB[7]를 처리하기 위한 SNMP 에이전트를 각각의 라우터에 추가하였다. 리눅스 시스템의 커널 버전 2.4 이후 버전에는 차별화 서비스를 위한 모듈이 이미 포함되어 있기 때문에 커널을 재구성하고 재컴파일 함으로써 차별화 서비스 라우터를 생성할 수 있다[18]. MIB-2와 차별화 서비스 MIB를 지원하는 SNMP 에이전트는 에이전트 확장성을 제공하는 UCD-SNMP 4.2.5 [19]를 이용하여 구현하였다. 차별화 서비스 MIB를 지원하는 SNMP 에이전트를 구현하기 위하여 차별화 서비스 MIB 모듈을 구현한 *diffServMibAgent.h*와 *diffServMibAgent.c* 파일을 UCD-SNMP 소스 디렉토리의 *agent/mibgroups* 서브 디렉토리에 배치하고 재컴파일 함으로써 UCD-SNMP 에이전트를 확장시킬 수 있다. 사용자 공간에 있는 이 SNMP 에이전트와 커널 공간에 있는 리눅스 트래픽 제어 커널 간

의 통신은 Netlink 소켓[20]을 사용하였다. Netlink는 커널 레벨과 사용자 공간 프로세서간의 정보 전달을 위해 사용되는 양방향 통신 링크이다. 이렇게 구현된 SNMP 에이전트는 EJB 기반 정책 서버 내의 하위 레벨 QoS 정책 빈들로부터 관리 오퍼레이션을 수신하여 리눅스 트래픽 제어 커널 내의 적절한 파라미터 변경을 수행한다. 이런 오퍼레이션의 결과로써 QoS 정책들이 정책 실행자인 리눅스 차별화 서비스 라우터들로 분배되고 이들의 결과에 대한 모니터링이 가능하게 된다.

4.2 QoS 관리 시스템

본 논문에서 제안한 차별화 서비스를 위한 QoS 관리 모델은 자바를 사용하여 Windows 2000 서버 플랫폼에서 구현된다. 차별화 서비스 QoS 관리 시스템은 다중 계층 MVC 구조[15]에서 표현 계층과 비즈니스 계층에 포함된다. 표현 계층은 관리자의 요청을 받아들여 서블릿이나 JSP를 실행시키는 웹 서버를 동작시키고, 비즈니스 계층은 하위 레벨 정책 빈들을 동작시키기 위한 EJB 컨테이너/서버를 동작시킨다. 웹 서버로는 Apache Tomcat 4.0.6[21]을 사용한다. Tomcat 4.0.6은 아파치 소프트웨어 재단의 Jakarta 프로젝트에서 개발된 오픈 소스 구현으로 서블릿 2.3과 JSP 1.2 규격을 구현한다. 각 QoS 정책 빈들을 관리하기 위한 EJB 컨테이너/서버로는 JBoss 3.0.3[22]을 사용하며, 모든 관리 빈들은 EJB 버전 1.1 표준을 사용하여 구현된다. JBoss는 오픈 소스이며 J2EE(Java 2 Enterprise Edition) 표준을 준수하고 100% 순수 자바로 구현된 어플리케이션 서버로 자유롭게 사용할 수 있도록 되어있다. 또한, EJB 기반 정책 서버의 SNMP 오퍼레이션을 처리하기 위하여 자바로 작성된 AdventNet SNMP v2c API[23]를 사용한다. AdventNet SNMP v2c API

는 망 관리 어플리케이션 개발자들이 망 관리 어플리케이션을 SNMP 인터페이스를 사용하여 쉽고 간단히 작성할 수 있도록 해주는 API이다. MIB 테이블로부터 얻어진 성능 정보 및 형상 정보 등을 저장하기 위한 데이터 베이스로는 Oracle 8i Enterprise Edition 버전[24]을 사용하여 구현된다. (그림 5)는 차별화 서비스 QoS 관리 시스템의 사용자 인터페이스이다.

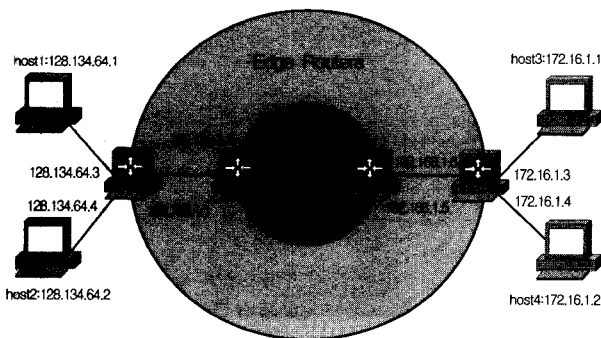


(그림 5) 차별화 서비스 QoS 관리 시스템을 위한 사용자 인터페이스

<표 1>과 <표 2> 그리고 (그림 5)와 같이, 하나의 정책은 정책 ID, 소스 IP 또는 소스 IP 그룹, 목적지 IP 또는 목적지 IP 그룹, 서비스 또는 서비스 그룹, 시간 또는 시간 그룹, 그리고 서비스 클래스(COS)로 구성된다. 만일 이들에 어떠한 값도 설정되지 않으면, 이들의 값은 기본적으로 any로 설정된다. COS는 고객들에 제공될 서비스 등급을 의미하며, 본 구현에서는 프리미엄, 골드, 실버, 브론즈 중 하나의 값을 갖게 된다.

4.3 실험

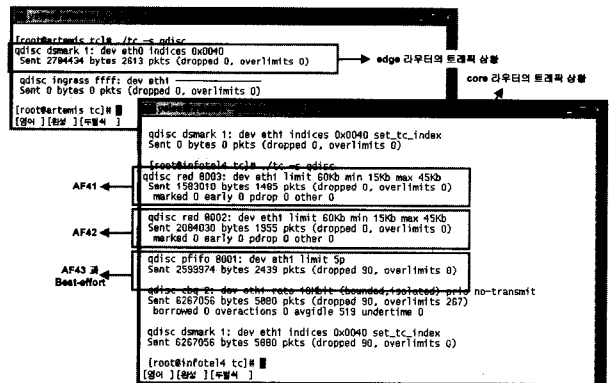
본 구현에서 구성된 리눅스 기반 차별화 서비스 망의 구성은 (그림 6)과 같으며, 호스트 1에서 호스트 4까지의 경로는 각 장비의 라우팅 정보를 수정하여 절대 경로로 설정하였다.



(그림 6) 리눅스 차별화 서비스 망의 구성도

(그림 6)의 구성도에서 에지 라우터의 입력 링크에서는 흐름의 소스가 호스트 1인 경우 흐름의 속도와 버스트(burst)를 측정하여 각 흐름을 세 개의 흐름으로 분류한다. 그리고 호스트 2의 주소를 가진 흐름의 경우는 속도와 버스트가 한계점을 넘는 경우 폐기시킨다. 출력 링크에서는 입력 링크에서의 결과에 따라 패킷을 네 개의 흐름으로 나누고 각 흐름에 DSCP 값을 마킹한다. 마킹의 결과로 각각의 흐름은 보장 전송(AF)의 AF41, AF42, AF43과 기본 전송(Best-effort)로 나뉘어져서 코어 라우터로 전송된다. 코어 라우터에서는 패킷의 DS 필드를 조사하여 DSCP를 얻고 DSCP에 따라 알맞은 PHB를 수행하게 된다. DSCP 값의 추출과 저장은 dsmark 큐잉 정책에 의해 수행되며 DSCP 값에 따라 올바른 클래스로 패킷을 분류하는 역할은 tcindex 필터가 담당한다. 코어 라우터에서는 AF41과 AF42를 각각 클래스 2:1과 클래스 2:2로 분류하고 나머지 패킷들은 클래스 2:3으로 분류한다.

(그림 7)은 실험의 결과로 첫 번째 에지 라우터에서의 트래픽 상황과 첫 번째 코어 라우터에서 발생하는 트래픽 상황을 보여준다.



(그림 7) 에지 라우터와 코어 라우터에서의 차별화된 서비스 수행

5. 결론

본 논문에서는 차별화 서비스 망을 위한 정책 기반 QoS 관리 모델을 제시하였다. 이 모델은 다중 계층 MVC 모델에 따라 설계되었으며, IETF 정책 프레임워크와 EJB 기술에 기반하여 설계되었다. 이 모델에서 상위 레벨 차별화 서비스 QoS 정책들은 정책 검증을 위하여 XML 스키마를 갖는 유효한 XML 문서들로 기술되고 EJB 기반 정책 서버의 EJB 컨테이너 내에 하위 레벨 QoS 정책 빈들로 변환된다. 이들 하위 레벨 QoS 정책 빈들은 에지 라우터에 배치되는 PC 정책 빈과 TC 정책 빈 그리고 코어 라우터에 배치되는 QS 정책 빈으로 구성된다. QoS 정책을 정의하기 위해 필요한 라우팅 형상과 라우터의 역할 정보 검출은 SNMP MIB-II를 사용하고, QoS 정책 분배와 QoS 모니터링은 SNMP 차별화 서비스 MIB를 사용한다. 이러한 연구를 지원하기 위

하여 리눅스 기반 차별화 서비스 라우터들에 차별화 서비스 MIB를 포함하는 SNMP 에이전트를 구현하였다.

현재 우리는 EJB 기반 정책 서버의 비즈니스 로직을 구현하고 있으며, 리눅스 기반 라우터를 사용하는 연구실의 테스트베드 상에서 우리의 시스템을 시험할 계획이다. 또한 NS-2를 사용하여 다양한 망 구성과 다양한 QoS 정책들을 적용하는 실험을 통하여 차별화 망에서의 정책 기반 QoS 관리의 필요성을 입증하려고 한다.

참 고 문 헌

[1] R. Braden, D. Clark, S. Shenker, "Integrated Services in the Internet Architecture an Overview," IETF RFC 1633, June, 1994.

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, "An Architecture for Differentiated Services," IETF RFC 2457, December, 1998.

[3] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin, "Resource ReSerVation Protocol (RSVP) Version 1 Functional Specification," IETF RFC 2205, September, 1997.

[4] D. Goderis, S. Van den Bosch, Y. T' Joens, et al, "A Service-Centric IP Quality of Service Architecture for Next Generation Networks," NOMS 2002, pp.139-154, 2002.

[5] Jae-Young Kim, James Won-Ki Hong, "Distributed QoS Monitoring and Edge-to Edge QoS Aggregation to Manage End-to-End Traffic Flows in Differentiated Services Networks," Journal of Communications and Networks (JCN), December, 2001.

[6] Sun Microsystems, Inc, "Telecom Network Management With Enterprise JavaBeans™ Technology-A Technical White Paper," May, 2001.

[7] F. Baker, K. Chen, A. Smith, "Management Information Base for the Differentiated Services Architecture," RFC 3289, 2002.

[8] Leonidas Lymberopoulos, Emil Lupu, Morris Sloman, "Policy Based Management of Differentiated Services Networks," Policy 2002, pp.147-158, June, 2002.

[9] S. Boros, "Policy-Based Network Management with SNMP," CTIT Tech. Report, No.00-16, October, 2001.

[10] Yasusi Kanada, "Policy Division and Fusion : Examples and A Method," IFIP/IEEE IM 2001, pp.545-560, May, 2001.

[11] Yasusi Kanada, Brian J. O'Keefe, "DiffServ Policies and Their Combinations in a Policy Server," IEICE SIG Information Networks & SIG Network Systems, March, 2002.

[12] Paris Flegkas, Panos Trimintzios, George Pavlou, "A Policy-Based Quality of Service Management System for IP DiffServ Networks," IEEE Network, pp.50-56, March/April, 2002.

[13] D. Levi and J. Schoenwaelder, "Difinitions of Managed Objects for the Delegation of Management Scripts," IETF RFC 2592, May, 1999.

[14] Y. Jiang, C. K. Tham, C. C. Ko, "Providing Quality of Service Monitoring : Challenges and Approaches," IEEE/IFIP NOMS 2000.

[15] Si-Ho Cha, Jae-Oh Lee, Kuk-Hyun Cho, "Towards a Component-based Operations Support System," APNOMS 2002,

pp.209-220, 2002.

[16] Sun Microsystems, "Designing Enterprise Applications with the Java™ 2 Platform, Enterprise Edition," Version 1.0.1, 2000.

[17] Floyd Mariescu, "EJB Design Patterns," John Wiley & Sons, 2002.

[18] Differentiated Services on Linux, <http://diffserv.sourceforge.net/>.

[19] UCD-SNMP 4.2.5, <http://www.net-snmp.org/>.

[20] Gowir Dhadapani et al., "Netlink Sockets Overview," Technical Paper, UK, September, 1999.

[21] Apache Org., "Jakarta Tomcat 4.0.6," <http://jakarta.apache.org/tomcat/>.

[22] JBoss Org., "JBoss Application Server," <http://www.jboss.org/>.

[23] AdventNet Inc., "AdventNet SNMP API v2c," <http://www.adventnet.com/products/>.

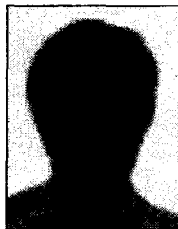
[24] Oracle Corporation, "Oracle 8i Enterprise Edition 8.1.6 for Windows NT," <http://otn.oracle.com/software/content.html>.



차 시 호

e-mail : sihoc@cs.kwangwoon.ac.kr
 1995년 순천대학교 전자계산학과(학사)
 1997년 광운대학교 대학원 전자계산학과 (이학석사)
 1997년~2000년 대우통신(주) 종합연구소 선임연구원
 2000년~현재 광운대학교 대학원 컴퓨터 과학과 박사과정 수료

관심분야 : 망관리 플랫폼, SLA/QoS 관리, 소프트웨어 컴포넌트 기술 등



강 영 만

e-mail : ymkang@cs.yosu.ac.kr
 1985년 광운대학교 전자계산학과(학사)
 1987년 광운대학교 대학원 전자계산학과 (이학석사)
 1985년~1991년 한국전자통신연구소 선임연구원

1991년~현재 여수대학교 정보기술학부 부교수
 1999년~현재 텔스전자 수석연구원 겸임
 2003년 현재 광운대학교 대학원 전자계산학과 박사과정 수료
 관심분야 : 무선 인터넷, 망관리 등



조 국 현

e-mail : khcho@cs.kwangwoon.ac.kr
 1977년 한양대학교 전자공학과(공학사)
 1984년 일본 동북대학교 대학원(공학박사)
 1984년~현재 광운대학교 전자정보대학 컴퓨터공학부 소프트웨어전공 교수
 개방형컴퓨터통신연구회(OSIA) 회장 역임

관심분야 : 망관리, 분산처리, 정보통신 분야의 표준화 등