중앙 집중형 네트워크 제어 플랫폼에서 SNMP 연결 관리의 고속화 방안 및 성능 분석

고 영 석[†] · 권 태 현[††] · 김 춘 희[†††] · 남 현 순[††] · 정 유 현[††††] · 차 영 욱[†††††]

요 약

차세대 네트워크의 성공적인 추진을 위하여 트래픽 엔지니어링이 보장되는 중앙 집중형 제어 및 관리 기술이 네트워크 제어 플랫폼인 NCP(Network Control Platform)와 서비스 품질을 보장하는 스위치인 QSS(Quality of Service Switch)로 실현되고 있다. 본 논문에서는 NCP와 QSS 사이의 SNMP 인터페이스에서 고속의 연결 관리를 위하여 병렬형 기법과 쓰레드 및 객체 풀을 도입하였다. 연결 관리의 테스트-베드를 구축하여 본 논문에서 도입한 고속화 방안을 실험실 환경에서 확인하였으며, 연결 설정의 지연과 완료율을 측정하여 성능을 비교 및 분석하였다. NCP와 QSS 사이의 SNMP 인터페이스에서 연결 관리의 고속화를 위하여 병렬형 방식과 객체 풀의 사용이 중요한 성능 파라미터임을 확인하였다.

키워드 : 중앙 집중형, 병렬형 연결 관리, 네트워크 제어 플랫폼 (NCP), SNMP, 서비스 품질 스위치 (QSS)

The Performance Analysis of A High-speed Mechanism for SNMP Connection Management in Centralized Network Control Platform

YoungSuk Ko[†] · TaeHyun Kwon[††] · ChoonHee Kim[†††] · HyunSoon Nam[††]
YouHyeon Jeong[††††] · YoungWook Cha[†††††]

ABSTRACT

Network Control Platform (NCP) and Quality of Service Switch (QSS) are being developed to realize centralized control and management technology, which is essential for guaranteeing traffic engineering and service quality in a next generation network. This paper adopts a parallel mechanism, and a thread and object pool to achieve high-speed connection management in the existing SNMP interface between NCP and QSS. We built up a connection management test-bed in laboratory environment to validate the functionality of high-speed connection management. We also measured and analyzed a performance of connection setup delay and a completion ratio using the test-bed. We ascertain that the parallel mechanism and the object pool are the most important performance parameters to achieve high-speed connection management in the SNMP interface between NCP and QSS.

Key Words : Centralization, Parallel Connection Management, NCP(Network Control Platform), SNMP, QSS(Quality of Service Switch)

1. 서 론

최근의 정보 통신 환경과 통신 시장은 차세대 네트워크의 실현이라는 목표를 향하여 기술의 통합과 융합이 이루어지고 있다. 해외에서는 NGN(Next Generation Network)[1]이라는 일반적인 용어로, 국내에서는 BcN(Broadband convergence Network)[2]이라는 용어로 통합과 융합에 대한 최적의 해법과 방안을 찾기 위한 연구 활동이 활발히 진행 중에 있다.

국내에서 추진 중인 BcN의 성공적인 추진을 위하여 트래픽 엔지니어링이 보장되는 중앙 집중형 제어 및 관리 기술이 네트워크 제어 플랫폼인 NCP(Network Control Platform)와 서비스 품질을 보장하는 스위치인 QSS(Quality of Service Switch)로 실현되고 있다[3]. 제어 및 관리 평면과 전달 평면의 기능이 분리된 네트워크에서 QSS 스위치들을 중앙 집중화된 방식으로 제어 및 관리하는 NCP는 토폴로지 관리, 경로 계산 및 자원 예약, 연결 관리와 같은 기능을 수행한다. QSS 스위치는 인접 노드에 대한 정보 수집 및 서비스 품질과 트래픽 엔지니어링을 제공하기 위한 흐름기반의 스위칭을 제공하는 전달 및 관리 평면의 기능을 수행한다[3,4].

※ 본 논문은 한국과학재단 우수연구센터(OIRC) 사업과 ETRI 정보통신 연구개발사업 위탁과제의 연구결과임
† 준 회 원: 안동대학교 컴퓨터공학과 석사과정
†† 정 회 원: 한국전자통신연구원 광대역통합망연구단 NCP기술팀 선임연구원
††† 정 회 원: 대구사이버대학교 컴퓨터정보학과 조교수
†††† 정 회 원: 한국전자통신연구원 광대역통합망연구단 NCP기술팀 팀장, 책임연구원
††††† 정 회 원: 안동대학교 컴퓨터공학과 부교수(교신저자)
논문접수 : 2007년 5월 15일, 심사완료 : 2007년 8월 6일

중앙 집중화된 네트워크 제어 플랫폼인 NCP와 QSS 스위치 사이의 연결 관리는 SNMP(Simple Network Management Protocol) 기반의 관리 방식[5]과 SNMP의 대체 방안으로 최근에 대두되고 있는 XML(eXtensible Markup Language) 기반의 웹 서비스 관리 방식[6,7] 그리고 GSMP(General Switch Management Protocol)를 사용하는 네트워크 개방형 인터페이스 기반의 관리 방식[8] 등이 가능하다. 현재, NCP와 QSS 스위치 사이에는 SNMP 기반의 연결 관리 방식을 이용하고 있다. 중앙 집중화된 NCP와 QSS 스위치들 사이에서 연결 제어의 고속화는 실시간 서비스를 요구하는 응용 서비스의 서비스 품질 보장을 위해서 필수적인 요구 사항이다. 특히, 연결 설정의 지연과 완료율은 서비스 품질을 보장하기 위한 중요한 성능 파라미터이다.

본 논문에서는 중앙 집중화된 NCP와 QSS 기반의 네트워크에서 사용되고 있는 SNMP 인터페이스에서 고속의 연결 관리를 위하여 병렬형 기법과 쓰레드 및 객체 풀을 도입하였다. 도입한 방안은 SNMP 매니저 응용의 동기와 비동기 동작 방식에 따라 쓰레드 및 객체 풀을 사용하는 병렬형 동기 쓰레드 풀 방식과 쓰레드 및 객체 풀을 사용하지 않는 병렬형 비동기 방식으로 구분된다.

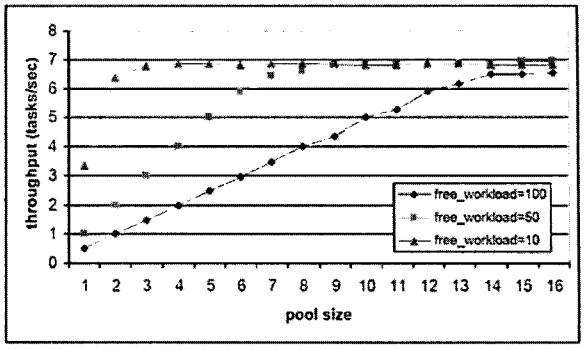
실험실 환경에서 SNMP 연결 관리의 테스트-베드를 구축하여 본 논문에서 도입한 고속화 방안을 실험실 환경에서 확인하였으며, 연결 설정의 지연과 완료율을 측정하여 성능을 비교 및 분석하였다.

2. 고성능 연결 관리의 기반 기술

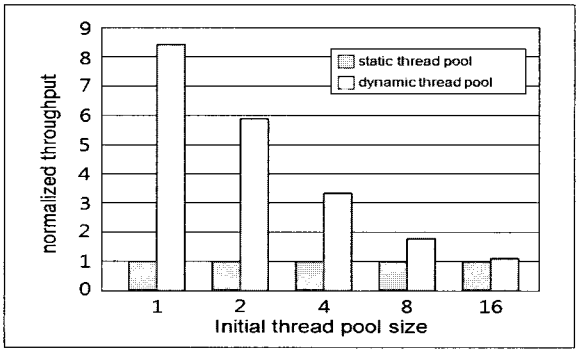
본 장에서는 고성능 연결 관리의 기반 기술에 대한 관련 연구 동향을 기술한다.

2.1 쓰레드 풀 기술

멀티 쓰레드 프로그래밍에서는 요청별 쓰레드 생성(thread-per-request) 또는 쓰레드 풀(thread pool) 모델이 사용된다. 쓰레드 풀 모델은 요청별 쓰레드 생성 모델보다 쓰레드의 생성과 소멸에 대한 오버헤드를 줄임으로써, 전체적인 성능을 개선시킬 수 있는 이점이 있다. 이러한 이점들 때문에 아파치 및 윈도우 IIS와 같이 유명한 서버 응용들은 쓰레드 풀 모델을 채택하고 있다. 쓰레드 풀은 멀티 쓰레드를 사용하는 응용에서 시스템의 성능을 개선하는데 도움이 되나, 시스템 성능의 개선 정도는 작업 부하 및 응용의 특성에 의존적이다. 쓰레드 풀 모델을 사용하는 시스템에서 성능을 결정짓는 중요한 요소 중 하나는 쓰레드 풀의 크기이다. (그림 1)과 (그림 2)는 참고문헌[9]에서 시스템의 처리율과 쓰레드 풀 크기의 관계, 그리고 정적 쓰레드 풀과 동적 쓰레드 풀의 성능을 정규화하여 비교한 그래프를 나타낸 것이다. (그림 1)에 따르면, 쓰레드 풀 크기가 상대적으로 작을 때에는 쓰레드 풀의 크기에 비례하여 처리율이 개선되나, 쓰레드 풀의 크기가 안정점(stable point)에 도달하면 처리율이 더



(그림 1) 시스템 처리율과 쓰레드 풀 크기의 관계



(그림 2) 정적과 동적 쓰레드 풀 방식의 처리율 비교

이상 증가되지 않고 유지된다. 쓰레드 풀의 크기가 계속 증가하여 하강점(degradation point)에 도달하면 처리율은 점차적으로 저하되며, 안정점과 하강점 사이를 안전 영역(safe zone)이라고 한다.

쓰레드 풀의 크기가 증가하면 빠른 응답 시간과 더불어 많은 태스크들을 동시에 처리할 수 있으나, 쓰레드 풀의 크기가 증가할수록 많은 수의 쓰레드들에 대한 관리 오버헤드가 발생하므로 적절한 풀의 크기를 유지하는 것이 중요하다. 쓰레드 풀은 계산이 요구되는 응용보다는 입출력 및 다른 데이터에 의존적인 응용에서 시스템의 성능을 개선시키는데 더 많은 이점이 있다. (그림 2)에 따르면, 안전 영역에 보다 빠르게 도달하여 안전 영역을 벗어나지 않도록 쓰레드 풀의 크기를 적절하게 유지하는 동적 쓰레드 풀을 사용한 응용이 정적 쓰레드 풀을 사용한 응용보다 우수한 처리율을 보여준다. 초기화된 쓰레드 풀 크기가 점차적으로 증가하면 성능 개선도 점차적으로 낮아지며, 쓰레드의 개수가 16개인 경우에는 정적과 동적 쓰레드 풀 방식의 성능이 유사함을 보이고 있다.

2.2 병렬형 연결 제어 기술

AT&T의 벨 연구소와 루슨트 테크놀러지(Lucent Technologies)는 ATM 망에서 중앙 집중화된 연결 제어 서버와 ATM 스위치들 사이의 추가적인 인터페이스에 대한 연결 제어의 지연을 개선하기 위하여 병렬형 연결 제어(PCC: Parallel Connection Control) 메커니즘을 제안하였다[10,11]. 제안된 병렬형 연결 제어 메커니즘은 중앙 집중화된 연결

서버가 ATM 스위치들에게 동시에 연결의 설정 및 해제를 요구하는 것이다. 하지만, 중앙 집중화된 연결 서버가 경로 계산을 위하여 스위치 망의 토폴로지만 관리할 뿐, 링크 상태와 스위치의 자원들은 각 스위치에 탑재된 스위치 자원 서버가 관리하므로 자원 예약 절차와 연결 구성 절차의 2 단계 절차가 요구된다. 즉, 연결 제어 서버는 단계 1에서 각 스위치에 탑재된 스위치 자원 서버에게 자원에 대한 가용 여부를 확인하며, 단계 2에서 스위치들에게 연결의 설정을 수행하게 하므로 전체적인 연결 설정의 지연이 길어지는 단점이 있다.

한편, 콜롬비아 대학에서는 ATM 스위칭 플랫폼을 위한 코바 기반의 개방형 프로그래머블 신호 시스템의 xbind를 위한 고성능 연결 제어 시스템을 제안하고 성능을 분석하였다[12]. 고성능의 연결 제어를 위하여 xbind 연결 매니저에 네트워크 상태 캐싱(caching), 연결 요청 메시지의 통합(aggregation) 그리고 연결 매니저와 스위치 서버 사이의 병렬형 연결 설정 방식을 적용하였다. 연결 식별자 캐싱을 이용하는 연결 매니저는 경로상의 스위치들에 대하여 가용한 출력 연결 식별자를 캐시에서 발견한다면 단일 단계(single phase)의 연결 설정을 수행하며, 캐시에서 발견하지 못하면 2 단계의 연결 설정을 수행하게 된다. 연결 식별자의 캐싱 크기와 메시지 통합 임계치(threshold)를 성능 제어 파라미터로 사용하였으며, 캐시 크기에 따른 연결 식별자의 히트율을 제시하고 있다. 또한, 병렬형 연결 제어와 더불어 부과되는 호의 도착율에 따라 메시지 통합 임계치를 적절히 조정함으로 처리율과 지연의 개선 효과를 제시하였다.

AT&T나 콜롬비아 대학의 연구에서는 경로상의 특정 스위치에서 연결 설정이 실패하는 경우에 대한 연결의 재설정 메커니즘이 정의되어 있지 않다.

3. 중앙 집중형 네트워크 제어 플랫폼에서 SNMP 연결 관리의 고속화 방안

본 장에서는 중앙 집중형 제어 및 관리 기능을 수행하는 NCP와 흐름기반의 스위칭을 지원하는 QSS 스위치들 사이에 적용되는 SNMP 인터페이스에서 연결 관리의 고속화를 위하여 단일 단계 기반의 병렬형 연결 관리 방안을 제시한다.

3.1 사설 MIB 기반 SNMP 연결 관리

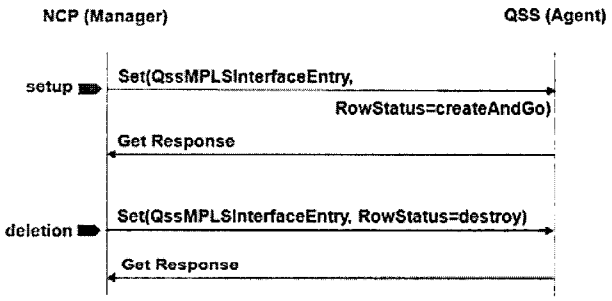
SNMP 기반의 연결 관리는 표준 MIB(Management Information Base)을 사용하는 방식과 사설 MIB을 사용하는 방식이 가능하다. IETF에서는 MPLS(Multiprotocol Label Switching)/GMPLS(Generalized MPLS)의 망 관리를 위하여 MPLS-FTN-STD-MIB, MPLS-LSR-STD-MIB, MPLS-TE-STD-MIB 그리고 GMPLS-LSR-STD-MIB, GMPLS-TE-STD-MIB 등의 표준 MIB을 정의하고 있다[13,14]. 이들 표준 MIB들은 LER(Label Edge Router) 및 LSR(Label Switching Router)에 모두 적용될 수 있으나, MPLS-FTN-STD-MIB은 입구 LER에만 적용된다. 표준

```
qssMPLSInterfaceTable OBJECT-TYPE
        SYNTAX          SEQUENCE OF QssMPLSInterfaceEntry
        Max-ACCESS      read-create
        STATUS          current
        DESCRIPTION     "MPLS Transit Node Configuration Info"
        ::= { qss120MplsMIB 3 }

qssMPLSInterfaceEntry OBJECT-TYPE
        SYNTAX          QssMPLSInterfaceEntry
        Max-ACCESS      read-create
        STATUS          current
        DESCRIPTION     "Row Description"
        INDEX           { qssIfName, qssInLabel }
        ::= { qssMPLSInterfaceTable 1 }

QssMPLSInterfaceEntry ::= SEQUENCE {
        qssIfName                       DisplayString,
        qssInLabel                      Integer32,
        qssNextHopIpAddr                DisplayString,
        qssLabelAction                  Integer32,
        qssSwapOutLabel                 Integer32,
        qssMPLSTELSPName                DisplayString,
        qssMPLSLSPType                  Integer32,
        qssMPLSIngressAddr              DisplayString,
        qssMPLSEgressAddr               DisplayString,
        qssMPLSLSPID                    Integer32,
        qssMPLSInterfaceRowStatus       RowStatus
        }
```

(그림 3) QSS120-MPLS-MIB의 연결 관리 테이블 및 엔트리



(그림 4) QSS120-MPLS-MIB을 사용한 연결 설정 및 해제 절차

MPLS-LSR-STD-MIB[15]을 사용하는 연결 관리에서는 입력 및 출력 세그먼트 테이블의 엔트리와 크로스-커넥트 테이블의 엔트리를 각각 별도의 SNMP 요청 및 응답 메시지를 이용하여 설정하므로 연결 설정의 지연이 발생 할 수 있다[16].

이러한 연결 설정 지연의 최소화를 위하여 하나의 SNMP 요청 및 응답 메시지로 연결을 설정할 수 있도록 표준 MPLS-LSR-STD-MIB을 단순화한 사설 QSS120-MPLS-MIB을 정의하였다. (그림 3)은 사설 QSS120-MPLS-MIB에 정의된 연결 관리 테이블 및 엔트리에 포함된 관리 정보를 나타낸다. qssMPLSInterfaceTable의 인덱스는 qssIfName과 qssInLabel를 사용한다. qssIfName과 qssNextHopIpAddr은 MPLS-LSR-STD-MIB의 입력 및 출력 세그먼트 테이블의 입력 및 출력 세그먼트 인터페이스와 동일한 의미를 가진다.

(그림 4)는 QSS120-MPLS-MIB을 사용하여 망 관리 매니저가 탑재된 NCP와 QSS 스위치사이에서 연결을 설정하고 해제하는 절차를 나타낸다.

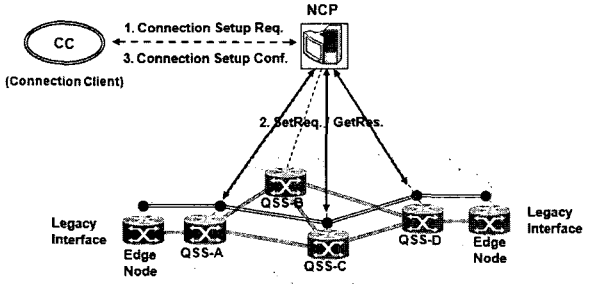
연결 설정을 위하여 표준 MPLS-LSR-STD-MIB을 사용할 경우, 망 관리 매니저는 크로스-커넥트, 입력 및 출력 세그먼트 테이블 엔트리의 설정을 에이전트에게 요구하여야

한다. 하지만, 사설 QSS120-MPLS-MIB을 사용할 경우는 QssMPLSInterfaceEntry의 설정만을 에이전트에게 요구한다. 연결 해제를 위하여 표준 MPLS-LSR-STD-MIB을 사용할 경우, 망 관리 매니저가 크로스-커넥트 테이블 엔트리의 상태 칼럼에'destory'값을 설정하여 에이전트에게 요구하면, 에이전트는 내부적으로 크로스-커넥트 테이블 엔트리와 입력 및 출력 세그먼트 테이블의 엔트리를 삭제하게 된다. 반면, 사설 QSS120-MPLS-MIB을 사용하여 연결 해제를 할 경우에는 QssMPLSInterfaceEntry의 상태 칼럼에 'destroy'값을 설정하여 에이전트에게 요구하며, 에이전트는 해제를 요청받은 해당 QssMPLSInterfaceEntry를 삭제한다.

3.2 단일 단계 기반의 병렬형 연결 관리 메커니즘

중앙 집중화된 NCP는 QSS 스위치들의 자원 상태와 토폴로지를 관리하므로 트래픽의 흐름에 기반하여 서비스 품질이 보장되는 통신 서비스를 효과적으로 제공할 수 있다. (그림 5)와 같은 중앙 집중형 망에서는 전통적인 스위치-바이-스위치 기반의 연결 관리에 비하여 NCP와 QSS 스위치들 사이의 관리 및 제어 메시지의 교환으로 인한 처리 지연의 오버헤드가 발생한다. 중앙 집중형 망의 핵심 기술 중의 하나는 NCP가 입구에서 출구 에지 노드까지의 경로상에 있는 모든 스위치들에게 가능한 신속하게 연결 설정을 완료하는 것이다. (그림 5)는 중앙 집중화된 NCP와 QSS들 사이에서 적용한 SNMP 인터페이스에서 연결 관리의 고속화를 위하여 본 논문에서 채택한 단일 단계 기반의 병렬형 연결 관리에 대한 전체적인 메커니즘을 나타낸다.

중앙 집중화된 NCP가 전달 망 내의 모든 QSS 스위치에 대한 토폴로지 및 자원 상태를 관리하고 있으므로, 경로 계산 및 연결 수락 제어가 NCP에서 수행된다. 이는 NCP가 다수의 QSS 스위치에 대하여 자원 예약과 설정 단계의 구분없이 단일 단계의 연결 설정에 기반한 병렬형 연결 관리를 가능하게 한다. NCP가 발신측과 수신측 주소, QoS를 포함한 연결의 생성 요구 메시지를 CC(Connection Client)로부터 수신하면, 경로 계산 및 연결 수락 제어를 수행한다. 경로 계산 및 연결 수락 제어에 문제가 없다면, NCP는 계산된 경로 및 연결 관리 정보들을 기반으로 경로상의 스위치들에게 SNMP의 SetRequest 메시지를 동시에 보낸다. QSS 스위치는 요구받은 연결을 설정한 후 GetResponse 메시지로 NCP에게 응답한다. 병렬형 연결 관리를 위한 NCP



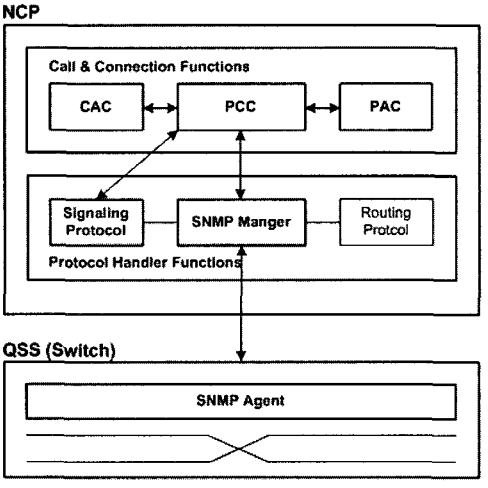
(그림 5) 중앙 집중형 망에서의 병렬형 연결 관리 메커니즘

의 기능 구조와 내부 메시지, 자료 구조와 상세 절차는 다음 절들에서 기술한다.

3.3 병렬형 연결 관리를 위한 NCP의 기능 구조 및 내부 메시지

(그림 6)은 병렬형 연결 관리를 위한 NCP의 기능 구조를 나타낸다. 병렬형 연결 관리를 위하여 NCP는 연결 관리를 담당하는 CCF(Call & Connection Functions), SNMP 매니저 그리고 신호 및 라우팅 프로토콜이 탑재되는 PHF(Protocol Handler Functions)로 구성된다. 연결 관리를 담당하는 CCF에는 CAC(Connection Admission Control), PAC(PAth Computation) 그리고 PCC(Parallel Connection Control)가 동작한다. CAC는 자원 상태를 이용하여 연결 수락 및 자원에 대한 제어를 담당하며, PAC는 연결 생성 요구 메시지에 포함된 발신지 및 착신지 주소와 QoS 파라미터에 기반하여 전달 망에 대한 최적의 경로 계산을 담당한다. 외부의 응용 서버 및 망 관리 매니저나 신호 프로토콜로부터 연결의 생성 요구 메시지를 수신한 PCC는 SNMP 매니저와 연동하여 PAC에서 계산된 경로상의 QSS 스위치들에게 병렬로 연결을 설정하거나 해제를 수행하는 병렬형 연결 관리 기능을 담당한다.

NCP의 CCF에서 동작하는 PCC는 PAC 및 CAC와의 상호 작용을 통해 병렬형 연결 관리를 수행한다. <표 1>은 상호 작용을 위하여 교환되는 내부 메시지를 나타낸다. 각 메시지는 메시지의 타입으로 구분되며 성공 및 실패에 대한 응답으로 Ack 또는 Nack 메시지를 사용한다.



CAC : Connection Admission Control, PCC : Parallel Connection Control, PAC : PAth Computation

(그림 6) 병렬형 연결 관리를 위한 NCP의 기능 구조도

〈표 1〉 NCP의 내부 메시지와 인터페이스

내부 메시지	인터페이스	용도
Request Path	PCC ↔ PAC	경로 계산 요청
Recalculate Path	PCC ↔ PAC	경로 재계산 요청
Reserve Resource	PCC ↔ CAC	자원 수락 요청 및 예약
Assign Resource	PCC ↔ CAC	자원 할당의 완료 요청
Release Resource	PCC ↔ CAC	자원 해제의 완료 요청

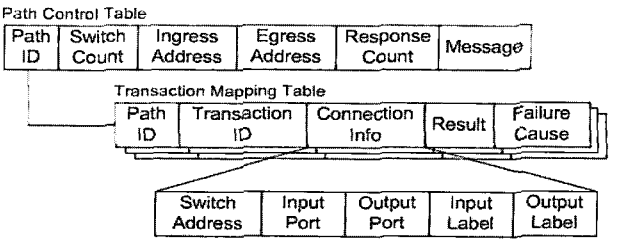
Request Path 메시지는 연결의 생성 요구 메시지를 수신한 PCC가 발신지 및 착신지 주소와 QoS 정보를 제공하여 PAC에게 경로 계산을 요청하는 메시지이다. Recalculate Path 메시지는 PCC가 PAC에게 경로의 재계산을 요청하는 메시지이다. 스위치의 내부 오류 또는 프로토콜의 오류로 인하여 연결 설정이 실패하거나 우회 경로로 연결을 재설정하는 경우 그리고 계산된 경로상의 스위치에 대한 자원 예약이 실패한 경우에 경로의 재계산 요청이 사용된다. Reserve Resource 메시지는 PCC가 경로상의 스위치들에 대한 주소 정보와 QoS 정보를 제공하여 CAC에게 연결 수락 제어 및 자원 예약을 요청하는 메시지이다. CAC는 각 스위치에 대한 연결 수락 여부와 경로상의 스위치들에 대한 입력 및 출력 포트와 레이블 정보를 PCC에게 반환한다. PCC는 성공적인 Reserve Resource 메시지를 CAC에게서 수신한 후 연결 설정을 요구하는 SNMP 메시지들을 QSS 스위치들에게 동시에 전달한다. 경로상의 모든 QSS 스위치로부터 연결 설정의 완료를 SNMP 메시지로 통보받으면 PCC는 CAC에게 Assign Resource 메시지를 전달하여 스위치들의 자원이 할당되었음을 알린다. Release Resource 메시지는 PCC가 CAC에게 스위치에 예약되었거나 할당되었던 자원의 해제를 요구하는 메시지이다.

3.4 병렬형 연결 관리를 위한 자료 구조

NCP의 PCC는 SNMP 매니저와 상호 동작하여 경로상에 있는 QSS 스위치들에게 연결 설정을 위한 SetRequest 메시지를 병렬로 보내게 된다. 병렬형 연결 관리를 위하여 NCP는 수신한 GetResponse 메시지가 어느 QSS 스위치로부터, 어떤 SetRequest 메시지에 대한 응답인지를 구분할 수 있어야 한다. 또한, NCP는 경로에 있는 모든 스위치들에 대한 연결 설정이 모두 완료되었음을 판단하기 위해 경로상의 스위치 개수와 스위치들로부터 수신한 응답의 개수가 일치하는지 확인할 수 있어야 한다. (그림 7)은 병렬형 연결 관리를 위하여 PCC에서 유지하는 경로 제어 테이블(PCT: Path Control Table)과 트랜잭션 매핑 테이블(TMT: Transaction Mapping Table)의 자료 구조를 나타낸다.

PCT 테이블의 각 엔트리에는 경로 식별자(PathID), 입구 및 출구 주소(Ingress/Egress Address), 경로상에 선택된 스위치의 개수(Swtich Count), 연결 설정 요청에 응답한 스위치의 개수(Response Count) 그리고 메시지(Message) 필드로 구성된다. 메시지 필드에는 PCC가 수신한 연결의 생성 요구 메시지의 원본이 저장되며, 경로의 재계산이나 연결의 재설정 시에 사용된다. PCT 테이블의 엔트리 하나에 대하여 해당 경로상에 선택된 스위치의 개수만큼 TMT 테이블의 엔트리들이 생성되며 경로 식별자 필드는 동일한 값을 갖는다.

TMT의 트랜잭션 식별자(TransactionID)는 SNMP 매니저와 에이전트 사이의 메시지를 식별하기 위한 필드로 SNMP 메시지의 RequestID와 매핑된다. 연결 설정과 해제 시에 사용되는 TMT의 연결 정보(Connection Info)필드는



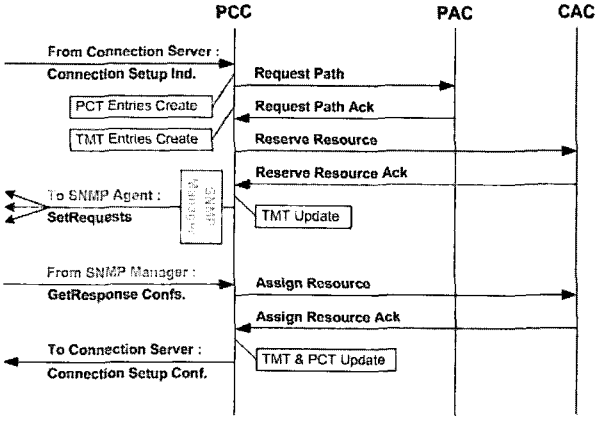
(그림 7) 병렬형 연결 관리를 위한 테이블

경로상의 특정 스위치에 대한 연결 정보를 나타내며, 스위치 주소, 입력 및 출력 포트와 레이블 정보를 갖는다. TMT의 결과(Result) 필드는 연결 설정 요청에 대한 결과를 나타내며, 실패 원인(Failure Cause) 필드는 스위치로부터 실패 응답을 수신한 경우에 대한 실패 원인을 나타낸다. PCC는 SNMP 매니저로부터 받은 응답 메시지가 어느 스위치, 어느 경로에 대한 응답인지를 구분하기 위하여 TMT의 TransactionID와 PathID 필드를 사용하며, PCT와 TMT의 PathID 필드를 매핑시킴으로써 내부 경로에 대한 연결 설정이 모두 완료되었는지 판단하게 된다.

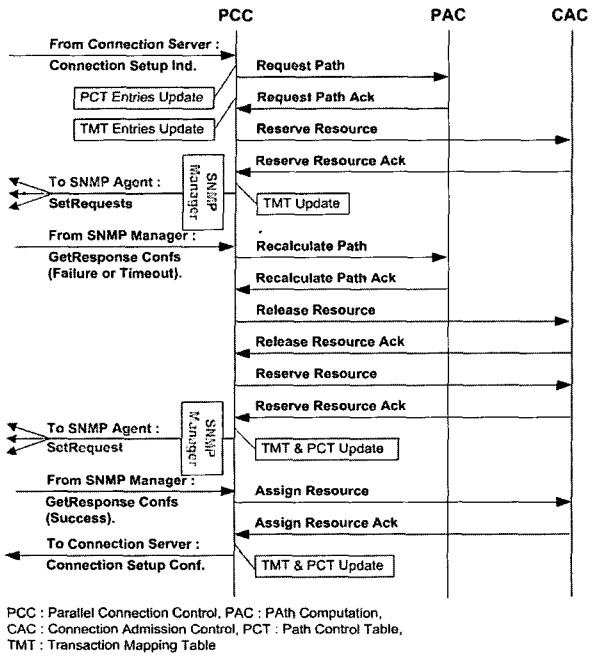
3.5 병렬형 연결 설정 및 연결 설정 실패 시 재설정 절차

3.5.1 병렬형 연결 설정 절차

(그림 8)은 NCP가 QSS 스위치들에게 성공적으로 병렬형 연결 설정을 수행한 경우의 절차를 나타낸다. PCC가 연결 생성 요구(Connection Setup Ind.)를 수신하면, PCT 테이블의 엔트리를 생성한 후, Request Path 메시지를 PAC에게 전달한다. PAC는 전달 망에 대하여 계산된 최적의 경로를 Ack 메시지에 포함하여 PCC에게 응답한다. PCC는 수신한 Request Path Ack 메시지의 정보를 기반으로 TMT의 엔트리들을 생성한 후, CAC에게 Reserve Resource 메시지를 전달한다. CAC는 연결 수락과 자원 예약을 수행한 후, 응답으로 Ack 메시지를 PCC에게 전달한다. PCC는 SNMP 매니저와 상호 작동하여 TMT 엔트리들에 기록된 연결 정보를 기반으로



(그림 8) 병렬형 연결 설정 절차



(그림 9) 연결 설정 실패 시의 재설정 절차

경로상에 선택된 스위치들에게 SetRequest 메시지를 병렬로 전송한다. 경로상의 각 스위치들로부터 연결 설정의 성공 응답을 모두 수신하면, PCC는 CAC에게 Assign Resource 메시지를 보낸다. CAC는 응답으로 Ack 메시지를 PCC에게 전달하며, PCC는 연결 설정을 요구한 응용에게 연결의 설정 결과(Connection Setup Conf.)를 전달한다.

3.5.2 연결 설정 실패시의 재설정 절차

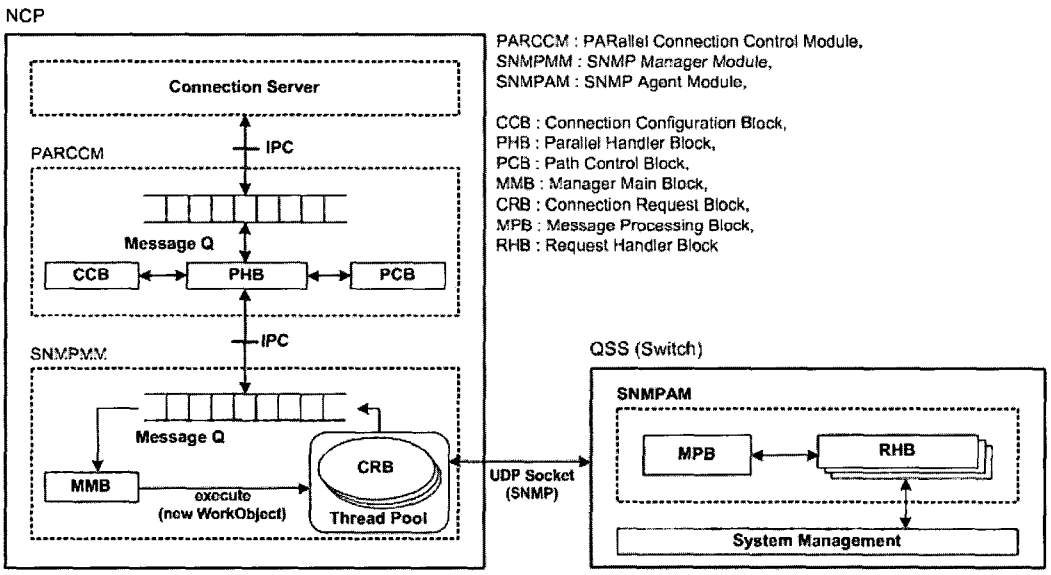
(그림 9)는 스위치 내부 혹은 프로토콜 오류로 인하여 연결 설정이 실패한 경우에 연결을 재설정하는 절차이다. SetRequest 메시지에 대하여 실패 결과를 포함한 GetResponse 메시지를 수신하면, SNMP 매니저는 PCC에게 연결 설정의 실패(GetResponse Conf.)를 전달한다. PCC는 PAC에게 실패한 스위치의 주소를 통보하여 실패한 스위치를 제외한 새로운 경로를 계산하도록 Recalculate Path 메시지를 전달한다. 재계산된 경로와 이전 경로를 비교하여 경로에서 제외된 스위치에 대해서는 자원을 해제하며, 새로 추가된 스위치들에 대해서는 자원을 예약하도록 CAC에게 Reserve 메시지를 전달한다. CAC는 새로 추가된 스위치들의 자원 예약에 대한 Ack 메시지를 PCC에게 전달하게 되며, 이후의 절차는 (그림 8)의 연결 설정 절차와 동일하다.

4. NCP와 QSS 연결 관리 메커니즘의 설계 및 구현

본 장에서는 SNMP 관리를 사용하는 NCP와 QSS의 연결 관리 메커니즘에 대한 설계 및 구현에 대하여 기술한다. 고속의 연결 관리를 위하여 NCP와 QSS 사이에는 병렬형 기법을 도입하며, NCP에는 쓰레드 및 객체 풀의 개념을 도입하였다.

4.1 연결 관리 테스트-베드의 구현 구조

(그림 10)은 NCP와 QSS 연결 관리 테스트-베드의 소프트웨어 구현 구조를 나타낸다. NCP에는 연결 서버(Connection Server), 병렬형 연결 관리를 수행하는 PARCCM(PARallel Connection Control Module) 그리고 SNMP 매니저 기능을 수행하는 SNMPMM(SNMP Manager Module) 모듈이 탑재된다. QSS 스위치에는 SNMP 에이전트 기능을 수행하는 SNMPAM(SNMP Agent Module) 모듈이 탑재된다. NCP의 연결 서버와 PARCCM 모듈 그리고 PARCCM과 SNMPMM 모듈은 상호 통신을 위하여 메시지-큐를 사용한다. NCP의 PARCCM과 QSS의 SNMPAM 모듈은 리눅스 환경에서 C로



(그림 10) NCP와 QSS의 소프트웨어 구현 구조

구현하였다. NCP의 SNMPMM 모듈은 AdventNet사의 Web NMS[17]에 포함된 SNMP API 중, 상위 레벨 API인 SnmpTarget를 사용하여 Java로 구현하였으며, QSS의 SNMPAM 모듈은 AdventNet사의 Agent Toolkit[17]을 이용하여 구현하였다. C로 구현된 PARCCM 모듈과 Java로 구현된 SNMPMM 모듈과는 메시지-큐 통신을 위하여 JNI (Java Native Interface)를 사용하였다.

연결 서버는 클라이언트로부터 연결의 생성 메시지를 수신하여 PARCCM 모듈에게 병렬형 연결 관리에 대한 처리를 요청한다. PARCCM 모듈은 CCB(Connection Configuration Block), PHB(Parallel Handler Block) 그리고 PCB(Path Control Block) 블록으로 구성된다. CCB 블록은 자원 관리 및 연결 설정 요청에 대한 연결 수락 기능을 담당하며, PCB 블록은 토폴로지 관리 및 경로 계산을 수행한다. PHB 블록은 다수의 QSS 스위치에게 SNMP의 SetRequest 메시지를 병렬로 보내기 위해 SNMPMM 모듈과 상호 동작하여 병렬형 연결 관리를 수행하게 된다. PHB 블록이 SNMPMM 모듈로부터 SetRequest 메시지에 대한 응답으로 GetResponse 메시지의 결과를 모두 수신하면, 연결 서버에게 병렬형 연결 관리에 대한 결과를 알려준다.

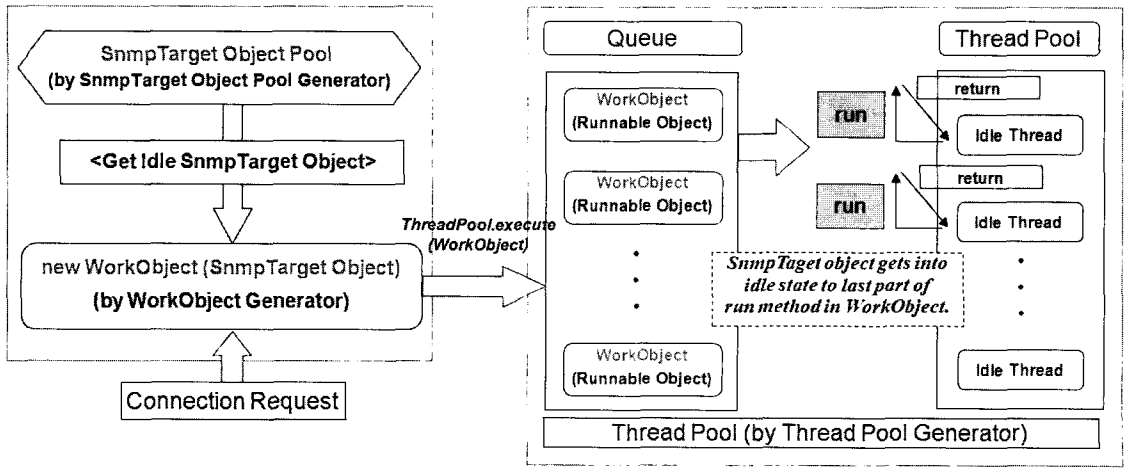
4.2 쓰레드 및 객체 풀

SNMPMM 모듈은 (그림 10)과 같이 MMB(Manager Main Block) 블록과 쓰레드 풀 기반에서 쓰레드로 동작하는 CRB(Connection Request Block) 블록들로 구성된다. SNMPMM 모듈이 실행되면 MMB 블록의 SnmpTarget 객체 풀 생성기(SnmpTarget Object Pool Generator)에 의해서 SnmpTarget 객체 풀이 생성되며, 쓰레드 풀 생성기(Thread Pool Generator)에 의해서 쓰레드 풀 객체가 생성된다. 쓰레드 풀은 Executors 클래스를 이용하여 구현하였다. Executors 클래스는 JDK(Java Development Kit) 1.5.0 이상 버전에서는 java.util.concurrent 패키지에 포함되어 있으며, JDK 1.4.2 버전에서는 backport-util-concurrent 패키지[18]를 사용하여 구현할 수 있다. SNMP 모듈에서 사용하는 SNMP API인 SnmpTarget 객체는 MIB 로딩 및 기본 파라미터 값의 설정으로 인하여 객체의 생성에 많은 시간이 소요된다. SnmpTarget 객체의 생성 시간에 대한 오버헤더를 개선하고자 SnmpTarget 객체 풀을 사용하였다. (그림 11)은 쓰레드 풀과 SnmpTarget 객체 풀의 연관 관계를 나타낸 것이다.

MMB 블록이 PARCCM 모듈로부터 연결 설정의 요청을 수신하면, WorkObject 생성기가 연결 설정에 대한 태스크를 포함하여 Runnable 인터페이스로 구현한 WorkObject 객체를 생성한다. 이 때, SnmpTarget 객체 풀에서 유휴(idle) 상태의 SnmpTarget 객체를 인자로 사용한다. 다음 단계로 WorkObject 생성기는 생성된 WorkObject 객체를 인자로하여 쓰레드 풀 객체의 execute 메소드를 호출한다. execute 메소드가 호출되면, WorkObject 객체가 큐에 인큐(enQueue) 및 디큐(deQueue)되면서 쓰레드 풀에서 유휴 상태에 있던 쓰레드에 WorkObject 객체가 할당되어 CRB 쓰레드로 활성화된다. CRB 쓰레드는 SNMP의 SetRequest 메시지를 QSS 스위치에 탑재된 SNMPAM 모듈에게 보내는 역할과 함께 SNMPAM 모듈로부터 응답 받은 결과를 JNI를 통해 PARCCM 모듈에게 알려주는 역할을 수행한다. CRB 쓰레드로 활성화된 WorkObject의 run 메소드 마지막 부분에서 연결 설정의 응답이 수신되면 SnmpTarget 객체는 유휴 상태로 변경되어 SnmpTarget 객체 풀에 반환되며, run 메소드가 종료되면 쓰레드도 쓰레드 풀에 반환되어진다.

5. 연결 설정 지연 및 완료율

본 장에서는 중앙 집중형 NCP와 QSS 스위치들로 구성되는 실험 환경을 기술하며 본 논문에서 구현한 고속의 SNMP 인터페이스에서 연결 관리 방식에 대한 연결 설정 지연과 완료율을 측정하여 비교 및 분석한다.



(그림 11) 쓰레드 및 객체 풀의 연관 관계

5.1 실험 환경 및 연결 관리 테스트-베드

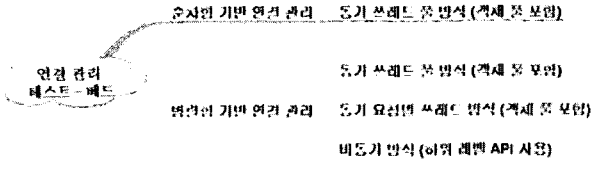
(그림 5)와 같은 네트워크 환경에서 QSS 스위치의 개수가 3대인 경우와 5대인 경우에 대하여 연결 성능을 측정하기 위한 실험 환경을 구축하였다. NCP의 연결 관리 기능은 3.20GHz Xeon CPU에 메모리 1GB를 장착한 서버급 PC에 탑재하였으며, QSS 스위치의 연결 관리 기능은 2.80GHz 펜티엄-4 CPU에 메모리 1GB를 장착한 일반 PC에 탑재하였다. 연결 요청을 생성하는 클라이언트인 CC(Connection Client)는 1.70GHz 펜티엄-4 CPU에 메모리 256MB를 장착한 일반 PC를 사용하였다. NCP의 운영체제는 커널 2.6 기반의 리눅스를 사용하였으며, QSS 스위치들과 CC 호스트의 운영체제는 커널 2.4 기반의 리눅스를 사용하였다. QSS 스위치에는 SNMP 에이전트의 기능을 수행하는 SNMPAM 모듈이 탑재되며, 평균 2ms의 지수분포를 갖는 지연을 두어 디바이스 드라이버를 통하여 연결을 설정하는 인위적인 프로세싱 작업을 하도록 하였다.

본 논문에서는 연결 설정 지연과 완료율에 대한 성능의 비교 분석을 위하여 병렬형과 순차형 연결 관리 방식을 각각 구현하였다. NCP의 매니저 기능을 수행하는 SNMPMM 모듈은 동기 기반의 쓰레드 풀 방식과 동기 요청별 쓰레드 방식 그리고 비동기 방식으로 구분하여 (그림 12)와 같이 크게 4가지의 연결 관리 테스트-베드를 구축하였다. 일반적으로 SNMP 관리 방식에서는 동기 방식이 많이 사용된다. 동기 방식은 SNMP 매니저가 요청을 보낸 후 에이전트로부터 응답을 수신할 때까지 대기하는 방식이다. 동기 방식에서 동시에 여러 연결을 설정하기 위해서는 쓰레드 기반의 구현이 요구된다. 비동기 방식은 SNMP 매니저가 요청을 보낸 후 새로운 요청을 에이전트에게 전달할 수 있는 방식이다. 비동기 방식은 AdventNet사의 Web NMS[17]에 포함된 SNMP API 중 하위 레벨 API인 SnmpClient 등을 사용하여 콜백(call back) 방식으로 구현하였다.

순차형 연결 관리 방식을 위하여 NCP가 담당하는 PARCCM 모듈과 대응되는 SEQCCM(SEQuential Connection Control Module) 모듈을 구현하였다. SEQCCM 모듈은 순차형 연결 관리를 담당하며, CCB와 PCB 그리고 SCB(Sequential Connection Block) 블록으로 구성된다. CCB 및 PCB 블록은 PARCCM 모듈의 CCB와 PCB 블록과 동일하다. SCB 블록은 연결 서버로부터 연결의 생성 요구 메시지를 수신하면 NCP가 경로상의 QSS 스위치들에게 차례대로 연결을 설정하는 기능을 수행한다.

5.2 연결 성능 측정 및 결과 분석

실험 환경에서 구축한 연결 관리 테스트-베드에 대하여 포아송 분포를 갖는 초당 연결의 평균 도착 개수가 20에서 300까지 증가하는 경우에 대하여 연결 설정 지연과 완료율을 측정하였다. 본 논문에서 초당 연결 요청의 수를 300까지 측정한 이유는 4ESS 스위치의 BHCA(Busy-Hour Call Attempts)가 500,000이기 때문이다[12]. 성능 측정을 위하여 10,000개의 메시지를 사용하였으며, 쓰레드 풀의 크기는 15개로 설정하였다. SnmpTarget 객체의 풀 크기는 300개로 설정하였으며, QSS 스위치의 수는 3대인 경우와 5대인 경우로 구분하였다. 연결 설정 지연은 연결 서버에서 측정하며, CC로부터 연결 설정의 요청을 받은 시점에서 PARCCM 모듈로부터 연결 설정의 완료를 통보 받은 시점까지로 규정하였다.

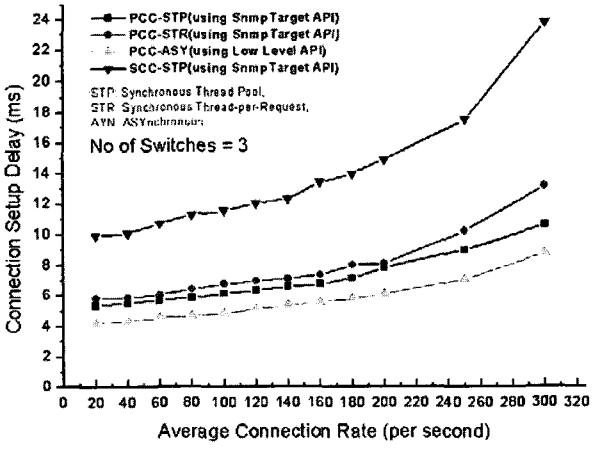


(그림 12) 실험실 환경의 연결 관리 테스트-베드

5.2.1 연결 설정 지연

(그림 13)은 실험실 환경에서 QSS 스위치의 수가 3대인 경우에 대하여 (그림 12)의 방식들에 대한 연결 성능 지연을 측정하여 그래프로 나타낸 것이다. 연결 요청의 수가 100인 경우를 기준으로 병렬형 동기 쓰레드 풀 방식(PCC-STP)과 각 방식을 비교해 보면, 병렬형 동기 쓰레드 풀 방식의 연결 설정 지연이 병렬형 동기 요청별 쓰레드 방식(PCC-STR)보다 약 1.11배 감소되었으며, 순차형 동기 쓰레드 풀 방식(SCC-STP)보다는 약 1.89배 감소되었다. 하지만, 쓰레드 풀을 이용하는 병렬형 동기 방식(PCC-STP)은 병렬형 비동기 방식(PCC-ASY)보다는 약 1.27배가 증가되었다. Web NMS의 상위 레벨 API인 SnmpTarget API를 사용하여 동기 방식으로 동작하는 연결 관리에서는 병렬형 쓰레드 풀 방식이 가장 우수하지만, 전체적으로는 하위 레벨 SNMP API를 사용하는 병렬형 비동기 방식이 가장 우수함을 알 수 있다.

(그림 14)는 실험실 환경에서 QSS 스위치의 수가 5대인 경우에 대하여 (그림 12)의 방식들에 대한 연결 설정 지연을 측정하여 그래프로 나타낸 것이다. 연결 요청의 수가 100인 경우를 기준으로 병렬형 동기 쓰레드 풀 방식(PCC-STP)과 각 방식을 비교해 보면, 병렬형 동기 쓰레드 풀 방식의 연결



(그림 13) 스위치의 수가 3대인 경우의 연결 설정 지연

설정 지연이 병렬형 동기 요청별 쓰레드 방식(PCC-STR)보다 약 1.2배 감소되었으며, 순차형 동기 쓰레드 풀 방식(SCC-STP)보다는 약 2.8배 감소되었다. 하지만, 쓰레드 풀을 이용하는 병렬형 동기 방식(PCC-STP)은 병렬형 비동기 방식(PCC-ASY)보다는 약 1.27배가 증가되었다. 스위치의 수가 3대인 경우와 동일하게 SnmpTarget API를 사용하여 동기 방식으로 동작하는 연결 관리에서는 병렬형 쓰레드 풀 방식이 가장 우수하지만, 전체적으로는 하위 레벨 SNMP API를 사용하는 병렬형 비동기 방식이 가장 우수함을 알 수 있다.

순차형 쓰레드 풀 방식이 초당 연결의 요청이 300일 때 연결 설정 지연이 급격하게 증가한 이유는 쓰레드 간에 문맥 교환이 빈번히 일어나기 때문이다. 순차형 연결 관리를 담당하는 SEQCCM 모듈은 요청별 쓰레드 방식으로 동작하면서 스위치들에 대하여 순차적으로 연결 설정을 수행한다. 즉, SEQCCM 모듈에서 동작하는 요청별 쓰레드에서 연결 설정을 완료하는 시간보다 더 짧은 주기로 연결 요청이 생성된다. 이는 짧은 시간에 요청별 쓰레드의 수가 많아지므로 쓰레드 간의 문맥 교환이 빈번히 이루어지게 된다.

병렬형 동기 요청별 쓰레드 방식에서 연결 요청의 수가 200부터 연결 설정 지연이 증가하는 이유 또한 쓰레드 간의 문맥 교환에 의하여 발생하는 오버헤더가 원인인 것으로 분석된다. 병렬형 연결 관리 방식을 사용함으로 순차형 연결 관리 방식보다는 연결을 설정하는 시간이 많이 감소하여 급격히 쓰레드의 수가 증가되지는 않았지만, 200 아래의 경우보다는 다소 많은 쓰레드가 생성되었기 때문이다.

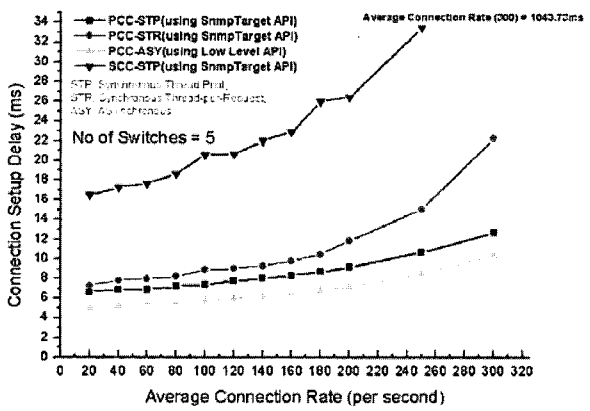
연결 설정 지연을 비교 및 분석한 결과, 순차형 쓰레드 풀 방식은 스위치가 증가하고 초당 연결 요청의 수가 증가하면 할수록 연결 설정 지연도 급격히 증가하게 된다. 반면, 병렬형 방식들은 다수의 QSS 스위치에게 동시에 연결 설정을 수행함으로 연결 설정 지연을 최소화 할 수 있다. 하지만, 동기 방식을 사용하는 연결 관리에서 병렬형 방식을 사용하더라도 쓰레드 풀을 도입하지 않은 병렬형 동기 요청별 쓰레드 방식은 스위치 및 초당 연결 요청의 수가 증가하면 할수록 짧은 시간에 요청별 쓰레드의 수가 많아지므로 쓰레드 풀을 도입한 병렬형 쓰레드 방식보다는 연결 설정 지연이 증가하게 된다.

5.2.2 연결 설정 완료율

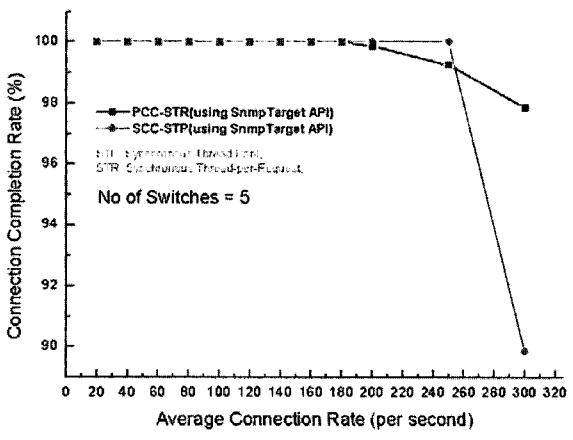
실험 환경에서 QSS 스위치의 수가 3대인 경우에 연결 설정의 완료율은 (그림 14)의 방식들에서 모두 동일하게 100%로 측정되었다. (그림 15)는 QSS 스위치의 수가 5대인 경우에 대하여 측정한 연결 설정의 완료율을 그래프로 나타낸 것이다. 완료율이 100%인 방식은 그래프에서 나타내지 않았다. 병렬형 동기 요청별 쓰레드 방식(PCC-STR)과 순차형 동기 쓰레드 풀 방식(SCC-STP)은 초당 연결의 평균 도착 개수에 따라 다소 감소하였다. 초당 연결의 도착 개수와 스위치의 수에 따른 연결 설정 지연과 더불어 쓰레드 및 객체 풀과 메시지-큐의 크기는 연결 설정의 완료율과 밀접한 관계를 가진다. 내부 통신을 위하여 메시지-큐를 사용하는 구현 구조에서 연결 요청의 평균 도착 시간보다 연결 설정 지연이 긴 경우에 메시지-큐의 크기는 완료율에 영향을 미치게 된다. 실험 환경에서 메시지-큐의 크기는 시스템의 기본 값(약 16KB)을 사용하였으며, 메시지-큐 크기의 85%를 가용범위의 임계치로 설정하였다. 병렬형 동기 요청별 쓰레드와 순차형 동기 쓰레드 풀 방식에서 메시지-큐 크기를 증가하면 완료율을 개선할 수 있음을 실험 과정에서 확인할 수 있다.

5.2.3 객체 풀의 성능 개선 효과

NCP의 SNMPMM 모듈은 AdventNet사의 Web NMS에서 제공하는 SNMP API인 SnmpTarget 객체를 사용한다. 매번 연결 설정 시에 생성되는 SnmpTarget 객체의 생성 오버헤더를 개선하기 위하여 객체 풀의 개념을 사용하였으며, 객체 풀의 사용과 미사용 시의 성능을 측정하여 비교하였다. <표 2>는 병렬형 쓰레드 풀 방식(PCC-STP)에서 SnmpTarget 객체 풀을 사용하는 경우와 사용하지 않은 경우의 연결 설정 지연과 완료율을 측정한 결과이다. 객체 풀을 사용하지 않은 경우의 연결 설정 지연은 20에서 300까지의 연결 요청에서 70.32와 380.95ms 사이의 값이 측정되었다. SnmpTarget



(그림 14) 스위치의 수가 5대인 경우의 연결 설정 지연



(그림 15) 완료율

〈표 2〉 객체 풀의 사용 여부에 따른 연결 설정 지연과 완료율

(단위 : ms, %)

초당 연결 요청	객체 풀 미사용		객체 풀 사용	
	연결 설정 지연	연결설정 완료율	연결 설정 지연	연결설정 완료율
20	70.32	100.00	6.61	100.00
40	269.41	87.37	6.85	100.00
60	337.01	59.09	6.89	100.00
80	348.25	44.76	7.22	100.00
100	256.09	36.03	7.36	100.00
120	358.59	30.43	7.77	100.00
140	363.08	26.25	8.02	100.00
160	365.69	21.17	8.27	100.00
180	368.32	20.77	8.69	100.00
200	371.61	18.84	9.15	100.00
250	379.81	15.28	10.68	100.00
300	380.95	13.11	12.66	100.00

객체 풀 크기를 300개로 설정한 경우에는 연결 설정 지연이 20에서 300까지의 연결 요청에서 6.61과 12.66ms 사이의 값이 측정되었다. 연결 요청의 수가 100인 경우에 객체 풀의 사용과 미사용에 따른 성능 차이는 약 34.8배 정도임을 알 수 있다. 이는 연결 요청이 생성되는 시간보다 (그림 10)의 WorkObject 객체내에서 SnmpTarget 객체를 매번 생성해야 하는 오버헤드가 발생하기 때문으로 분석된다. 완료율 또한 급격히 저하됨을 확인 할 수 있다. 결과적으로 동기 방식으로 동작하는 연결 관리 테스트-베드에서 SnmpTarget 객체 풀의 사용은 연결 설정 지연과 완료율을 개선하는데 있어서 중요한 성능 파라미터임을 알 수 있다.

6. 결 론

본 논문에서는 중앙 집중화된 네트워크 제어 플랫폼인 NCP와 흐름기반의 스위칭을 지원하는 QSS 스위치 사이에 사용되고 있는 SNMP 인터페이스에서 고속의 연결 관리를 위하여 병렬형 기법과 쓰레드 및 객체 풀을 도입하였다. 또한, 병렬형 연결 관리를 위하여 NCP의 모듈 구조와 자료구조 그리고 내부 메시지를 정의하였다. 연결 설정 지연과 완료율에 대한 성능 측정과 분석을 위하여 NCP의 연결 관리 모듈을 병렬형과 순차형 방식으로 구분하여 구현하였으며, NCP의 매니저 모듈은 동기 기반의 쓰레드 풀과 요청별 쓰레드 방식 그리고 비동기 방식으로 구분하여 구현하였다.

측정 결과 스위치의 수가 5개인 경우에, 쓰레드 및 객체 풀을 사용하는 동기 방식에서 초당 연결 요청의 수가 100인 경우에 병렬형 방식이 순차형 방식보다 연결 설정 지연이 약 2.8배 개선되었으며, 병렬형 연결 관리 방식에서 쓰레드 및 객체 풀을 사용한 방식이 요청별 쓰레드 방식보다 연결 설정 지연이 약 1.2배 개선되었음을 확인할 수 있었다. 연결 설정의 완료율은 쓰레드 및 객체 풀 그리고 병렬형 기법을 도입한 SNMP 연결 관리 방식의 실험에서 100%의 완료율을 보였으나, 요청별 쓰레드 방식과 순차형 연결 관리에서는 초당 연결의 도착 개수가 증가함에 따라 완료율이 감소되었다. NCP의 망 관리 매니저에서 SNMP API인 SnmpTarget 객체 풀을 사용하는 방식이 사용하지 않는 방식에 비하여 연결 설정 지연이 약 34.8배, 완료율은 약 2.7배 이상 개선됨을 확인할 수 있었다. 전체적인 실험 결과, 동기 방식을 사용하는 NCP와 QSS 기반의 SNMP 연결 관리에서 병렬형 방식과 SnmpTarget 객체 풀의 사용이 고속화를 위한 중요한 기술임을 확인하였다.

향후 연구 과제로는 본 논문에서 제시한 병렬형 연결 관리 메커니즘을 SNMP 뿐만 아니라, 추후 NCP에 도입될 수 있는 웹 서비스, COPS, GSMP 등의 다양한 연결 관리에 적용 가능한 개방형의 병렬형 연결 관리 메커니즘을 정의하는 것이다. 또한, 메시지 통합 메커니즘을 본 논문에서 제안한 병렬형 연결 관리 메커니즘과 결합한 고성능의 연결 관리 메커니즘을 연구하는 것이다.

참 고 문 헌

[1] ITU-T, Recommendation Y.2012(formerly Y.NGN-FRA), "Functional requirements and architecture of the NGN of Release 1," September 2006.

[2] 신상철, 이영로, 이승택, 외 13명, "BcN 동향 2004," 한국전산원, 2004년 12월.

[3] 이순석, 김영선, 전경표, "BcN의 핵심 인프라:Flow 기반 QoS 보장 네트워크," Telecommunications Review, 제 15권 제6호, pp.866-877, 2005년 12월.

[4] 이종현, 예병호, "BcN QoS 스위치 및 라우터 개발방향," TTA저널, 제 96호, pp. 69-77, 2004년 12월.

[5] J. Case, M. Fedor, M. Schoffstall and J. Davin, "A Simple Network Management Protocol (SNMP)," IETF RFC 1157, May, 1990.

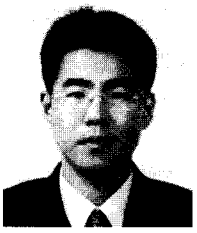
[6] R. Enns, "NETCONF Configuration Protocol," IETF RFC 4741, December 2006.

[7] T. Goddard, "Using NETCONF over the Simple Object Access Protocol (SOAP)," IETF RFC 4743, December 2006

[8] A. Doria, F. Hellstrand, K. Sundell and T. Worster, "General Switch Management Protocol v3," IETF RFC 3292, June, 2002.

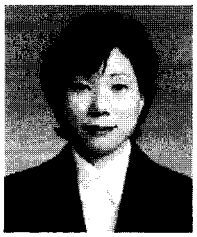
[9] D.P. Xu and B. Bode, "Performance Study and Dynamic Optimization Design for Thread Pool System," Proceedings of the International Conference on Computing, CCCT' 2004, Austin, Texas, August 2004.

[10] M. Veeraraghavan and M. Kshirsagar, "PCC:Parallel Connection Control Algorithm for ATM Networks," Proc. of IEEE ICC 1996, pp.1635-1641, June 1996.
[11] M. Veeraraghaven, G. Choudhury, M. Kshirsagar, "Implementation and Analysis of PCC(Parallel Connection Control)," Proc. of IEEE INFOCOM 1997, pp.832-841, September 1997
[12] Mun Choon Chan and Aurel A. Lazar "Designing a CORBA-Based High Performance Open Programmable Signaling System for ATM Switching Platforms," IEEE Journal on Selected Areas in Communications, vol. 17, no. 9, pp. 1537-1548, September 1999.
[13] IETF, Multiprotocol Label Switching (mpls) Working Group, http://www.ietf.org/html.charters/mpls-charter.html
[14] IETF, Common control and Measurement Plane (ccamp) Working Group, http://www.ietf.org/html.charters/ccamp-charter.html
[15] C. Strinivasn, A.Viswanathan and T. Nadeau, "Multi-protocol Label Switching (MPLS) Label Switch Router (LSR) Management Information Base (MIB)," IETF RFC 3813, June 2004.
[16] 권태현, 임영은, 최인상, 김춘희, 차영욱, "중앙 집중형 서버 기반 망에서의 연결 관리 방식 비교 및 성능 분석," 정보처리학회논문지, 제13-C권 제7호, pp.923-932, 2006년 12월
[17] AdventNet, WebNMS 4 and Agent Toolkit, http://www.adventnet.com
[18] backport-util-concurrent Package, http://dcl.mathcs.emory.edu/util/backport-util-concurrent



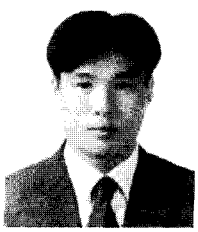
권 태 현

e-mail : thkwon@etri.re.kr
2001년 안동대학교 컴퓨터공학과 (학사)
2003년 안동대학교 컴퓨터공학과 (공학석사)
2006년 8월 안동대학교 컴퓨터공학과 (공학박사)
2003년 3월~2007년 08월 안동대학교 컴퓨터공학과 시간강사
2006년 9월~2007년 08월 안동대학교 연수 연구원
2007년 9월~현재 한국전자통신연구원 광대역통합망연구단 NCP기술팀, 선임연구원
관심분야 : 망 제어 및 관리, BcN/NGN, QoS, 광인터넷, 개방형 통신망 등



김 춘 희

e-mail : chkim@dcu.ac.kr
1988년 전남대학교 전산통계학과 (학사)
1992년 충남대학교 전자계산학과 (이학석사)
2000년 8월 경북대학교 컴퓨터공학과 (공학박사)
1988년~1995년 한국전자통신연구원 연구원
2002년~현재 대구사이버대학교 컴퓨터정보학과 조교수
관심분야 : 고속통신망, 트래픽 제어, 망 관리 등

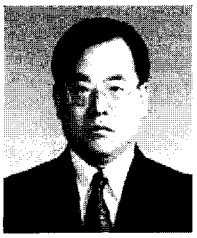


고 영 석

e-mail : zios@comeng.andong.ac.kr
2006년 안동대학교 정보통계학과 (학사)
2006년~현재 안동대학교 컴퓨터공학과 (석사과정)
관심분야 : 망 관리, 웹 서비스, NGN 등

남 현 순

e-mail : namhs@etri.re.kr
1995년 한국외국어대학교 독어교육학과 (학사)
2000년 동국대학교 컴퓨터공학과 (공학석사)
2000년~현재 한국전자통신연구원 광대역통합망연구단 NCP기술팀 선임연구원
관심분야 : 망 제어 및 관리, 분산 네트워크 등



정 유 현

e-mail : yhjeong@etri.re.kr

1980년 광운대학교 전자계산학과 (학사)

1989년 광운대학교 대학원 컴퓨터과학과 (공학석사)

1998년 광운대학교 대학원 컴퓨터공학과 (공학박사)

1980년~현재 한국전자통신연구원 광대역통합망연구단 NCP기술팀 팀장, 책임연구원

관심분야 : 인터넷 QoS, 광 인터넷, 이미지 트랜스코딩 기술, 음성정보처리(음성인식 & 합성)기술 등

차 영 욱

e-mail : ywcha@andong.ac.kr

1987년 경북대학교 전자공학과 (학사)

1992년 충남대학교 계산통계학과 (이학석사)

1998년 경북대학교 컴퓨터공학과 (공학박사)

1987년~1999년 한국전자통신연구원 선임연구원

2003년~2004년 매사추세츠 주립대학 방문학자

1999년~현재 안동대학교 컴퓨터공학과 부교수

관심분야 : 광 인터넷, 개방형 통신망, 망 제어 및 관리, NGN 등