

유연성을 갖는 분산 해쉬 테이블 기반의 피어 투 피어 인터넷 텔레포니 서비스의 구현

Implementation of a Flexible Peer-to-Peer Internet Telephony Service Using an Underlying DHT

이 주 호^{*} 김 재 봉^{**} 정 충 교^{***}
Lee, Ju-Ho Kim, Jae-Bong Jeong, Choong-Kyo

Abstract

Internet telephony provides voice communication services with added flexibility for multimedia extension at a lower cost compared to traditional telephone systems. We implemented an internet telephony system as an overlay network without a centralized server, using a distributed hash table (DHT). Compared to the current server-based internet telephony system, our system is fault-tolerant, scalable, and can be flexible extended to various services and advanced to integrated service. To demonstrate the high flexibility of our DHT-based internet telephony system, we made our system cooperate with web servers. Web users can check up others' online states and establish voice communication sessions to online users at a mouse click. This technology can be applied to more complex services such as multimedia messaging or video conference service.

키워드 : 오버레이, 피어 투 피어, 분산 해쉬 테이블, 인터넷 텔레포니
Keywords : *overlay, peer-to-peer, distributed hash table, internet telephony*

1. 서론

인터넷 텔레포니에 관한 지금까지의 연구들은 대부분 서비스의 아키텍처 측면보다는 음성 데이터가 송·수신되는 프로토콜에 그 중점을 두었고 서비스는 중앙 집중형 구조를 그 기반으로 하였다. 그러나 기존의 중앙 집중형 서비스 아키텍처는 중앙 서버에의 극심한 의존성 때문에 확장성, 강인성, 유연성 등에서 분해를 가지고 있었다.

본 연구는 기존 서비스의 이런 한계에 대한 해결책을 제시하기 위해, 분산 해쉬 테이블을 사용하여 오버레이 네트워크를 구성하고 이를 기반으로

피어 투 피어 인터넷 텔레포니 서비스를 구현하였다. 오버레이 네트워크는 응용 계층 위에서 구성되며 하위 계층에 투명성을 갖는데, 이 연구에서는 오버레이 네트워크의 우수한 확장성, 강인성, 그리고 결합 저항성을 위해서 분산 해쉬 테이블 기술을 사용하였다. 또한 분산 해쉬 테이블을 사용하여 오버레이 네트워크를 구성하였을 경우 이를 이용하는 서비스가 다른 서비스들로 유연하게 확장되어 질 수 있음을 보이기 위해 간단한 웹 기반 통합 서비스 프로토타입을 구현하였다.

2. 관련 연구

2.1 피어 투 피어 오버레이 네트워크

오버레이 네트워크는 응용 계층위에 논리적 topology를 구성하여 구현되고 하위 계층에 투명성

* 강원대학교 대학원 컴퓨터정보통신공학과 석사과정
** 강원대학교 컴퓨터학부 컴퓨터정보통신전공 학사과정
*** 강원대학교 컴퓨터학부 교수, 공학박사

을 가지며 물리적 토폴로지에 독립적이다. 이러한 오버레이 네트워크는 구조적인 오버레이 네트워크와 비구조적인 오버레이 네트워크로 나뉜다.

구조적인 오버레이 네트워크는 분산 해쉬 테이블 등을 사용하여 트리(tree), 링(ring), 나면체(hypercube), 혹은 하이브리드(hybrid) 구조의 논리적 토폴로지를 구성하며, 그 구조에 따라 데이터를 검색 혹은 메시지를 라우팅하는 알고리즘이 다르다. 이러한 구조적인 오버레이 네트워크는 우수한 확장성, 강인성, 그리고 결합 허용성을 갖는다는 연구 결과가 연구 [5, 6, 7, 8]에 잘 나타나있다.

비구조적 오버레이 네트워크는 Napster[1], Gnutella[2] 서비스에서 사용된 기술로서 특정한 논리적 토폴로지가 존재하지 않고, 플러딩이나 중앙 서버 인덱싱을 사용하여 데이터를 검색한다

앞으로 특별한 언급이 없는 한 이 논문에서 오버레이 네트워크는 구조적 오버레이 네트워크의 의미로 사용한다.

연구 [3]에서는 분산 해쉬 테이블을 사용한 피어 투 피어 SIP (Session Initiation Protocol) 텔레포니 서비스의 구현 방안을 개념적으로 제안하였다.

Skype [13, 14]는 많은 사용자 층을 확보하고 있는 피어 투 피어 인터넷 텔레포니 서비스이다. 그러나 Skype 서비스는 로그인 서버에 일부만 의존하기 때문에 순수한 피어 투 피어 기반으로 볼 수 없고 이전의 중앙 집중형 구조가 가진 문제점을 완전하게 해결하지 못하였다.

우리의 연구에서는 SIP 프로토콜의 구현에 국한되지 않는 일반적인 인터넷 텔레포니 서비스의 완전 분산형 구현이 가능함을 보이고 이렇게 할 경우 다양한 서비스로의 확장된 기능이 쉽게 구축될 수 있음을 보이는 데 초점을 맞추었다.

2.2 분산 해쉬 테이블

우리가 구현한 피어 투 피어 인터넷 텔레포니 서비스는 중앙 서버에 의존하지 않기 때문에 피어 노드들이 유기적인 관계를 가지고 피어 투 피어 오버레이 네트워크를 구성해야 한다. 우리는 오버레이 네트워크를 구현하기 위해 Bamboo[4, 5] 분산 해쉬 테이블을 사용하였다. Bamboo 외에도 CAN[6], Chord[7], Pastry[8]와 같은 분산 해쉬 테이블이 사용될 수 있지만 Bamboo는 다른 기술에 비해 물리적 인접성을 고려할 수 있으므로 텔레포니와 같은 실시간성 서비스에 보다 적합하고 특히나 무선 환경에서 더 나은 효율성을 가질 수 있다.

Bamboo 오버레이 네트워크는 링 구조와 트리 구조가 혼합된 하이브리드 구조를 사용하는데, 오버레이 네트워크 내의 모든 노드는 고유한 노드 아이디를 가지고 전체 링에서 한 좌표를 차지한다. 각각의 노드는 논리적 토폴로지 상에서 자신과 근

접한 이웃 노드와 정보를 교류하여 트리 구조로 라우팅 테이블을 관리 및 유지한다. 즉, 네트워크는 여러 노드들이 모여 링형 구조를 형성하지만, 각 노드의 라우팅 테이블은 트리 구조를 가진다. Bamboo 오버레이 네트워크에 저장되는 모든 객체는 (key, value) 형태로 나타내지며 key와 가장 가까운 노드 아이디를 갖는 노드에 저장된다.

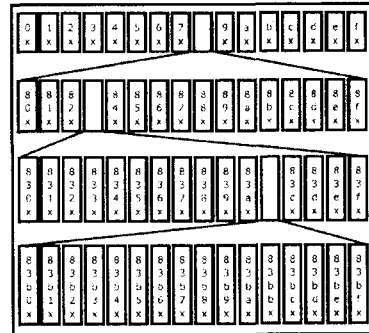


그림 1 Bamboo 노드 83bx의 라우팅 테이블

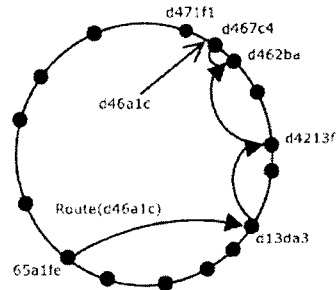


그림 2 노드 65a1fe에서 노드 d46a1c로의 메시지 라우팅

그림 1은 노드 83bx의 라우팅 테이블을 나타낸다. 여기서 x는 임의의 접미사를 의미한다. 모든 노드 아이디는 일정 단위의 순차적 배열로 이루어 지는데 b는 단위의 크기를 결정하는데 사용된다. 일정 단위 값의 크기는 0부터 최대 $2^b - 1$ 로 설정 되는데, 예를 들어 b를 기본 설정 값인 4로 하였을 때 일정 단위로써 16진수 단위를 사용한다.

그림 2는 Bamboo 오버레이 네트워크 상에 있는 노드 65a1fe에서 d46a1c의 키를 갖는 객체를 저장하고자 할 때 이 키와 가장 가까운 아이디를 갖는 노드 d467c4로 객체가 라우팅되는 과정을 보여 준다. 매 단계마다 색채의 이웃 노드 아이디가 상위 비트의 더 많은 비트들이 일치하도록 맞추어 나가므로 최대 $\log(N)$ 스텝에 목적지에 도달하게 된다. 여기에서 N은 노드의 수이다.

Bamboo에서는 일반적으로 IP(Internet Protocol)

주소와 포트 번호 혹은 공개키에 해쉬 함수 (SHA1)를 적용시켜 160bit 크기의 노드 아이디를 생성한다. 우리의 연구에서는 사용자가 사용하는 노드가 변하더라도 그 사용자가 사용하는 노드 아이디는 변하지 않게 하기 위해 이메일 주소에 해쉬 함수를 적용하여 노드의 아이디를 생성하였다.

3. 인터넷 텔레포니 서비스 디자인

서비스는 Bamboo 오버레이 네트워크를 기반으로 하여 제공되기 때문에 중앙 집중형 구조에 비해 결함에 대한 지향성과 확장성을 갖고 다른 서비스로 유연하게 확장되어질 수 있다.

또 Bamboo는 기본적으로 파일 공유 서비스를 제공하고 다양한 서비스의 지원 가능성을 내재하고 있다. 그러므로 Bamboo는 다양한 서비스들을 통합적으로 제공할 수 있는 그 기반이 될 수 있다.

이 연구에서는 통합적인 서비스의 일환으로 피어 투 피어 인터넷 텔레포니 서비스를 구현하였고 다른 서비스로의 유연한 확장을 입증하기 위해 웹 기반의 피어 투 피어 인터넷 텔레포니 서비스로 발전시켰다.

Bamboo 오버레이 네트워크 내의 모든 노드는 노드 아이디를 가지고 구별되며, 이러한 아이디는 노드의 IP 주소와 대응된다. 서비스 내에서 발신자와 착신자는 사용자 개체이고, 발신 노드와 착신 노드는 시스템 개체이다. 발신자는 발신 노드의 사용자이고 착신자는 착신 노드의 사용자이다. 서비스 내에서 발신자와 착신자는 그들의 이메일 주소를 통해 식별된다.

3.1 서비스의 구성

3.1.1 착신 노드의 IP 주소 검색 기능

Bamboo의 라우팅은 오버레이 네트워크 수준에서 이루어지기 때문에 네트워크 계층에서 이루어지는 라우팅에 비해 효율적이지 못하다. 따라서 우리는 발신자가 착신자의 현재 위치를 검색할 때에만 Bamboo 라우팅을 사용하고 그 이후 통화 세션은 네트워크 계층 라우팅을 사용하도록 구현하였다. 텔레포니 서비스가 기본적으로는 1:1 통신 기반이고 최대한 실시간성을 충족시켜야 하기 때문이다. 여기에서 발신자가 착신자의 현재 위치를 검색한다는 것은 발신자가 착신자의 이메일 주소를 가지고 착신자가 현재 사용하고 있는 노드의 IP 주소를 알아낸다는 의미로 이는 Bamboo 룩업을 통하여 이루어진다. 착신자가 사용하는 착신 노드는 가변적이며 사용자가 서비스에 로그인할 때 그 서비스를 지원하는 착신 노드는 사용자의 이메일 주소를 사용하여 자신의 노드 아이디를 생성한다. 여기서 서비스에 로그인 한다는 의미는 착신자가

서비스를 지원하는 임의의 노드를 사용하여 오버레이 네트워크에 참여한다는 것을 의미한다.

발신자는 서비스에 로그인한 후 음성 통신을 희망하는 착신자의 이메일 주소를 지정하여 통화를 신청하는데, 이렇게 하면 발신 노드는 착신자의 이메일 주소로부터 키를 생성하고 이 키를 이용하여 Bamboo 룩업을 수행한다. 착신 노드는 이에 대해 자신의 IP 주소로 응답한다. 그 이후에는 발신 노드가 착신 노드의 IP 주소를 이용하여 음성 통신 세션을 개설하고 물리적인 통신이 이루어진다. 이 착신자 룩업 기능은 우리가 구현한 프로토타입 시스템에서 LookupStage이라는 이름의 응용 모듈로 구현되어 있다.

3.1.2 음성 데이터 송·수신 기능

발신 노드는 착신 노드의 IP 주소를 알아낸 후에, 음성 데이터 송·수신 모듈을 사용하여 사진에 약속된 포트에 음성 데이터 및 제어 메시지를 송·수신한다. 발신 노드가 착신 노드의 IP 주소로 음성 통신 세션 연결 요청 메시지를 전달하면 착신 노드는 이에 대한 수락 여부를 결정한다. 수락할 경우에는 양 노드 간에 세션이 설립되고, 음성 데이터를 송·수신한다. 우리가 구현한 프로토타입 시스템에서는 클라이언트 응용 프로그램 VoipClient와 서버 응용 프로그램 VoipListener 그리고 관련 클래스 패키지들이 이 기능을 수행한다.

3.1.3 웹 페이지의 음성 통신 중개 기능

웹 페이지는 음성 통신을 위한 연결 고리 역할을 할 수 있다. 기존의 웹 페이지는 메일 주소를 아웃룩, 썬더버드와 같은 메일 에이전트와 링크시키는 기능을 제공할 수 있었다. 이러한 기능에 음성 통신을 위한 링크 기능을 추가하기 위해서 웹 페이지 링크를 구현하였다. 이러한 기능은 정적인 HTML 페이지를 사용해서는 제공할 수 없기 때문에 JSP (Java Server Page) 페이지를 사용하였다.

사용자가 웹 페이지를 읽을 때 JSP 스크립트에서는 페이지 내에 기록되어 있는 메일 주소를 가지고 Bamboo 룩업을 수행하여 착신 노드의 IP 주소를 검색할 수 있다. 검색은 LookupStage 응용 모듈이 담당하며, 성공 여부는 그 이메일 주소의 사용자가 현재 서비스에 로그인해 있는지 여부에 의해 결정된다. 페이지 내에 포함되어 있는 이메일 주소의 사용자가 현재 시스템에 로그인되어 있다면 검색은 성공하게 될 것이고 이 경우 웹 페이지에 그 이메일 주소의 사용자와 음성 통신이 가능하다는 것을 나타내는 애플릿 아이콘이 보여진다. 사용자가 이 애플릿 아이콘을 클릭하면 애플릿이 실행되고 애플릿은 발·착신 노드의 VoipListener가 정상적으로 실행되고 있는지를 확인하고나서 두 노드 간에 음성 송·수신 세션이 설립되도록 돕는

다.

웹 페이지 링크 기능은 웹 서버가 오버레이 네트워크에 참여하고 인터넷 텔레포니 서비스를 지원할 경우 이전의 서비스가 웹 기반의 인터넷 텔레포니 서비스로 유연하게 확장될 수 있다는 것을 잘 보여준다.

3.2 서비스 시나리오

서비스는 두 가지 시나리오에서 동작될 수 있다. 그림 3은 일반적인 피어 투 피어 인터넷 텔레포니 서비스를 나타내고, 그림 4는 이를 더 발전시킨 웹 기반의 피어 투 피어 인터넷 텔레포니 서비스를 나타낸다.

첫 번째 시나리오는 노드 A가 hermes@dmrl.kangwon.ac.kr을 착신자로 하여 착신 노드 B와 음성 통신 연결 세션을 설립하는 과정을 나타낸다. 노드 A와 노드 B는 논문에서 구현한 인터넷 텔레포니 서비스를 지원한다고 전제한다. 아래는 서비스가 이루어지는 절차를 나타낸다.

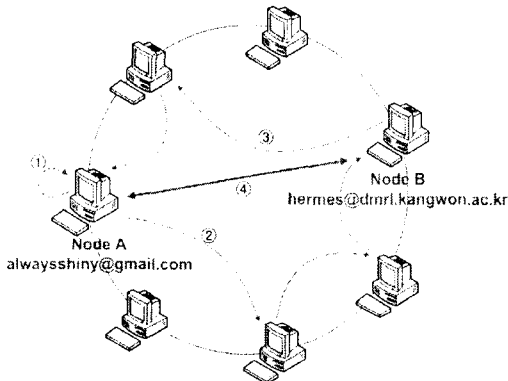


그림 3 서비스 시나리오 - 1

- ① 노드 A의 VoipClient는 로컬에서 실행 중인 LookupStage에게 착신자의 메일 주소 hermes@dmrl.kangwon.ac.kr을 알린다.
- ② 노드 A의 LookupStage는 착신 노드의 노드 아이디를 생성하고 그 아이디에 해당하는 노드를 Bamboo 라우팅을 통해 검색한다. 이는 노드 B의 LookupStage에게 IP 주소 요청 메시지가 전달됨을 의미한다.
- ③ 노드 B의 LookupStage는 자신의 IP 주소를 응답 메시지에 작성하여 노드 A의 LookupStage에게 응답한다. 이 역시 Bamboo 룩업 절차의 일부이다.
- ④ 노드 A의 VoipClient는 로컬의 LookupStage로부터 노드 B의 IP 주소를 응답받고, 노드 B의 VoipListener에게 음성 통신 세션 설립을 요청하여 세션을 설립한다.

두 번째 시나리오에는 발·착신 노드들 사이의

연결 고리 역할을 담당하는 웹 서버가 추가되어 있다. 노드 A, 노드 B, 그리고 노드 C는 우리가 구현한 인터넷 텔레포니 서비스를 지원하며 모두 Bamboo 오버레이 네트워크에 속한다고 가정한다. 서비스가 이루어지는 절차는 아래와 같다.

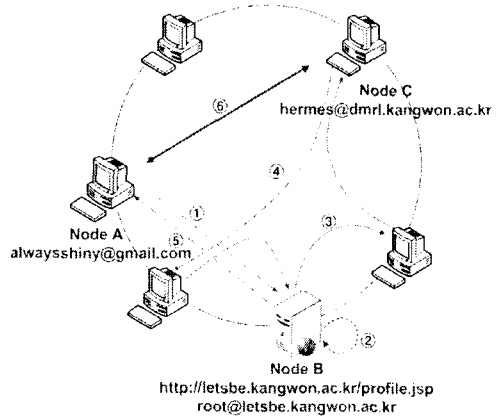


그림 4 서비스 시나리오 - 2

- ① 노드 A는 웹 서버 노드 B에게 웹 페이지를 요청한다.
- ② 노드 B는 웹 페이지에 기록되어 있는 이메일 주소를 로컬 노드에서 실행 중인 LookupStage에게 전달한다.
- ③ 노드 B의 LookupStage는 이메일 주소를 사용하여 노드 아이디를 생성해내고 이에 해당하는 노드 C의 LookupStage에게 IP 주소를 요청한다.
- ④ 노드 C의 LookupStage는 자신의 IP 주소를 응답 메시지에 넣어 노드 B의 LookupStage에게 응답한다. 이 응답으로 Bamboo 룩업이 완료된다.
- ⑤ 노드 B는 착신 노드인 노드 C가 오버레이 네트워크 내에 존재한다는 것을 확인하고 애플릿 버튼을 추가한 페이지를 작성하여 노드 A에게 응답한다.
- ⑥ 노드 A는 애플릿 버튼을 눌러서 노드 C의 VoipListener에게 음성 송·수신 세션 설립을 위한 요청 메시지를 전송하고 세션을 설립한다.

4. 구현

모든 프로그램 코드는 자바로 구현하였다. Bamboo 자체가 Java로 구현되어 있고 Java는 플랫폼에 독립적이라는 장점을 가지고 있기 때문이다. 착신 노드 검색 응용 모듈 LookupStage는 Bamboo 응용 프로그래머며 Bamboo 패키지의 bamboo.web.WebInterface와 유사한 방식으로 구현하였다. Bamboo 응용 프로그램을 위한 튜토리

열 [9, 10]과 Bamboo를 사용하여 멀티캐스트 서비스를 구현한 연구 [11]이 많은 참고가 되었다. 연구 [11]은 또한 우리의 서비스가 1:1 통신 기반에서 1:N 통신으로 발전할 수 있다는 가능성을 긍정적으로 뒷받침해준다.

음성 송·수신 응용 프로그램은 Java 확장 패키지인 Swing와 JMF(Java Media Framework)[12]을 사용하였는데, 이는 각각 사용자 UI(User Interface)와 음성 데이터 송·수신을 위하여 사용되었다. 실시간 음성 통신을 위한 프로토콜로는 RTP(Real-Time Protocol)을 사용하였다. 그러나 우리가 구현한 서비스는 착신자를 검색하는 Bamboo 록업 과정과 음성 통신 세션을 설립하는 과정이 서로 독립적이므로 특정 프로토콜에 종속적이지 않다.

웹 페이지에서 음성 통신을 위한 중개 기능은 JSP(Java Server Page)와 서명된(signed) 애플릿(Java Applet)을 사용하여 구현하였다. JSP 이외에도 ASP, PHP와 같은 스크립트 언어가 사용될 수 있으나 JSP의 경우 Java 기반의 JSP Engine을 사용하여 동작하기 때문에 같은 Java 기반의 Bamboo에 더 적합하다. 서명된 애플릿을 사용한 까닭은 서명되지 않은 애플릿은 기본적으로 웹 브라우저 자체에서 보안상의 이유로 거부하기 때문이다.

4.1 착신 노드 검색

Bamboo는 쉘 스크립트로 동작하는 이벤트-드라이브 응용 프로그램으로 다수의 스테이지들로 구성되어 있다. Bamboo 응용 프로그램이라 함은 Bamboo 스테이지를 의미하는데, 모든 스테이지는 Bamboo 시스템에 등록할 때 TCP/UDP 포트와 같은 역할을 하는 스테이지 아이디를 부여 받는다. 모든 스테이지들은 페이로드를 포함한 이벤트를 발생시켜서 로컬 노드 혹은 원격 노드에게 메시지

를 전달하고, 수신한 메시지를 처리한다.

아래는 Bamboo에서 제공하는 API 중에서 메시지를 전달 혹은 처리하기 위하여 발생시키는 이벤트들이다.

- **bambooRouteInit** 특정 노드의 특정 스테이지로 페이로드 객체를 전달하고자 할 때 발생시키는 이벤트
- **bambooRouteDeliever** 이벤트를 수신하였을 때 발생하는 이벤트로 로컬의 특정 스테이지에게 페이로드 객체를 전달할 때 사용한다.

그림 5는 LookupStage 클래스와 관련 클래스들의 관계 다이어그램을 간략하게 UML 표기법으로 나타내고 있다.

페이로드에는 전달하고자 하는 메시지가 작성되는데, LookupStage에서는 Payload 클래스 객체를 페이로드로 사용한다. Payload 클래스에는 착신 노드의 IP 주소 요청에 관련된 내용들이 기록된다. LookupStage는 ready라는 이름의 자식 쓰레드를 생성하는데, ready 쓰레드는 소켓 통신을 통하여 임의의 사용자로부터 메일 주소를 수신하고 bambooRouteInit(src_id, dest_id, stage_id, pay) 이벤트를 생성한다. 이렇게 생성된 이벤트는 요청 관련 메시지가 기록된 페이로드를 포함하고, dest_id를 키로 하여 착신 노드의 LookupStage에게 전달된다. 착신 노드의 Bamboo 시스템이 이를 수신하면 bambooRouteDeliever(src_id, dest_id, stage_id, pay) 이벤트를 발생시켜서 해당 스테이지, 여기에서는 LookupStage에게 전달한다. 착신 노드의 LookupStage는 자신의 IP 주소를 페이로드에 기록하여 발신 노드의 LookupStage에게 응답한다.

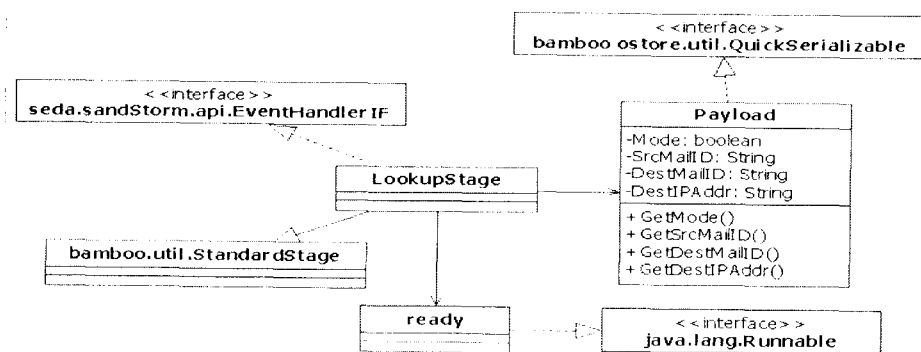


그림 5 LookupStage 클래스와 관련 클래스의 관계 다이어그램

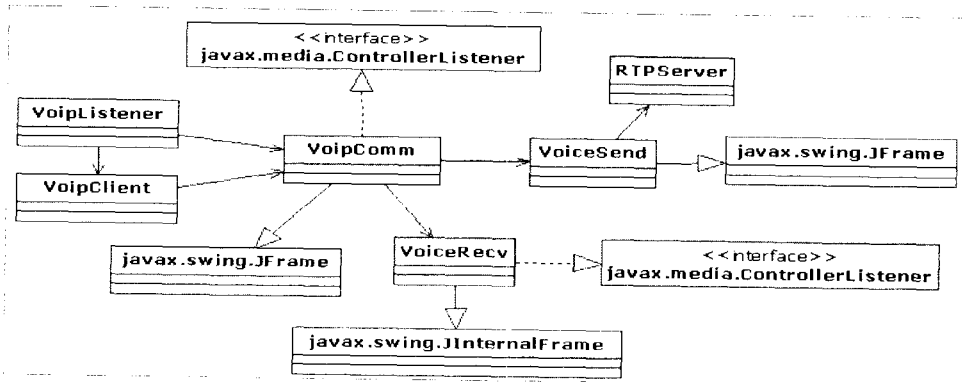


그림 6 음성 송·수신 관련 클래스들의 관계 다이어그램

4.2 음성 송·수신

클라이언트 응용 프로그램 VoipClient는 LookupStage를 사용하여 착신 노드의 IP 주소를 구하고 그 IP 주소에서 실행중인 VoipListener에게 연결 요청 제어 메시지를 전달한다. VoipListener는 연결에 대한 수락 여부를 결정하고 수락할 경우에 VoipClient와 세션을 설립한다. VoipListener는 CALL TO 제어 메시지와 착신 노드의 IP 주소를 수신하면 VoipClient를 실행하여 수신한 주소로 음성 통신 세션 연결 요청 제어 메시지를 전송한다. 웹 페이지의 중개 기능에서 애플릿은 발신 노드의 VoipListener에게 이러한 부류의 메시지를 전송하여 발신 노드가 착신 노드와 세션을 설립하도록 한다. 그림 6은 음성 데이터 송·수신에 관련한 클래스들의 관계 다이어그램을 간략하게 UML 표기법으로 나타내고 있다.

음성 송·수신 세션 설립을 위해 사용하는 제어 메시지는 다음과 같다.

- HELLO VoipListener의 정상적인 동작 여부를 확인할 때 사용
- REQUEST 음성 통신 세션 연결 요청에 사용
- CALL TO 특정 IP 주소로 연결 요청할 것을 지시할 때 사용
- ACCEPT 요청에 대한 수락의 의미로 사용
- REJECT 요청에 대한 거절의 의미로 사용

4.3 중개 페이지

우리의 연구에서는 음성 통신을 위한 중개 기능을 구현하기 위해서 JSP를 사용하였다. JSP 페이지가 임의의 클라이언트로부터 요청되면 웹 서버는 JSP 페이지 내에 기록되어 있는 메일 주소의 소유자의 서비스 로그인 여부를 확인한 후, 로그인되어 있는 사용자의 메일 주소 오른쪽에는 애플릿 아이콘을 추가한 웹 페이지를 작성하여 응답한다. 클라이언트가 로그인되어 있는 사용자와 음성 통

신 세션을 개설하고자 할 경우에는 애플릿 버튼을 클릭한다. 애플릿이 실행되면 애플릿은 HELLO 제어 메시지를 사용하여 발신 노드와 착신 노드에서 VoipListener이 정상적으로 동작하고 있는지를 확인한 후 세션 개설을 돕는다. 그림 7은 JSP 스크립트의 일부를 나타내고 있다.

```

<X
DestIPAddrReq req =
    new DestIPAddrReq("localhost", "3632",
        "alwaysshiny@gmail.com");
if ( req.getState() == true ) {
X>
<a href="mailto:alwaysshiny@dmri.kangwon.ac.kr">
    alwaysshiny@gmail.com
</a>
<applet code="VoiceConn" archive="VoiceConn.jar"
width="20" height="20">
<param name="src_ip" value=
"<X=pageContext.getRequest().getRemoteAddr()X">
<param name="dst_ip" value=
"<X=req.getDestIPAddr()X">
</applet>
<X } else { X>
<a href="mailto:alwaysshiny@dmri.kangwon.ac.kr">
    alwaysshiny@gmail.com
</a>
<X } X>
    
```

그림 7 중개 페이지의 JSP 스크립트 일부

5. 테스트

테스트는 논문에서 중점을 둔 시나리오 2를 기반으로 하여 실행되었다. 시나리오 1은 시나리오 2에 비해 그 실행 및 테스트가 간단하기 때문에 생략하였다. 표 1과 그림 8은 테스트가 이루어진 오버레이 네트워크의 상황과 각 노드의 시스템 정보를 나타내고 있다.

유연성을 갖는 분산 해쉬 테이블 기반의 피어 투 피어 인터넷 텔레포니 서비스의 구현

표 1 각 노드의 시스템 정보

노드	시스템 환경
A	OS: Windows XP, JDK : 1.5.0_06
B	OS: Fedora Linux - 4, JDK : 1.5.04
C	OS: Windows XP, JDK : 1.5.0_06
D	OS: Debian Linux - Woody, JDK : 1.5.04
E	OS: Gentoo Linux - 2004.2, JDK: 1.5.04, JSP/Servlet Engine : Tomcat 5
F	OS : Fedora Linux - 4, JDK : 1.5.04

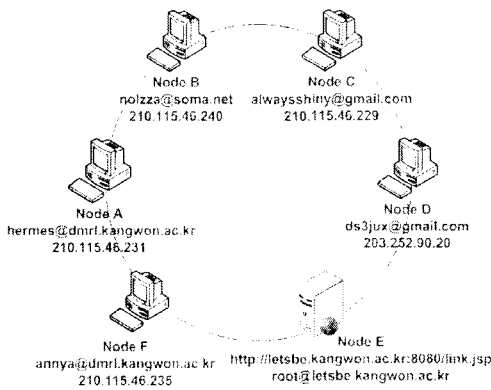


그림 8 테스트 오버레이 네트워크

테스트 과정은 다음과 같다.

- ① 노드 A는 웹 서버 노드 E에게 <http://letsbe.kangwon.ac.kr:8080/link.jsp> 페이지를 요청한다.
- ② 노드 E는 Bamboo 록업을 사용하여 웹 페이지 내에 기록되어 있는 메일 주소들의 IP 주소들을 검색한 후에 검색에 성공한 이메일 주소 오른쪽에는 애플릿 버튼을 추가하여 웹 페이지를 작성하고 이를 응답한다.
- ③ 노드 A는 웹 페이지에서 alwaysshiny@gmail.com 오른쪽의 애플릿 버튼을 클릭하여 음성 통신 세션 설립을 요청한다.
- ④ 노드 C는 노드 A로부터 음성 통신 세션 설립 요청을 수신하고 이를 수락하여 세션을 설립한다.

그림 9는 노드 A의 웹 브라우저에 출력되는 화면과 음성 통신이 이루어지는 상황을 보여준다. 이 그림에는 예로서 특정 인물들의 프로필을 나타내고 있다. 각각의 프로필에는 이메일 주소가 포함되어 있는데, 웹 페이지가 요청될 때 프로필에 기록되어 있는 이메일 주소를 사용하여 서비스에 로그인한 사용자의 경우에, 추가적으로 애플릿 아이콘

이 이메일 주소의 오른쪽에 보여진다. 웹 브라우저 오른쪽의 윈도우는 웹 페이지를 요청한 클라이언트 노드 A가 alwaysshiny@gmail.com 메일 주소의 사용자와 세션을 개설하고 실질적으로 음성 데이터를 송·수신하는 것을 나타낸다.

애플릿 아이콘은 이 웹 페이지가 보여질 때마다 갱신된다. 웹 서버는 아이콘 갱신을 위해서 Bamboo 록업을 사용하여 사용자의 로그인 여부를 확인한다.

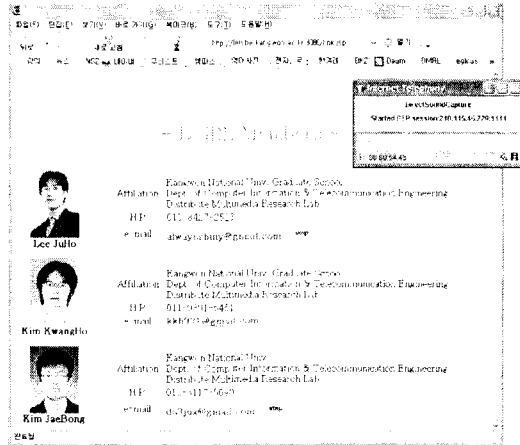


그림 9 실행 테스트 결과

6. 결론

본 논문에서는 피어 투 피어 인터넷 텔레포니 서비스가 기존의 중앙 집중형 아키텍처 기반의 서비스가 가지던 많은 한계점들을 해결하고, 나아가 더욱 다양한 서비스로 유연하게 확장 및 통합 발전할 수 있다는 가능성에 대해서 연구하였다.

이를 위해 우리는 Bamboo 오버레이 네트워크 기반의 인터넷 텔레포니 서비스를 구현하였고 서비스의 유연한 확장을 입증하고자 웹 기반의 인터넷 텔레포니 서비스로 확장하여 이를 테스트하였다.

웹 기반의 인터넷 텔레포니 서비스는 WWW(World Wide Web) 사용자들로 하여금 비실시간 이메일 서비스에 추가적으로 실시간 음성 통신 서비스를 사용할 수 있게 하였다.

앞으로도 우리는 다양한 서비스들의 추가적인 확장 기능 및 통합에 대하여 연구를 진행할 것이다.

참 고 문 헌

- [1] Napster - <http://www.napster.com/>.
- [2] Gnutella - <http://gnutella.wego.com/>
- [3] Kundan Singh, Henning Schulzrinne: "Peer-to-peer internet telephony using SIP". NOSSDAV 2005: 63-68
- [4] Bamboo - <http://www.bamboo-dht.org/>
- [5] Sean Rhea, Dennis Geels, Timothy Roscoe, and John Kubiawicz. Handling Churn in a DHT. Proceedings of the USENIX Annual Technical Conference, June 2004.
- [6] S. Ratnasamy, P. Francis, M. Handley, R. Karp and S. Shenker: A Scalable Content-Addressable Network, Proc. of ACM SIGCOMM 2001.
- [7] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications", in Proceedings of ACM SIGCOMM'01, San Diego, September 2001
- [8] A. Rowstron and P. Druschel. Pastry: Scalable distributed object location and routing for large-scale peer-to-peer systems. In Proc. MIDDLEWARE'01.
- [9] Sean C. Rhea. "A Programmer's Tutorial on Event-Driven Programming, Asynchronous Input/Output, and the Bamboo DHT" December 15, 2005
- [10] Marcel Dischinger. "Bamboo - A Tutorial" April 2004
- [11] M. Dischinger, "A flexible and scalable peer-to-peer multicast application using Bamboo", Report of the University of Cambridge Computer Laboratory, 2004
- [12] <http://java.sun.com/products/java-media/jmf/index.jsp>
- [13] Skype : <http://www.skype.com>
- [14] Salman A. Baset and Henning Schulzrinne, "An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol", IEEE Infocom 2006