프로그래머블 네트워크 기술 분석

Analysis of Programmable Networks Technology

정영식(Y.S. Chung) 라우터S/W팀 책임연구원
주성순(S.S. Joo) 라우터S/W팀 책임연구원, 팀장

프로그래머블 네트워크 기술은 사용자의 요구에 따라 빠르게 새로운 서비스를 생성, 전개하고자 하는 필요에 따라 발전하였다. 프로그래머블 네트워크 기술은 새로운 구조와 서비스 및 프로토콜들을 네트워크에 적용하기 위해 네트워크의 programmability를 제어하고 안전하게 실행하는 방법이다. 네트워크 programmability 증진을 위하여 전송 하드웨어와 제어 소프트웨어의 분리, 개방형 프로그래머블 네트워크 인터페이스 제공, 네트워킹 기반구조의 가상화 촉진 같은 하드웨어상에 상이한 네트워크 구조의 공존 및 자원분할 기술 등이 연구되고 있다. 이 글에서는 프로그래머블 네트워크 기술 분야의 연구 프로젝트를 살펴보고 프로그래머블 통신 추상화, programmability 수준, 프로그래밍 방법론 등의 특징을 비교 분석하였다.

I. 서 론

새로운 서비스의 등장은 네트워크상에서 서비스를 생성하기 위한 도구와 방법론을 필요로 하고, 서비스 관련 연산과 처리 및 스위칭을 제공하며, 개방적이고 확장 가능한 네트워킹 기반구조(infrastructure)를 제공하는 새로운 네트워크 프로그래밍 환경이 설계되어야 한다.

네트워크를 프로그래머블(programmable)하게 하는 기본요소는 네트워크 기반구조에서 서비스의 생성, 전개 및 관리를 명시적으로 인지할 수 있는 새로운 네트워크 프로그래밍 환경의 개발과 관련이 있다. 새로운 네트워크 프로그래밍 환경에서 공통적으로 제기되고 있는 문제점은 새로운 구조와 서비스 및 프로토콜들을 네트워크에 적용하기 위해 어떻게 네트워크의 programmability를 제어하고 안전하게 실행하는가 하는 것이다.

네트워크를 프로그래머블하게 하는 다른 기본요소는 제어 소프트웨어로부터 통신 하드웨어(스위칭 패브릭, 라우팅 엔진 등)를 분리하는 것이다. 현재는 스위치와 라우터들이 수직적으로 통합되어 있어서 분리를 실현하기 어려운 상태이며, 서비스 제공자 또는 사용자가 스위치/라우터 제어 환경, 알고리즘, 상태 등을 직접 참조(access)할 수 없는 네트워크 노드의 폐쇄성이 새로운 네트워크 서비스의 적용을 불가능하게 한다. 문제는, 제3공급자의 소프트웨어와 서비스를 적용하기 위해 "어떻게 폐쇄성을 개방할 것인가?" 이다.

이 글에서는 프로그래머블 네트워크의 최근 기술 동향을 참고문헌에 기술된 내용을 중심으로 살펴본다[1]. Ⅱ장에서는 개방 시그널링(Open Signalling: Opensig)과 액티브 네트워크(Active Network: AN)로 대표되는 두 커뮤니티의 주장에 대하여 살펴본다. Ⅲ장에서는 프로그래머블 네트워크의 공통된 특성과 관련 프로젝트들을 각 특성별로 분류하여 기술한다. Ⅳ장에서는 Ⅲ장에서 조사한 내용들을 비교 분

석하고, V장에서 결론을 맺는다.

II. 방법론

네트워크상에 새로운 서비스의 추가나 새로운 응용의 필요에 의한 기존 네트워크 서비스의 변경에 대한 요구가 증가하고 있다. 이러한 예는 최근에 IP 네트워크상에 향상된 서비스 품질(Quality of Service: QoS) 보장을 위한 종합/차별화된(integrated/differentiated) 서비스의 등장에서 나타난다. 프로그래머블 네트워킹의 궁극적인 목표는 새로운 네트워크 서비스의 전개를 단순화하는 것이다. 이를 위한 프로그래머블 네트워크의 구조는 재구성이 가능해야 하고, 개방된 프로그래머블 인터페이스와 서비스 구성 방법론 및 도구들의 범위를 명확히 정의하여 이용 가능해야 한다.

네트워크를 프로그래머블하게 하는 방법론은 국제 워크샵을 통해 구성된 Opensig 커뮤니티와 DARPA(Defence Advanced Research Project Agency)에서 추진하는 다양한 액티브 네트워크 프로젝트로 구성된 두 개 커뮤니티의 연구로 구분된다.

1. Opensig 커뮤니티

Opensig 커뮤니티에서는 스위치와 라우터를 참조할 수 있는 개방형 프로그래머블 네트워크 인터페이스를 이용하여 통신 하드웨어를 모델링하며, 이런 방식으로 스위치를 개방하여 새롭고 상이한 구조 및 서비스들을 개발할 수 있다고 주장한다. 개방 시그널링은 통신망(telecommunication)을 기본으로 하는 접근방식으로 네트워크를 프로그래머블하게 만드는 문제를 다루고 있다. 여기서는 프로그래머블 네트워크를 지원하고 품질보장형 서비스 생성에 중점을 두는 제어 및 관리와 전송 사이를 명확히 구분하고 있다. 최근에 IEEE P1520 PIN(Programmable Interfaces for Networks) 프로젝트에서는 Opensig 방식을 사용하여 ATM 스위치, IP 라우터 및 이동통신 네트워크를 위한 프로그래밍 인터페이스를 표준화하는 데 노력하고 있다[2]. 물리적인 네트워크 장비들이 잘 정의된 개방형 프로그래머블 인터페이스를 가진 분산된 컴퓨팅 객체들(예: 가상 스위치, switchlets, virtual base stations 등)로 추상화(abstraction)된다. 개방 인터페이스는 서비스 제공자가 새로운 네트워크 서비스를 구축하고 관리할 수 있도록 CORBA(Common Object Request Broker Architecture)와 같은 미들웨어 도구들을 사용하여 네트워크의 상태를 다룰 수 있게 한다.

2. 액티브 네트워크 커뮤니티

액티브 네트워크 커뮤니티에서는 주로 기존 IP 네트워크 영역 내에서 실행시간에 새로운 서비스를 동적으로 전개하는 방식을 주장하고 있다. 새로운 서비스를 위한 동적 실행시간 지원의 수준은(액티브 패킷이라는 패킷을 가져와서 실행하고, 다시 포워딩하는 것을 고려할 때) Opensig 커뮤니티에서 제안된 것의 범위를 훨씬 뛰어넘는 것이다. 액티브 네트워킹에서 하나의 예는 코드와 데이터로 구성된 실행 가능한 프로그램을 포함하는 "캡슐"이라는 것이며, 코드의 이동성은 프로그램을 전달하고 제어 및 서비스를 구축하기 위한 매체로 표현된다. 제어의 정도(granularity of control)는 수신된 패킷에 의해 수정될 수 있는 스위치/라우터 동작의 범위를 말한다. 극단적으로, 하나의 패킷(캡슐)이 그 노드에 도착하는 모든 패킷에서 보여지는 모든 소프트웨어 환경을 부트 할 수 있으며, 때로는 그 패킷에서만 보여지는 동작을 변경할 수 있다. 액티브 네트워크에서는 네트워크 서비스의 구성을 가능하게 하며, 서비스 생성을 지원하는 데 최대 유연성(flexibility)을 제공한다[3]. 액티브 네트워크 방식은 Opensig의 준-정적인 네트워크 프로그래밍 인터페이스보다 동적으로 작동한다.

두 커뮤니티는 통신 네트워크에 새로운 서비스를 구축하고, 전개 및 관리하기 위한 기존의 방식이나 기술을 뛰어넘는 공통의 목표를 가지고 있으며, 다양한 기술적인 방식을 가진 여러 가지 프로젝트들이

수행되고 있다. Opensig 방식에서는 정보 전달로부터 네트워크 제어를 확실하게 분리하며, 일정 수준의 QoS를 지원하는 프로그래머블 스위치에 초점을 맞추고 있는 반면, 액티브 네트워크 방식에서는 제어 및 데이터를 위한 경로가 결합되어 있는 전형적인 IP 네트워크에 초점을 맞추고 있다.

III. 프로그래머블 네트워크

현재 프로그래머블 네트워킹 분야에서 수행되고 있는 많은 프로젝트들에 대하여 특징별로 각 프로젝트가 가지고 있는 고유의 기능에 대하여 알아본다.

다수의 연구 그룹에서 수행하고 있는 프로젝트들의 내용을 살펴보면 프로그래머블 네트워크를 구축하는 데 사용되는 공통된 특징들을 추출할 수 있으며, 이런 특징들은 다음과 같이 분류할 수 있다.

- 네트워킹 기술(networking technology)
- programmability의 수준(level of programmability)
- 프로그래머블 통신 추상화(programmable communications abstractions)
- 구조적 영역(architectural domain)

1. 네트워킹 기술

네트워킹 기술은 고-수준(high level)으로 전달될 수 있는 programmability를 한정하는 것이다. 다수의 프로그래머블 네트워크 프로토타입이 특정 네트워킹 기술을 목표로 진행되고 있으며, 통신서비스를 지원하는 데 부족한 부분을 극복하기 위해 목표로 하는 네트워킹 기술을 프로그래머블하게 만드는 것이다. 예로, ATM과 같은 기술은 QoS 부분에 좀 더 프로그래머블하며 인터넷과 같은 기술은 확장성(scalable) 부분에, 이동 네트워크와 같은 기술은 사용 가능한 대역폭에 제한을 가지고 있다.

가. IP 네트워크: Smart Packets

스마트 패킷은 프로그래머블 IP 환경에서 코드를 기반으로 한 패킷 개념으로 구현된 것으로 캔사스 대학에서 개발하였다[1)][4]. 스마트 패킷은 자바 클래스를 기반으로 in-band 또는 out-band 이동성 코드의 요소를 표현하며, 연속된 객체의 형태로 상태 정보를 전파하고 인증을 위한 식별자(identifier)를 전달한다. 액티브 노드 구조가 추상적 개념의 자원과 프리미티브의 집합을 스마트 패킷이 참조할 수 있게 한다. 액티브 노드는 다음의 요소들이 결합되어 있다.

- 자원 제어기(resource controllers): 노드 자원에 인터페이스를 제공
- 노드 관리자(node managers): 자원 사용의 정적 한계를 설정
- 상태 관리자(state managers): 스마트 패킷이 액티브 노드에 남겨 놓을 수 있는 정보의 양을 제어

이 외에 액티브 노드는 default 라우팅 방법과 스마트 패킷으로 전달되는 얼터너티브 라우팅 방법 모두를 지원하는 라우터 관리자(router manager)를 포함하고 있다. 액티브 노드는 경쟁하는 task들 사이에 CPU 사이클을 분할해 주기 위한 피드백 스케줄링 알고리즘과 안정된 대역폭 사용을 위한 credit 기반 흐름제어 알고리즘을 지원한다. 각 스마트 패킷에는 한 개의 쓰레드(thread)와 일정 양의 노드 자원이 할당된다.

스마트 패킷 테스트베드에서는 HTTP(Hyper Text Transfer Protocol)와 SMTP(Simple Mail Transfer Protocol) 프로토콜에서 과다하게 일어나는 Ack/Nak 응답을 줄여줌으로써 향상된 성능을 보이는 데 사용되었다.

나. ATM 네트워크: xbind

ATM 기술이 연결형 통신을 제공하고 멀티미디어 네트워크에 QoS 제공을 위한 admission 제어 및 자원 예약과 같은 기본적인 기능들이 지원된다 하더라도 복잡성 때문에 시그널링 요소들이 실제 사용되

1) 주의: BBN의 스마트 패킷 프로젝트와 다른 것임.

기에는 적합하지 않다. xbind는 통신 하드웨어로부터 제어 알고리즘을 분리하므로써 서비스 생성의 제약점을 극복하며, 가상(virtual) 스위치와 가상 링크의 추상화를 이용하여 노드 자원과 기능을 참조할 수 있는 개방 인터페이스 개발을 강조하고 있다[5]. 인터페이스는 ATM 네트워크에 있는 관리 및 제어 부분의 programmability를 지원하도록 설계되었다.

XRM 모델에 기반을 둔 xbind 광대역 커널은 광대역 네트워크, 멀티미디어 네트워크, 서비스 네트워크 등의 추상화된 세 가지 네트워크 모델을 통합한 것이다. 멀티미디어 네트워크는 프로그래머블 네트워크 관리, 네트워크 제어, 상태 관리, 연결 관리, 미디어 스트림 제어 등을 지원한다[5].

xbind 테스트베드는 다양한 광대역 서비스와 전송 및 QoS 보장형 시그널링 시스템을 지원하기 위한 서비스 생성과 개방 시그널링을 사용하여 멀티-벤더 ATM 스위치를 통합한다.

다. Mobile 네트워크: Mobiware

Mobiware는 xbind 모델의 programmability를 이동 서비스를 전달하는 패킷 기반 이동 네트워크로 확장한 것으로, 개방형 프로그래머블 이동구조를 가진다. Mobiware의 제어부분은 객체 기반의 CORBA programmability와 결합되어 있으며, 데이터 경로로 전송되는 자바 바이트 코드 기반의 액티브 전달 객체(예: 코드 플러그-인)를 받을 수 있다. 전달 계층에서는 액티브 전달 환경이 네트워크 내부에서 부가가치 서비스를 제공하는 베이스 스테이션으로 알고리즘을 삽입한다. 네트워크 계층에서는 이동장비와 액세스 포인트 및 이동 가능한 스위치에서 실행되는 분산 객체들이 프로그래머블 핸드오프 제어와 QoS 적용을 위한 서로 다른 형식을 지원하기 위해 상호작용을 한다. 링크 계층도 역시 프로그래머블하게 만들어질 수 있다.

다음의 이동 서비스가 Mobiware 도구를 사용하여 프로그램 되었다.

- QoS-controlled handoff: 적응 가능한 QoS API (Application Program Interface) 및 무선 채널 상태를 기반으로 한 오류 제어와 자동적인 미디어 scaling을 지원
- mobile soft-state: 정기적인 자원 예약과 재협상 과정을 통해 시시각각으로 변하는 QoS 요구에 부응하는 능력을 가진 이동장비를 제공
- flow bundling: cellular 참조 네트워크 방식에서 신속한 핸드오프를 지원

Mobiware 테스트베드는 이동장비에 웹 기반의 데이터 서비스 이외에 다양한 scalable 오디오/비디오 서비스를 지원한다.

2. Programmability의 수준

Programmability의 수준이라는 것은 네트워크 기반구조 내부에 적용할 수 있는 새로운 서비스의 정도를 표현한다. 이 수준은 동적인 수준에서부터 보수적인 수준까지의 범위를 가진다. 이 수준의 한쪽 끝에서는 캡슐이라는 것이 코드와 데이터를 전달한다. 캡슐은 코드를 가장 동적인 수단으로 표현하며, 서비스를 네트워크 내부로 적용시키는 기능을 한다. 수준의 다른 끝에는 새로운 서비스를 적용하기 위해 분산된 제어기들 사이에 RPC(Remote Procedure Call)를 사용한 준-정적인 네트워크 프로그래밍 인터페이스를 기반으로 네트워크 programmability에 접근하는 보수적인 방식이 있다. 이 수준의 범위 내에는 동적인 플러그-인과 액티브 메시지 및 RPC를 연결하는 다수의 방법론이 존재한다. 다른 접근방식에서는 새로운 서비스가 네트워크 기반구조에 적용되는 곳에서 속도, 유연성, 안전성, 보안성, 성능 등과 직접적인 관계를 가지고 있다.

가. Capsules: ANTS

ANTS(Active Network Transfer System)는 MIT에서 개발된 것으로 액티브 네트워크에 여러 가지 통신 프로토콜의 차별화된 적용을 가능하게 한다[6]. 액티브 네트워크에서는 이동 코드의 전달, 요구에 의한 코드의 로딩, 캐싱 기법 등과 같은 핵심

서비스를 제공한다. 핵심 서비스는 네트워크 구조를 새로 구성하거나 기존 네트워크 프로토콜의 확장을 가능하게 한다. ANTS는 새로운 캡슐-기반의 프로그래머블 네트워크 구조를 구축하기 위한 네트워크 프로그래밍 환경을 제공한다. 프로그램된 네트워크 서비스의 예로는 향상된 멀티캐스트 서비스, 이동 IP 라우팅, 응용-수준의 필터링 등과 같은 것들이 있다. ANTS의 캡슐-기반 실행 모델은 다른 API 접근방식에 비해 최대의 네트워크 programmability를 위한 기반을 제공한다. 캡슐은 네트워크 programmability의 기본 단위로 제공되며 노드의 참조, 캡슐 관리, 제어 연산, soft-state 저장공간 서비스 등과 같은 프리미티브들과 연결되어 처리된다. 액티브 노드는 캡슐과 포워딩 루틴을 실행하고, 지역적인 상태를 관리하며, 새로운 서비스를 자동으로 적용하기 위한 코드 분배 서비스를 지원한다. ANTS 도구는 처리되는 캡슐의 양을 노드 자원 관리를 위한 매트릭으로 제공한다.

나. Active Extensions: Switchware

Switchware는 펜실바니아 대학에서 개발한 것으로, 인터넷과 같은 공유 기반구조에서 필요로 하는 안전성 및 보안성과 이에 대비한 프로그래머블 네트워크의 유연성과 균형을 이루도록 하였다[7]. Switchware 도구는 보안성을 가진 네트워크 구조를 프로그래밍할 때 유연성, 안전성, 보안성, 성능 및 가용성들간의 trade-off를 가지는 네트워크 구축을 가능하게 한다. 운영체제 수준에서 액티브 IP 라우터 요소는 시스템의 완전성을 보장하는 보안 기반을 제공한다. 액티브 확장(active extensions)은 암호와 인증 및 프로그램 검증이 포함된 보안 메커니즘을 통해 secure 액티브 라우터로 동적으로 적재될 수 있다. 액티브 확장의 동작은 'heavyweight' 방식을 적용한 검증을 수행한다.

액티브 확장은 액티브 패킷을 사용하여 동적인 네트워크 프로그래밍을 위한 인터페이스를 제공한다. 액티브 패킷은 이동이 가능하고 캡슐에서와 비슷한 방식으로 네트워크를 재구성할 수 있다. 액티브 패킷은 Caml이나 PLAN(Programming Language for Active Network)과 같은 함수 언어(functional language)로 작성되며, 액티브 확장에서 지원되는 노드-상주 서비스 루틴을 실행시키는 lightweight 프로그램을 전달한다. 이미 액티브 확장에서 일정 수준의 보안검증이 적용되었기 때문에 액티브 패킷은 액티브 확장에서보다 적은 시험/검증이 요구된다. 액티브 패킷은 액티브 확장 소프트웨어에서 정의된 인터페이스를 통해서만 노드의 상태를 참조할 수 있다. Switchware는 액티브 확장에서 heavyweight 보안 검증을 적용하고 액티브 패킷에서는 lightweight 보안 검증을 수행한다. 이러한 방식은 보안성과 성능 요구사항 사이에 균형을 유지하는 네트워크 구성을 가능하게 한다.

다. Composition Languages: CANEs

캡슐과 액티브 메시지 및 액티브 확장 등은 기존 서비스에 요소를 추가하거나 새로운 구성요소를 통해 새로운 서비스 생성에 도움을 주고 있다. 켄터키 대학과 죠지아 대학에서 수행한 CANEs 프로젝트는 네트워크 programmability를 위한 일반 모델로 서비스 합성 규칙을 정의하고 적용하는 것을 목표로 하고 있다[8]. 합성방식은 요소들로부터 합성된 네트워크 서비스를 구축하는 데 사용되며, 프로그래머블 네트워크 서비스 구축을 위한 요소들을 관리하는 기능을 가진 프로그래밍 언어로 기술된다. 합성방식의 기능은 다음과 같다.

- 요소들이 실행될 순서 제어
- 요소들 사이의 공유 데이터 제어
- 합성물의 생성과 실행시간을 포함하는 바인딩 시간
- 합성물을 실행시키는 이벤트로 정의되는 시동 방법
- 액티브 노드에 존재하거나 패킷으로 전달되는 다중 요소들 사이의 기능 분리

PLAN, ANTS, Netscript 등이 합성방식의 예로 들 수 있다. LIANE(Language Independent Active Network Environment)가 CANEs 프로젝트에서 기술한 기능 모두를 통합하는 합성방식으로 제안되

었다. LIANE의 핵심 아이디어는 처리 슬롯을 포함하고 있는 기본 프로그램들로 구성된 서비스들이다. 사용자들은 구성을 위한 프로그램을 처리 슬롯에 넣는다. 서비스 합성에 대한 CANEs의 정의는 어떻게 서로 다른 접근방식을 서로 보완할 것인가를 나타내는 네트워크 programmability에 대한 Opensig 접근방식을 포함한다.

라. 네트워크 APIs: xbind

xbind 광대역 커널은 바인딩 구조와 BIB(Binding Interface Base)라는 노드 인터페이스의 집합을 기본으로 한다. BIB는 노드 상태와 네트워크 자원에 대한 추상화를 제공한다[9]. 바인딩 알고리즘이 BIB 위에서 실행되며, 추상화를 통하여 네트워크 자원에 대한 QoS 요구를 바인드한다. BIB는 고급 프로그래밍 언어를 통한 서비스 생성을 지원하기 위해 설계되었다. 인터페이스는 일반적인 programmability를 지원하는 동안에는 정적으로 작동한다. BIB 인터페이스의 준-정적인 성질은 어떤 서비스가 전개되기 전에, 에뮬레이션 플랫폼상에 새로운 기능의 완전성을 검증하거나 시험하는 것을 가능하게 한다. 액티브 패킷이나 프로그램과 사용자 데이터를 포함한 캡슐의 개념은 programmability를 위한 xbind 방식에서는 고려하지 않는다. 통신은 분산 객체와 제어기 사이를 CORBA를 기본으로 하는 RPC를 사용하여 수행된다. xbind 방식은 멀티-벤더 스위치들 사이에 자원의 공유 및 분할을 제어할 수 있도록 지원하므로써 상호운용성(interoperability)을 증가시킨다.

3. 프로그래머블 통신 추상화

프로그래머블 통신 추상화는 서로 다른 미들웨어와 네트워크 노드에서 지원을 필요로 하는 네트워킹 기반구조의 가상화(virtualization) 및 programmability의 수준을 나타낸다. 자원의 가상화 및 programmability는 프로그래머블 네트워킹에 기본적인 개념이다. 프로그래머블 통신 추상화는 노드 자원에서부터 완전한 프로그래머블 가상 네트워크까지의 범위를 가질 수 있으며, 여기에는 프로그래머블 가상 라우터, 가상 링크 및 이동 채널 등이 포함된다. 가상화를 통하여 네트워크 기반구조를 추상화하고 그것을 프로그래머블하게 만드는 것은 이 분야의 많은 프로젝트들이 기여한 주요 내용이다.

가. Active Node Abstractions: NodeOS

NodeOS라는 노드 운영체제는 액티브 네트워킹 하부구조(framework)의 최하위 수준을 표현한다[10]. NodeOS는 라우터에서 쓰레드, 채널, 플로우 등과 같은 통신 추상화를 지원하는 다수의 실행환경에서 이용되는 노드 커널 인터페이스를 제공한다. 액티브 네트워크 캡슐화 프로토콜(Active Network Encapsulation Protocol: ANEP)을 기반으로 한 캡슐화 기법은 단일 액티브 노드 내에서 다수의 실행환경의 적용을 지원한다. ANEP는 같은 물리적 노드상에 공존하는 다수의 실행환경을 통하여 패킷의 라우팅을 가능하게 하는 캡슐화 형식을 정의한다. 서로 다른 타입의 물리적 노드에 대한 실행환경의 이식성(portability)은 공통 및 표준 인터페이스를 이용하는 NodeOS에 의해 달성된다. 이 인터페이스는 쓰레드, 메모리, 채널, 플로우 등의 네 가지 프로그래머블 노드 추상화를 정의한다. 쓰레드, 메모리, 채널 등은 실행환경에서 사용되는 연산, 저장소, 통신 능력 등을 추상화하며 플로우는 보안, 인증, 수신제어 기능을 가진 사용자 데이터-경로를 추상화한다. 실행환경은 쓰레드와 플로우와 연관된 채널을 생성하기 위해 NodeOS 인터페이스를 사용한다. NodeOS는 노드 연산과 통신 자원에 대한 참조를 통제하는 스케줄링 기법을 사용하여 QoS를 지원한다. 액티브 네트워킹을 위한 하부구조는 새로운 액티브 구조의 시제품 연구를 위한 테스트베드인 ABONE(Active network backBONE)에 구현되었다.

나. Virtual Active Networks: Netscript

콜롬비아 대학에서 수행하는 Netscript 프로젝트

는 범용 언어 추상화를 사용하여 네트워크의 programmability를 갖기 위하여 함수형 언어 기반의 방식을 가진다[11]. Netscript는 네트워크 노드의 기능을 프로그램하기 위한 범용 추상화를 생성하는 강한 타입형 언어이다. 언어 기반 방식을 가지는 다른 액티브 네트워크 프로젝트와는 달리 Netscript는 프로그래머블 추상화로서 가상 액티브 네트워크를 지원하기 위해 개발되었다. Netscript는 MIB(Management Information Base)를 생성하는 언어 확장을 통하여 관리를 자동화한다. Netscrip에서 구분되는 특성은 postscript와 비슷한 방식으로 액티브 네트워크를 위한 범용 언어의 제공을 추구하는 것이다. 즉, postscript가 프린터 엔진의 programmability를 포착하는 것처럼 Netscript는 네트워크 노드 함수의 programmability를 포착한다. Net script 통신 추상화는 노드와 가상 액티브 네트워크를 구성하는 가상 링크의 집합을 포함한다.

다. Virtual ATM Networks: Tempest

캠브리지 대학의 Tempest 프로젝트는 광대역 ATM 환경에서 다수가 공존하는 제어구조의 발전을 연구한 것이다[12]. 여기에는 네트워크 제어를 구성하기 위한 소프트웨어 이동 에이전트를 사용한다는 것과 단일 서비스에 집중된 제어구조를 고려한다는 것 등의 새로운 기술적인 방식을 포함하고 있다. Tempest는 두 가지 수준의 programmability와 추상화를 지원한다. 첫째로, ATM 스위치 자원의 분할로 인한 논리적 네트워크 요소인 switchlet은 운용중인 네트워크로 차선의 제어구조 사용을 가능하게 한다. 둘째로, 기존 제어구조를 구성하는 프로그램을 동적으로 로딩하므로써 서비스들을 개량할 수 있다. ATM 네트워크에 있는 자원들은 ariel이라는 스위치 제어 인터페이스와 prospero라는 자원 분할기의 두 가지 소프트웨어 요소를 사용하여 분리할 수 있다. ATM 스위치상에 있는 ariel 서버와 통신하는 prospero는 자원을 분할하고, 생성된 각 switchlet을 위하여 분리된 제어 인터페이스를 export한다. 네트워크 builder는 제어구조를 생성하고, 변경 및 관리한다.

4.구조적 영역

구조적 영역은 목표 구조나 응용 영역을 나타낸다. 대부분의 프로그래머블 네트워크 프로젝트는 네트워크 내의 서비스와 관련이 있으며 QoS 제어, 시그널링, 관리, 전송 및 응용 등과 같은 특정 구조 영역을 목표로 하고 있다. 다음에 살펴볼 세 개의 프로젝트는 응용, 자원 관리 및 네트워크 관리 등의 영역에 관하여 기술한다.

가. Application-Level: Active Services

액티브 네트워킹 연구의 주요 부분에 비해, 참고문헌에서는 응용 계층의 연산 모델을 제한하므로써 인터넷 구조의 모든 라우팅 및 포워딩 semantic의 보호를 요구하고 있다[13]. AS1(액티브 서비스 버전 1) 프로그래머블 서비스 구조는 클라이언트가 네트워크 내의 전략적 위치에서 서비스 에이전트를 다운로드하고 실행할 수 있게 한다. “servents”라는 서비스 에이전트는 IP 계층의 integrity를 위반하는 라우팅 테이블과 포워딩 기능 처리로부터 제한된다. AS1 구조는 다음과 같은 다수의 구조적 요소들을 포함한다.

- 서비스 환경: 프로그래밍 모델과 servent에서 사용 가능한 일련의 인터페이스 정의
- 서비스 위치: 클라이언트가 AS1 환경과 랑테뷰하는 것을 허용
- 서비스 관리시스템: 로드가 심한 상황에서 servent의 로드 균형과 허용 제어를 사용하여 servent에 자원을 할당
- 서비스 제어시스템: AS1 구조 내에서 시작된 servent의 동적인 클라이언트 제어를 제공
- 서비스 연결 장치: soft-state 게이트웨이를 통하여 AS1 환경과 직접 상호작용할 수 없는 클라이언트를 위한 방법을 제공
- 서비스 합성방식: 클라이언트가 다수의 서비스 클러스터와 접촉하거나 클러스터 내부/외부에서 실

행되는 servent를 연결하는 것을 허용

AS1 구조는 응용 영역의 범위를 지원하는 응용 계층에서 프로그래머블하다. 참고문헌에서 MeGa 구조는 액티브 미디어 게이트웨이 서비스를 지원하기 위해 AS1을 사용하여 프로그램 되었다[13]. 이 경우, servent는 응용-레벨의 비율(rate) 제어와 trans-coding 기법을 위한 지원이 제공된다.

나. Resource Management: Darwin

카네기 멜론 대학의 Darwin 프로젝트는 인터넷 사용자에게 부가가치 및 구성 가능한 서비스를 위한 플랫폼을 제공하는 차세대 IP 네트워크의 미들웨어 환경을 개발하는 것이다[14]. Darwin 프로젝트는 QoS를 지원하는 구성 가능한 자원 관리에 초점을 맞추고 있다. Darwin의 하부구조는 사용자 요구사항을 내부 자원에 매핑시키는 서비스 중개자인 Xena, Beagle 시그널링 프로토콜을 사용하여 Xena와 통신하는 자원 관리자, 서비스 측면을 기본으로 하는 계층적 스케줄링 분야 등을 포함하고 있다. Xena 구조는 IP 포워딩/라우팅 함수가 있어야 하며, 시스템 내에서 액티브 패킷 기술의 제한된 사용만을 허용한다.

Darwin에서는 광대역 커널 및 액티브 서비스와 비슷한 개념을 구축하는 제어 단계를 소개하고 있다. Xena 구조는 제한적인 방식으로 액티브 기술을 결합하고 프로그래머블하게 만든다. 일련의 서비스 대리자가 액티브 패킷을 위한 지원을 제공하며, 대리자는 응용 관련 처리나 부가가치 서비스를 지원하기 위해 IP 라우터나 서버로 동적으로 주입될 수 있다. Darwin 방식의 특징은 공간, 조직, 시간 제약 등으로 정의된 사용자의 특정 서비스 요구에 따라 구성 가능한 메커니즘들이 있다는 것이다. 이러한 구조적 메커니즘들이 조화를 이루며 작동할 때 가장 효과적이며, QoS 구조 요소들과 결합될 수 있다. 예로, Beagle 시그널링 시스템은 자원 예약을 위한 RSVP (Resource ReSerVation Protocol) 시그널링을 지원하기 위해 프로그램될 수 있는 반면, Xena 자원 중개자 및 계층 스케줄러는 트래픽 제어를 지원할 수 있다.

다. Network Management: Smart Packets

BBN의 스마트 패킷 프로젝트는 액티브 네트워킹 기술을 기반으로 크고 복잡한 네트워크의 성능을 향상시키는 것을 목표로 하고 있다[15]. 스마트 패킷은 관리를 결정할 지점을 관리될 노드쪽에 가깝게 이동하고, 언어 구축을 위한 관리 개념을 추상화하는 데 사용된다. 관리 센터에서는 관리되는 노드에 프로그램을 보낼 수 있다. 이와 같은 관리 과정은 백트래픽 및 검사를 요하는 데이터의 양을 줄이려는 관리 센터의 특정 관심분야에 맞추어질 수 있다. 스마트 패킷은 ANEP를 사용하여 캡슐화된 데이터 부분(payload)과 헤더로 구성되어 있다. 스마트 패킷들은 실행될 프로그램을 전달할 수 있으며, 실행 결과로 발생하는 정보나 오류 메시지를 전달할 수 있다. 스마트 패킷은 다음의 두 가지 프로그래밍 언어로 작성된다.

- sprocket: 고급언어 C와 비슷한 언어
- spanner: 저수준의 어셈블리와 비슷한 언어

sprocket 프로그램은 spanner 코드로 번역되며, spanner 코드가 머신에 독립적인 이진 코드로 어셈블되어 스마트 패킷으로 적재된다. 이러한 프로그램들은 네트워킹 함수, MIB 정보 검색 등과 같은 작업을 수행한다.

IV. 비교 및 분석

이 장에서는 지금까지 살펴본 다양한 프로그래머블 네트워크 프로젝트를 특징별로 분석한 내용들을 기술한다. <표 1>에 IV장에서 조사된 프로그래머블 네트워크의 특징을 각 프로젝트별로 비교하였다.

1. 개방형 프로그래머블 인터페이스

많은 프로그래머블 네트워크 프로젝트들이 개방형 프로그래머블 네트워크 인터페이스를 사용한다. 개방 인터페이스는 서비스 프로그래밍과 새로운 네트워크 구조를 개발하기 위한 기초를 제공한다.

<표 1> 프로그래머블 네트워크 비교

프로젝트	특징							
	구조적 영역	네트워킹 기술	프로그래머블 통신 추상화	Programma-bility 수준	프로그래밍 방법론			
					합성언어	분산객체	이동코드	캡슐화
Active Services	composing application level services	인터넷	application services	dynamic	tcl/otcl		X	
Smart Packets, BBN	network management	인터넷	managed nodes	dynamic, discrete	sprocket & spanner		X	X
NetScript	composing network services & VANs	인터넷	VANS	dynamic, discrete	NetScript		X	X
ANTS	composing network services	인터넷	internet protocols	dynamic, integrated	JAVA		X	X
CANEs	composing services	인터넷	composable services	dynamic	LIANE		X	X
SwitchWare	composing network services	인터넷	internet protocols	dynamic	PLAN & Caml		X	X
Smart Packet, 캔사스대학	composing network services	인터넷	internet protocols	dynamic	JAVA		X	X
Liquid S/W	Investigating mobile code technology	인터넷		dynamic	JAVA		X	
NodeOS	enabling network programability	인터넷	network node				X	X
Xbind	enabling telecomm. Service creation	ATM	multimedia networks	static	CORBA /IDL	X		
Darwin	integrated resource management & value-added services	인터넷	flows	quasi-static		X	X	
Mobiware	wireless QoS & mobile QoS control	Mobile	universal mobile channels	quasi-static	CORBA/IDL & JAVA	X	X	
Tempest	enabling alternative control arch.	ATM	network control architectures	quasi-static	CORBA /IDL	X	X	

xbind 광대역 커널은 ATM 네트워크에 있는 장치, 상태, 제어 등을 추상화하기 위해 CORBA/IDL (Interface Description Language)을 사용한 포괄적인 BIB를 지원한다. ANTS, Switchware, CANEs 등과 같은 다수의 다른 프로젝트들은 새로운 서비스의 합성과 네트워크 programmability를 가능하게 하고, 노드 프리미티브를 추상화하는 개방형 API의 사용을 증진시키는 IP 네트워크를 프로그래밍하는 데 초점을 맞추고 있다. 많은 네트워크 프로그래밍 환경들은 서비스 합성을 위한 개방 인터페이스를 제공하는데 서로 다른 접근방법을 가지고 있다.

두 가지 상반되는 제안이 xbind와 ANTS API에 포함되어 있다. 유연한 새로운 API를 전개하기 위해 ANTS 방식에서는 높은 동적 프로그래밍 방법론을 제시하는 반면, 간단한 RPC 모델에 비해 복잡한 프로그래밍 모델로 표현된다. 반면, xbind 바인딩 인터페이스와 프로그래밍 패러다임은 CORBA/IDL과 RPC 매커니즘을 기본으로 하고 있다. 캡슐-기반 programmability와 비교하면 xbind 방식은 더 정적이며 프로그래밍 모델도 덜 복잡하다. 이들 방식들이 네트워크 programmability의 양 극을 표현하고 있다.

2. 가상화 및 자원 분할

많은 프로젝트들이 다른 타입의 통신 추상화의 programmability를 지원하기 위해 가상화 기술을 사용한다. Tempest 하부구조는 네트워크 기반구조의 가상화의 사용 예로 나타난다. 저수준의 물리적인 스위치 인터페이스는 switchlet이라는 스위치 분할을 위한 인터페이스의 집합을 생성하기 위해 추상화된다. Switchlet은 다수의 제어구조가 공존하는 것을 허용하며, 같은 물리적인 스위치 자원(용량, 스위칭 테이블, 이름 공간 등)을 공유한다. 프로그래머블 네트워크에 나타나는 추상화는 다수의 서비스 및 프로토콜을 가능하게 하는 안전한 자원분할 전략과 공존하는 서로 다른 프로그래머블 네트워킹 구조로 이루어진다. 이러한 방식으로 네트워크의 추상화는 이전에는 고려하지 못했던 프로그래머블 네트워크의 새로운 수준을 표현한다. 네트워크 요소의 모든 타입은 추상화되고 스위치와 링크로부터 switchlet, 액티브 노드, routelet, 가상 네트워크 등으로 프로그래머블하게 만들어질 수 있다.

NodeOS 인터페이스는 노드 자원에 비슷한 추상화를 제공한다. 개방형 인터페이스의 사용은 다수의 네트워크 프로그래밍 환경(또는 액티브 네트워킹을 사용하는 실행환경)이 공통의 물리적인 노드 구조 내에 공존하는 것을 가능하게 한다. 이럴 경우, ANEP 프로토콜은 상이한 실행환경에 패킷을 전달하는 메커니즘으로 캡슐화를 제공한다. 이런 방식으로 캡슐화를 사용하는 것은 하나의 공통 노드 커널에서 상이한 실행환경의 중복을 허용하는 것이다.

3. 프로그래머블 가상 네트워킹

새로운 서비스의 동적 합성과 전개는 완전한 네트워크 구조를 가상 네트워크로 합성하는 것을 포함하기 위해 확장될 수 있다. Netscript 프로젝트는 IP 네트워크 위에 가상 액티브 네트워크의 개념을 지원한다. 가상 네트워크 엔진들이 가상 액티브 네트워크로(부터) 가상 노드들과 가상 링크들을 서로 연결시킨다. Tempest 하부구조는 ATM 하드웨어 위에 안전한 분할을 사용하여 가상 네트워크의 개념을 지원한다. Tempest 하부구조 내에서 물리적인 스위치 분할의 추상화는 다수가 공존하는 제어구조의 구현을 이끌어 냈다. Tempest의 전략은 연결형 ATM 기술을 통하여 QoS를 나타내고 선택적인 제어구조들 사이에 물리적인 자원공유 기술을 추구하는 것을 목표로 하고 있다. Darwin과 Netscript 프로젝트는 구성 가능한 방식으로 기존의 물리적인 기반구조를 공유하는 개념을 지원한다. NodeOS 프로젝트는 공존하는 실행환경을 위한 기능들을 제공한다.

V. 결 론

이 글에서는 프로그래머블 네트워크의 최근 기술 동향에 대하여 알아보았다. 프로그래머블 네트워킹 분야에서 현재 수행되고 있는 프로젝트 사이의 관계를 이해하기 쉽도록 프로그래머블 네트워크의 네트워킹 기술, programmability 수준, 프로그래머블 통신 추상화, 구조적인 영역, 프로그래밍 방법론 등의 특징별로 구분하여 기술하였다.

결론적으로 네트워크 programmability를 증진시키기 위하여 다음과 같은 패러다임의 전환이 중요하다.

- separation: 소프트웨어로부터 하드웨어의 분리
- availability: 개방형 프로그래머블 인터페이스의 유용성
- virtualization: 네트워킹 기반구조의 가상화
- rapid creation & deployment: 새로운 네트워크 서비스의 신속한 생성과 전개
- 안전한 자원분할 및 공존: 같은 물리적인 네트워킹 하드웨어상에 상이한 네트워크 구조의 공존 및 안전한 자원분할

프로그래머블 네트워크는 개방형 프로그래머블 인터페이스, 자원분할, 네트워킹 기반구조의 가상화 등을 통하여 가상 네트워크 구조를 구축하고 합성 및 전개하기 위한 기반을 제공한다.

참 고 문 헌

[1] Andrew T. Campbell *et al.*, "A Survey of Programmable Networks," *ACM Computer Communications Review*, Vol. 29, No. 12, Apr. 1999, pp. 7 – 23.

[2] 이남희, "Open Signaling 및 개방형 시스템 제어 기술," 한국통신학회지, 제17권 2호, Feb. 1999, pp. 54 – 67.

[3] 하석재, 최양희, "액티브 네트워크," 정보과학회지, 제17권 3호, Mar. 1999, pp. 37 – 45.

[4] A.B. Kulkarni *et al.*, "Implementation of a Prototype Active Network," *First International Conference on OPENARCH*, San Francisco, Apr. 1998.

[5] M.C. Chan *et al.*, "On Realizing a Broadband Kernel for Multimedia Networks," *3rd COST 237 Workshop on Multimedia Telecommunications and Applications*, Barcelona, Spain, Nov. 1996, pp. 25 – 27.

[6] D. Wetherall *et al.*, "ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols," *IEEE OPENARCH '98 Proc.*, San Francisco, Apr. 1998.

[7] D.S. Alexander *et al.*, "The SwitchWare Active Network Architecture," *IEEE Network*, Vol. 12, No. 3, 1998.

[8] "CANEs: Composable Active Network Elements," http://www.cc.gatech.edu/projects/canes/

[9] C.M. Adam *et al.*, "The Binding Interface Base Specification Revision 2.0," OPENSIG Workshop, Cambridge, UK, Apr. 1997.

[10] L. Peterson, "NodeOS Interface Specification," *Technical Report, Active Network NodeOS Working Group*, Feb. 1999.

[11] Sushil da Silva *et al.*, "Composing Active Services in Netscript," *DARPA Active Networks Workshop*, Tucson AZ, Mar. 1998.

[12] Van der Merwe *et al.*, "The Tempest–A Practical Framework for Network Programmability," *IEEE Network*, Nov. 1997.

[13] E. Amir *et al.*, "An Active Service Framework and its Application to Real-time Multimedia Transcoding," *Proc. ACM SIGCOMM '98*, Vancouver, 1998.

[14] P. Chandra *et al.*, "Darwin: Customizable Resource Management for Value-added Network Services," *6th IEEE ICNP '98*, Austin, Oct. 1998.

[15] B. Schwartz *et al.*, "Smart Packets for Active Networks," *2nd International Conference on OPENARCH*, New York, 1999.