효과적인 협업지원을 위한 Jabber 메시징 시스템의 확장

이 근 웅[†] · 안 건 태[†] · 황 의 윤[†] · 김 진 홍[†] · 이 명 준[††]

요 약

BioPlace 시스템은 유전체 관련 연구자들의 효과적인 정보교환과 연구 활동을 지원하기 위한 웹 기반 협업지원 시스템이다. BioPlace 시스템은 웹상의 가상공간인 개인작업장과 팀작업장을 통하여 그룹 멤버들간의 협업을 지원하고 있다. Jabber 메시징 시스템은 XML 기반의 개방형 메시징 시스템으로서 실시간 의사소통을 지원하기 위한 다양한 기능과 타 메시징 시스템과의 연동을 제공하고 있어서 능률적인 메시징 서비스의 개발을 지원한다. 본 논문에서는 Jabber 실시간 메시징 시스템을 확장하여 BioPlace 협업지원 시스템에서 실시간 의사소통 수단으로 사용될 수 있도록 Jabber XML 프로토콜을 설계하였다. 또한, 개발된 프로토콜을 기반으로 확장된 Jabber 서버와 BioPlace 메신저 클라이언트 시스템을 개발하였다.

Extending Jabber Messaging System for Effective Collaboration

Keon-Woong Lee[†] · Geon-Tae Ahn[†] · Eui-Yoon Hwang[†]
Jin-Hong Kim[†] · Myung-Joon Lee[††]

ABSTRACT

The BioPlace system is a web-based collaborative system which offers effective exchange of information and research activities among the genomic researchers. BioPlace supports collaborative works among the related group members through *Personal Workspaces* and *Team Workspaces* which are virtual spaces on the web. Jabber is an open messaging system based on XML, giving an efficient way of developing messaging services by providing various functionality for real-time communication and interoperability with other foreign messaging systems. In this paper, we have designed additional Jabber XML protocol to extend Jabber messaging system to be used as real-time communication methods on the BioPlace collaborative system. Also, according to the protocol, we have developed both the extended Jabber server and the BioPlace messenger client.

키워드 : BioPlace, Jabber 메시징(Jabber Messaging), XML 기반 통신(XML-based Communication)

1. 서 론

BioPlace 협업지원 시스템[1]은 유전체 관련 연구자들에게 유전체 데이터베이스에 대한 검색기능을 제공하고, 특정 연구 분야의 연구자들 사이에 연구 관련 정보를 효과적으로 교환하고 공유할 수 있는 수단을 제공하는 웹기반 공동작업 환경이다. BioPlace 시스템에서는 연구자의 개인 정보 관리를 위한 개인작업 공간과 관련 연구자들의 공동작업에 필요한 공동 데이터 관리를 위한 공동작업 공간을 통하여 유전체 연구 관련 정보의 관리 및 공유를 실현하고 있다. 현재 BioPlace 시스템이 정보를 교환하는 방법은 대부분 비동기적이며, 실제 공동작업의 과정에서 발생할 수 있는 여러 가지 전달사항이나 수시로 변하는 정보를 실시간적으로 전달할 수 있는 방법은 제공하고 있지 않다. Jabber 메시징 시스템[2, 3]은 이러한 문제를 효과적으로 해결할 수 있는 방법을 제시해 준다.

Jabber 메시징 시스템은 실시간 메시징을 위한 XML 기반의 데이터 모델과 프로토콜에 대한 구현이다. Jabber 시스템은 오픈소스 프로젝트로서 일반적인 메시징 시스템이 제공하는 기본적인 기능을 충실하게 지원하면서 다양한 실시간 메시징 시스템과의 통신 기능까지 제공함으로써 보다 능률적인 메시징 서비스를 개발 할 수 있는 방법을 제공해 준다.

본 논문에서는 공동작업의 효율성을 높이기 위한 방법으로 BioPlace 협업지원 시스템에 실시간 메시징 기능을 지원하기 위하여 Jabber 메시징 시스템을 확장한 BioPlace 메신저 시스템을 설계하고 구현하였다. BioPlace 시스템의 모든 사용자는 특정 부서에 소속되어 있으며, 이후에 특정 작업을 위한 팀을 생성하는 경우, 하나 이상의 팀작업장에 소속된다. Jabber에는 이러한 개념을 표현하기 위한 XML 프로토콜이 존재하지 않으므로 이러한 정보를 표현하기 위한 XML 프로토콜을 먼저 설계하였다. BioPlace 시스템과 Jabber 메

※ 이 논문은 2003년 울산대학교의 연구비에 의하여 연구되었음.
† 준 회 원 : 울산대학교 대학원 컴퓨터정보통신공학부
†† 정 회 원 : 울산대학교 컴퓨터정보통신공학부 교수
논문접수 : 2003년 3월 5일, 심사완료 : 2003년 8월 1일

시징 시스템을 연동시키기 위하여 Jabber 서버에서 BioPlace로의 인증을 담당해 주는 컴포넌트와 BioPlace의 사용자 부서 및 팀작업장 정보를 Jabber 클라이언트측으로 전달해 주기 위한 컴포넌트를 추가 구현 구현하였다. 또한, BioPlace에서 사용되는 사용자 부서 및 팀작업장에 대한 정보의 표현을 지원하는 추가적인 Jabber 프로토콜과 확장된 Jabber 서버와 연동하기 위한 사용자 중심의 메신저 클라이언트를 개발하였다.

메신저 클라이언트 개발에는 Jabber 클라이언트 개발을 용이하게 지원하는 *Matrix COM* 동적 *라이브러리*[4,5]를 이용하였으며, 클라이언트가 확장된 Jabber 프로토콜을 지원할 수 있도록 Matrix Com 라이브러리도 추가 수정되었다. 메신저 클라이언트는 기존의 Jabber 인터페이스에 BioPlace의 사용자 부서 및 팀작업장에 대한 정보 등 BioPlace 시스템과의 연동을 위한 기본정보에 추가된 확장 인터페이스를 제공한다.

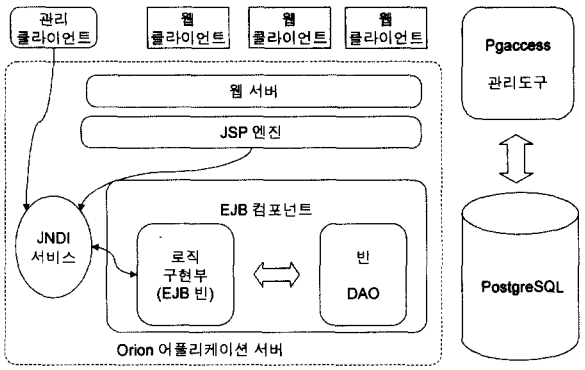
본 논문의 구성은 다음과 같다. 1장 서론에 이어, 2장에서는 BioPlace 협업지원 시스템과 대표적인 메시징 시스템 개발 플랫폼인 *JXTA*[6-8] 및 Jabber 아키텍처에 대하여 기술한다. 3장에서는 BioPlace 메신저 시스템의 구조 및 확장된 Jabber XML 프로토콜의 설계에 관하여 기술한다. 4장에서는 BioPlace 메신저 시스템의 서버와 클라이언트의 구현에 대하여 설명하고, 마지막 5장에서는 결론 및 향후 연구과제에 대하여 기술하고자 한다.

2. 관련 연구

본 논문에서는 BioPlace 메신저 시스템의 개발과 관련하여 대표적인 P2P[9] 플랫폼인 JXTA와 Jabber 프로토콜을 여러 가지 측면에서 상호 비교하였다. 본 장에서는 BioPlace 협업지원 시스템과, JXTA, 그리고 Jabber 프로토콜에 대하여 기술한다.

2.1 BioPlace

BioPlace 협업지원 시스템은 유전체 관련 연구자들의 효과적인 정보교환 및 사용자 중심의 유전체 연구 관련 소프트웨어를 지원하여 공동작업 환경을 제공하는 웹 기반 협업지원 시스템으로서 EJB(Enterprise JavaBeans)[10] 컴포넌트와 JSP(JavaServer Pages)[11] 기술을 기반으로 제작되었다. BioPlace 시스템은 웹상의 개인적인 공간인 개인작업장과 팀의 멤버가 공유하는 공간인 팀 작업장을 통하여 멤버들 사이의 정보 공유와 재사용을 효과적으로 지원한다. 또한 BioPlace 시스템은 J2EE 플랫폼을 통하여 서비스를 지원하고 EJB 기술을 이용하여 개발함으로써 서버의 확장성을 높이고 안정적인 서비스를 제공한다. 특히 플랫폼에 독립적인 유전체 관련 공개 소프트웨어를 자유롭게 설치 확장 가능하여 서버 개발의 비용을 줄일 수 있는 장점을 가지고 있다. (그림 1)은 BioPlace 시스템의 전체적인 구조이다.



(그림 1) BioPlace 협업지원 시스템의 구조

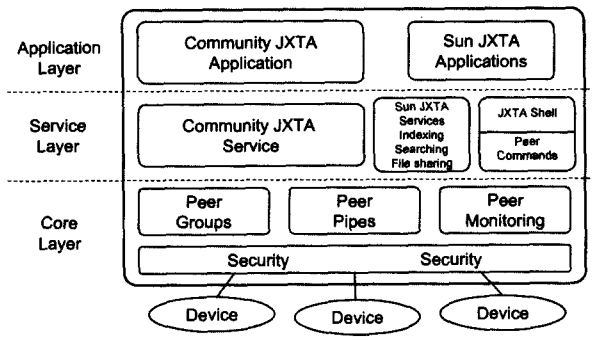
BioPlace 시스템은 웹 클라이언트와 EJB 서버 컴포넌트, 그리고 데이터베이스의 3계층으로 이루어져 있다. 웹 클라이언트 계층은 사용자 요청을 받아들이는 JSP/Servlet의 서버 스크립트 언어로 구현되었으며, EJB 컴포넌트로 구성된 서비스 계층이 클라이언트의 요청을 받아들여, 비즈니스 로직을 수행한다. Orion 어플리케이션 서버[12]에서 제공하는 EJB 컨테이너를 사용하고 있으며, 비즈니스 로직 구현에 필요한 데이터의 일관성 유지를 위해서는 공개용 데이터베이스 시스템인 PostgreSQL[13]을 사용하였다.

2.2 JXTA

JXTA[6]는 오픈소스로서 P2P 서비스와 관련 어플리케이션 개발을 위한 프로젝트이며, XML 프로토콜을 사용한다. JXTA는 설계에 있어서 Microsoft사의 닷넷에 비해 단순함을 추구하였으며, 오픈소스 커뮤니티 모델로 유동적인 환경에서 구축된다는 점에서 구별된다. JXTA는 현존하거나 개발 중인 많은 P2P 시스템을 보다 간편하게 개발할 수 있는 방법을 제공하고 있다.

JXTA 시스템의 대표적인 특징으로는 시스템간의 상호 *운용성(Interoperability)*, *플랫폼 독립성(Platform Indefendance)*, 그리고 *편재성(Ubiquity)*을 들 수 있다. 기술적인 측면에서 JXTA는 전형적인 P2P 소프트웨어 명세(Specification)를 3계층으로 구분하고 있다. (그림 2)는 JXTA 시스템의 계층 구조를 도식화한 것이다. 제일 아래 계층인 핵심계층(Core Layer)은 라우팅과 같은 peer 확립과 통신 관리를 담당한다. 핵심계층 위에 존재하는 서비스 계층(Service Layter)은 인덱싱(indexing)과 검색, 파일 공유 등의 기능을 수행하는 보다 높은 레벨의 개념을 다룬다. 그리고 제일 상위 계층인 응용 계층(Application Layer)에는 전자메일이나 스토리지 시스템과 같은 어플리케이션이 존재한다. JXTA 기술은 서비스와 애플리케이션이 구축된 최상위에 하나의 계층을 제공하도록 설계되었다. 이 계층들은 서비스와 어플리케이션에서 사용하는 여러 가지 기능을 제공하면서도 가능한 간단하게 설계되어 있다. 이 계층을 이와 같이 설계한 이유는 다른 *P2P Contributor*와는 다른 경쟁력을 지닌 상태에서 상호 운용성을 지원하면서 개발자들이 개선할 만한 여지

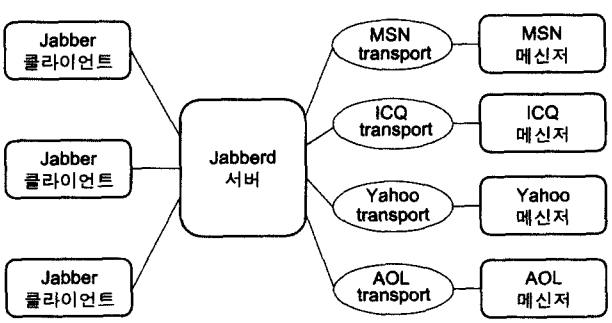
가 최대한 제공되도록 하기 위해서이다.



(그림 2) 확장된 Jabber 서버 시스템의 구조

2.3 Jabber

Jabber는 JXTA와 더불어 XML 프로토콜을 사용하는 대표적인 P2P 기반의 메시징 시스템 개발 플랫폼이다. 1998년 최초로 서버개발을 시작한 이래 현재 적어도 백만 여명 이상이 이 기술에 관여되어 있다. Jabber는 메신저 시스템을 효과적으로 개발하기 위한 몇 가지 특징을 가지고 있다. 첫째, Jabber는 XML 프로토콜을 사용하며, JXTA와는 다른 클라이언트·서버 구조를 가진다. 둘째, 여러 대의 서버를 유지하는 분산 네트워킹 기법을 사용하며, 대표적인 Jabber 서버는 컴포넌트의 추가, 삭제가 쉬운 모듈구조로 구성되어 있다. 마지막으로 Jabber는 간편한 클라이언트 제작을 지원한다는 것이다. 또한, Jabber에는 각종 *트랜스포트(Transport) 컴포넌트*[14]를 통하여 다른 메신저 시스템에서 보내온 메시지를 Jabber의 사용자에게 보내주기도 하고, 반대로 Jabber에서 보내는 메시지를 다른 메신저 시스템으로 전달해 주는 기능도 제공한다. (그림 3)은 각종 메시징 시스템과 Jabber 서버와의 연계를 보여주고 있다.



(그림 3) Jabber 서버와 다른 메신저와의 연계

본 논문에서 웹 기반 메시징을 구현하기 위한 기본 아키텍처로 Jabber를 선택한 이유는 다음과 같다. 첫째, JXTA는 응용프로그램들의 대부분이 Java 어플리케이션으로 작성되어 있어 가상머신에 대한 오버헤드가 따른다는 단점이 있으나 다양한 플랫폼의 응용프로그램들이 개발되고 있어 보다 활용에 유리한 면이 있다. 둘째, 복잡한 구조로 작성된 JXTA에 비해 배우기가 쉬우면서 확장에 유리한 XML 프로토콜을 제공하고 있다는 점이다.

본 논문에서 웹 기반 메시징을 구현하기 위한 기본 아키텍처로 Jabber를 선택한 이유는 다음과 같다.

첫째, JXTA는 응용프로그램들의 대부분이 Java 어플리케이션으로 작성되어 있어 가상머신에 대한 오버헤드가 따른다는 단점이 있으나 다양한 플랫폼의 응용프로그램들이 개발되고 있어 보다 활용에 유리한 면이 있다.

둘째, 복잡한 구조로 작성된 JXTA에 비해 배우기가 쉬우면서 확장에 유리한 XML 프로토콜을 제공하고 있다는 점이다. Jabber는 그 기능면에 있어서 JXTA보다 다양하지 못한 것은 사실이지만 본 논문의 목적인 협업 지원을 위한 실시간 메시징을 구현하기에는 충분한 프로토콜을 제공하고 있어서, 오히려 JXTA와 같은 복잡한 프로토콜에 대한 연구는 비효율적이라고 할 수 있다.

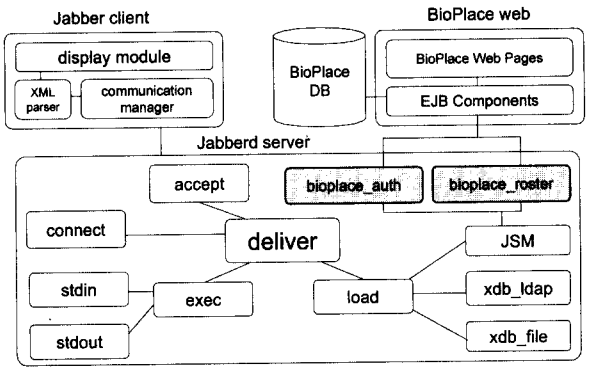
셋째, Jabber에서는 추가적인 기능을 구현하기 쉬운 모듈 구조로 된 대표적인 Jabber 서버를 제공하고 있다. 사실, Jabber나 JXTA 자체는 위에서 설명하였듯이 XML 프로토콜로 이루어져 있으므로 사용자가 완전한 메시징 시스템을 구축하기 위해서는 각각의 프로토콜을 기반으로 하는 서버/클라이언트 혹은 P2P 어플리케이션을 제작하여야 하지만 Jabber에는 이미 잘 만들어진, 대표적인 서버 시스템인 Jabber 서버를 제공하고 있다. 마지막으로 Jabber는 기존 타 메신저 시스템과의 호환을 지원하고 있다. 본 논문 이전에도 협업지원 시스템에서 실시간 의사소통을 지원하기 위한 도구가 개발된 적이 있었으나 사용자들은 기존에 사용하던 메시징 시스템을 포기하거나 새로운 메시징 시스템을 추가로 사용하는 것에 대한 부담으로 인하여 사용률이 저조하게 나타난 경우가 있었다[15]. 하지만 Jabber에서는 대표적인 메시징 시스템인 MSN, AIM, AOL, Yahoo 등의 사용자들과의 메시지 교환을 *트랜스포트*라는 컴포넌트를 통하여 지원하고 있어 이러한 문제점을 해결하였다.

3. BioPlace 메신저 시스템의 설계

3.1 시스템 구조

본 논문에서 구현한 BioPlace 메신저 시스템은 Jabber 서버를 기반으로 하고 있으며, 확장이 편리한 컴포넌트 구조를 하고 있다. 이들 컴포넌트 사이에 데이터 흐름을 제어함으로써 메시징 서비스를 구현하고 있다. 따라서 BioPlace 시스템과 같이 다른 시스템과의 연동을 위해서는 추가적인 컴포넌트의 제작이 필요하다. (그림 4)에서 보는 바와 같이 BioPlace 시스템과의 연동을 위하여 Jabber 서버의 *JSM (Jabber Session Manager)* 하부 모듈인 bioplace_auth와 bioplace_roster 컴포넌트를 확장 구현하였다. JSM은 사용자의 세션에 관련된 각종 이벤트를 다루는 중요한 역할을 담당하므로 타 시스템과의 연동을 위해서는 이 컴포넌트의 확장이 요구된다. (그림 4)는 BioPlace와의 연동을 위하여

새로운 컴포넌트가 추가된 확장 Jabber 서버의 구조를 나타낸 것이다.

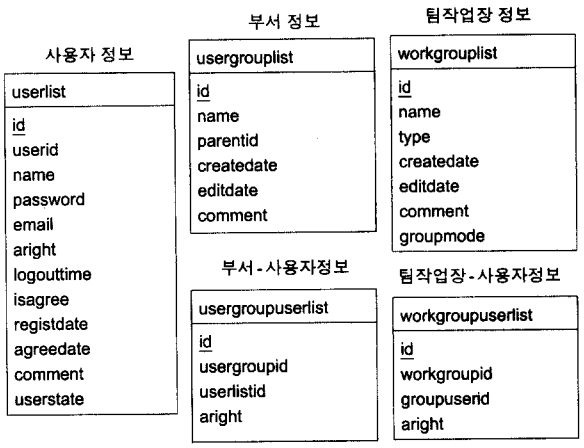


(그림 4) 확장된 Jabber 서버 시스템의 구조

(그림 4)에서 추가된 컴포넌트를 보면, 우선 Jabber 서버 자체는 BioPlace의 사용자 및 내부 정보를 읽어 들일 수 있는 방법이 존재하지 않는다. 따라서, JSM로부터 BioPlace의 인증에 필요한 정보를 BioPlace에서 읽어온 후, Jabber 사용자 로그인과는 구분되는 BioPlace 사용자 로그인을 담당해 주는 bioplace_auth라는 컴포넌트를 추가하였다. bioplace_auth 컴포넌트는 JSM 컴포넌트를 거쳐 보내진 Jabber 클라이언트의 정보가 담긴 XML 패킷을 분석하여 이 중 ID 및 패스워드에 해당하는 정보를 추출해낸 후, BioPlace의 컴포넌트에 전달한다. 그러면 BioPlace 시스템의 EJB 컴포넌트는 BioPlace에서 사용자 인증에 관련된 ID 및 패스워드 정보를 읽어 들인 다음 인증의 성공 여부를 결정한다. EJB 컴포넌트는 그 결과를 다시 bioplace_auth 컴포넌트로 전달하여 해당 유저를 로그인 시키거나 거부할 수 있도록 한다. 또 하나의 추가된 컴포넌트인 bioplace_roster는 특정 사용자가 BioPlace의 사용자로 Jabber에 로그인하게 되면 기존 Jabber의 친구(roster) 정보 이외에 BioPlace 내의 모든 부서 및 팀작업장의 정보를 보내준다. 따라서 bioplace_roster 컴포넌트에서는 BioPlace의 부서 및 팀작업장 정보를 데이터베이스에서 읽어온 후 이것을 Jabber 클라이언트에서 인식 가능한 XML 문서의 형태로 변환하는 역할을 담당한다. 이렇게 변환된 XML 문서를 클라이언트에 전송하면 클라이언트에서는 전송된 정보를 기반으로 트리형식의 사용자 인터페이스를 제공해주게 된다.

3.2 데이터베이스 구조

BioPlace 시스템에서는 자체적인 데이터베이스를 구축하기 위하여 데이터베이스 시스템인 PostgreSQL을 사용하였다. BioPlace의 수많은 데이터베이스 테이블 중에서 본 논문에서 구현된 BioPlace 메신저 시스템과 관련된 테이블은 사용자 정보에 관련된 userlist 테이블, 부서정보에 관련된 usergrouplist 및 usergroupuserlist 테이블, 그리고 팀작업장 정보에 관련된 workgrouplist 및 workgroupuserlist 테이블이다. (그림 5)는 BioPlace의 데이터베이스 테이블 중에서 BioPlace 메신저 시스템과 관련된 테이블의 구조를 나타낸 것이다.



(그림 5) 메시징 시스템과 관련된 BioPlace의 데이터베이스 테이블

userlist 테이블 내의 id 필드는 데이터베이스에서 프라이머리키(Primary Key)로 사용되는 식별자이며, userid와 password는 BioPlace에 로그인하기 위한 사용자의 ID와 패스워드를 나타낸다. usergrouplist의 name 필드에는 실제 BioPlace의 부서 이름이 저장된다. usergroupuserlist 테이블에서 usergroupid는 사용자 부서의 식별자로서 usergrouplist 테이블의 id 필드와 관계를 맺고 있다. userlistid는 사용자에 대한 식별자로서 userlist 테이블의 id 필드와 관계를 맺고 있다. 이렇게 usergroupuserlist 테이블에는 한 쌍의 사용자 부서와 사용자의 id를 저장하여, 해당 사용자가 해당 부서에 소속되어 있음을 나타내 준다.

3.3 확장된 Jabber XML 프로토콜의 설계

BioPlace에서 각각의 사용자는 하나의 부서와 여러 개의 팀작업장에 동시에 소속될 수 있다. Jabber 서버에서는 (그림 6)과 같이 <item></item> 태그를 사용하여 등록된 사용자를 표현하며, <group></group> 태그를 사용하여 사용자들의 분류를 나타낼 수 있다. xmlns 및 xdbns는 Jabber에서 일어나는 이벤트에 대한 유형을 정의한 NameSpace를 나타낸다. 각각의 item 태그는 등록되거나 등록 예정인 한 명의 사용자를 나타낸다. item 태그 안에는 group 하부태그가 존재하는데 이 태그를 사용하는 경우, 클라이언트 인터페이스에서는 해당 유저가 특정 그룹 안에 포함되어 있음을 나타내준다. (그림 6)은 "my friends"라는 그룹에 tommy와 billy라는 사용자가 존재하고 있음을 의미한다.

(그림 6)의 XML을 살펴보면 Jabber에서 한 사용자는 여러 개의 group에 등록되는 것이 불가능함을 알 수 있다. 따라서 BioPlace 협업지원 시스템 상에서 사용되는 부서·팀작업장

의 정보를 올바르게 나타내 주기 위해서는 새로운 Jabber XML 스키마가 필요하다. (그림 7)은 다중 그룹 정보를 Jabber에서 지원할 수 있도록 기본 XML을 확장한 예이다.

```
< query xmlns = 'jabber : iq : roster' xdbns = 'jabber : iq : roster' >
    < item jid = 'tommy@203.250.77.118' name = 'tommy'
        subscription = 'to' > < group > my friends </ group >
    </ item >
    < item jid = 'billy@203.250.77.118' name = 'billy' subscription = 'to' >
        < group > my friends </ group >
    </ item >
        <item jid = 'guest@203.250.77.118' subscription = 'from' />
</ query >
```

(그림 6) Jabber XDB 파일에서 roster 정보 표현의 예

```
< query xmlns = 'jabber : iq : bioplace_roster'
        xdbns = 'jabber : iq : bioplace_roster' >
    < item jid = 'tom@203.250.77.118' name = 'tom' subscription = 'to' >
            < group > my friends </ group >
            < type > usergroup </ type >
    </ item >
    < item jid = 'billy@203.250.77.118' name = 'billy' subscription = 'to' >
            < group > my friends </ group >
            < type > usergroup </ type >
    </ item >
    < item jid = 'Tom@203.250.77.118' subscription = 'from' />
            < group > study group </ group >
            < type > workgroup </ type >
    </item>
</query>
```

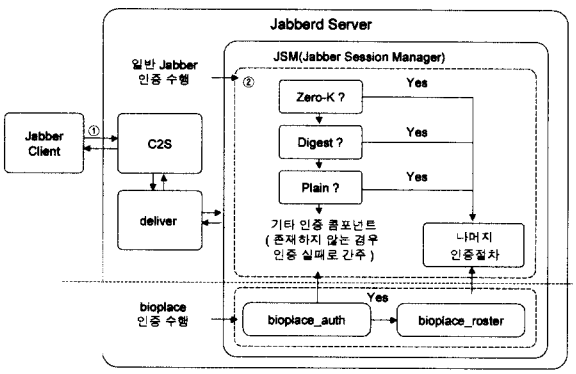
(그림 7) 부서 및 작업팀 정보의 표현을 위한 XML 문서의 예

BioPlace 사용자들은 최초 접속시 Jabber 서버의 일반 사용자 목록을 전송 받은 후, 추가적으로 소속 부서, 팀작업장의 정보를 전송받게 된다. (그림 7)의 XML 문서에서는 jabber : iq : bioplace_roster라는 xml NameSpace를 추가적으로 사용하였으며, 사용자 부서 및 팀작업장의 구별을 위하여 < type ></ type > 태그도 사용하고 있다. 이 항목이 usergroup이면 사용자 부서를 나타내며, workgroup이면 팀작업장을 나타낸다.

4. BioPlace 메신저 시스템의 구현

4.1 서버 구현

BioPlace 메신저 시스템과 BioPlace의 연동을 위하여 Jabber 서버에 추가한 모듈은 Jabber 클라이언트의 BioPlace 인증을 담당하는 bioplace_auth 컴포넌트와 BioPlace의 부서 및 작업장 정보를 클라이언트로 전달하기 위한 bioplace_ roster 이다. 이들은 모두 JSM의 하부 컴포넌트로서 구현되었는데, JSM의 컴포넌트들은 모두 jabber.xml 내의 < load main = "jsm" /> 엔트리에 등록되어 있다. 이 중에서 mod_auth_0k, mod_auth_digest, 그리고 mod_ auth_plain은 Jabber의 사용자 인증에 관련된 컴포넌트로서 이들의 동작 과정은 (그림 8)에서 잘 나타나 있다. (그림 8)의 ①을 보면 클라이언트에서 Jabber 서버로 전송되는 모든 메시지는 먼저 C2S 컴포넌트를 통해 접수된다. C2S는 전달받은 메시지의 유형에 따라 어느 컴포넌트로 전송할 것인지를 결정하게 되는데 사용자 인증에 관계된 패킷은 JSM으로 전달된다. JSM은 전달된 패킷의 내용을 분석하여 자신의 하부 컴포넌트들 중에서 인증에 관련된 것을 가려낸 다음, 해당 컴포넌트에게 순차적으로 전송한다. (그림 8)에서는 먼저 Zero-K(mod_auth_0k) 컴포넌트에게 전송된 후, digest(mod_auth_digest), plain(mod_auth_plain)로 하여금 차례로 검사하도록 하는데, 이중 하나라도 성공하는 경우, 나머지 검사를 건너뛰고 즉시 agent 정보 및 roster 정보 등을 교환하는 정식 인증 과정을 수행하게 된다.



(그림 8) Jabberd의 모듈별 인증 과정

반대로 모든 인증 컴포넌트에서 인증이 실패하여 더 이상 검사할 필요가 없다고 판단되는 경우 인증실패로 간주하여 C2S로 하여금 클라이언트에게 인증실패 사실을 통보하도록 한다. (그림 8)의 ②는 사용자가 BioPlace 인증을 요청한 경우, Jabber 서버의 모듈별 동작과정을 나타낸 것이다. 먼저 클라이언트 측에서 보내온 ID와 패스워드를 bioplace_auth 컴포넌트가 검사한다. 이과정이 성공할 경우, bioplace_roster는 BioPlace의 사용자 부서 및 팀작업장의 정보가 추가된 XML 프로토콜을 사용하여 전송을 해주며, 이후 과정은 일반적인 Jabber 클라이언트를 처리하는 과정과 동일하게 수행된다. 하지만 일반 Jabber에서 사용되는 사용자 ID와 BioPlace에서 사용되는 사용자 ID가 중복되는 경우 문제가 발생할 수 있다. 따라서 (그림 9)와 같이 최초 클라이언트 측에서 인증을 요청하는 Jabber XML 프로토콜의 NameSpace에 "jabber : iq : bioplace_auth"를 추가적으로 기술하여 JSM에서 bioplace_auth 컴포넌트가 인증을 수행하도록 하였다.

```
< query xmlns = "jabber : iq : bioplace_auth" >
    < username > tommy </ username > <! -- BioPlace User ID -- >
    < password > tommy </ password > <! -- BioPlace Password -- >
</ query >
```

(그림 9) BioPlace 인증을 요청하기 위한 XML 프로토콜

다음에서는 본 논문에서 확장한 Jabber 서버에서 BioPlace 사용자로서의 인증을 수행하기 위한 XML 스트림의 흐름을 세부적인 동작과정과 함께 기술한다.

4.1.1 최초 클라이언트 접속시 XML 스트림 개설

(그림 10)은 처음 클라이언트가 접속 시 설정되는 서버와 클라이언트의 XML 부분을 표현하고 있다. 스트림 개설 후 인증과정이 수행된다.

```
server : < stream : stream xmlns : stream = 'http://etherx.jabber.org/streams'
          id = '3DA6393F' xmlns = 'jabber : client' from = '203.250.77.118' >

client : < stream : stream to = "203.250.77.118" xmlns = jabber : client
          xmlns : stream = "http://etherx.jabber.org/streams" >
```

(그림 10) 최초 클라이언트 접속시 XML 스트림 개설

4.1.2 유효한 Jabber ID 인지에 대한 검사 요청(인증에 필요한 사항도 같이)

(그림 11) 클라이언트 인증을 위한 클라이언트·서버 메시지 구조 (그림 11)은 유효한 Jabber ID 인지에 대한 검사 요청 시 전송되는 XML의 내용을 표현하고 있다. 다음과 같은 과정을 통하여 BioPlace 메신저 시스템은 인증과정을 수행하게 된다.

```
client : < iq type = "get" id = "JCOM_0" >
            < query xmlns = "jabber : iq : bioplace_auth" >
                < username > tommy </ username >
            </ query >
         </ iq >
server : < iq type = 'result' id = 'JCOM_0' >
            < query xmlns = 'jabber : iq : bioplace_auth' >
                < username > tommy </ username >
                < password />
            </ query >
         </ iq >
```

(그림 11) 클라이언트 인증을 위한 클라이언트·서버 메시지 구조

① js_packet() 함수를 통해 인증을 수행할 컴포넌트로 XML 패킷을 전송
② mod_auth에서 js_user() 호출
③ js_user()에서는 유저 리스트에서 해당 유저를 확인
④ 실패하는 경우 xdb_file에게 해당유저 검색을 요청
⑤ 실패하는 경우 인증실패로 간주
⑥ 각 단계에서 성공하는 경우 해당 유저 JID를 반환
⑦ 유효 ID 검사 결과를 통보(인증 요구 사항과 함께)

4.1.3 클라이언트는 인증 형식에 따라 인증을 요청

```
client : < iq type = "set" id = "JCOM_1" >
            < query xmlns = "jabber : iq : bioplace_auth" >
                < username > tommy </ username >
                < password > tommy </ password >
            </ query > </ iq >
server : < iq type = 'result' id = 'JCOM_1'/>
```

(그림 12) 클라이언트 인증 형식에 따른 전송 및 결과 메시지

① js_mapi_call()에서는 js_mapi_register()를 통해 등록된 event 에 대한 콜백 함수를 수행(e_BIOPLACE_ AUTH)
예) mod_bioplace_auth에서 다음과 같이 이벤트 핸들러 등록 js_mapi_register(si, e_BIOPLACE_AUTH, mod_bioplace _auth, NULL) ;
② mod_bioplace_auth에서 텍스트 비교를 통해 인증 성공 유무 결정
③ 성공하는 경우, 결과를 클라이언트에 전송

일단 Jabber 서버에서 JID를 생성하게 되면 해당 클라이언트의 인증을 허가한다는 의미로 간주할 수 있다. 이제부터 나머지 인증과정을 수행하게 되며, 그 세부 절차는 다음과 같다.

4.1.4 클라이언트는 agent의 목록을 전송 요청

```
client : < iq id = "AGENTLIST" to = "203.250.77.118" type = "get" >
            < query xmlns = "jabber : iq : agents" /> </ iq >
server : < iq id = 'AGENTLIST' to = 'tommy@203.250.77.118/myjabber'
          type = 'result' from = '203.250.77.118' >

            < query xmlns = 'jabber : iq : agents' >
                < agent jid = 'users.203.250.77.118' >
                    < name > JabberUserDirectory </ name >
                    < service > jud </ service >
                    < search /> < register />
                </ agent >
                </ query >
```

(그림 13) agent 목록 요청을 위한 클라이언트·서버 메시지

① js_mapi_call()에서는 js_mapi_register()를 통해 등록된 event에 대한 콜백함수를 수행(e_SESSION).
② mod_agent에서 서버에서 동작중인 agent들의 리스트를 검색하여 반환해 줌
③ 클라이언트에게 서버에서 동작중인 agent 리스트 전송 (JUD)

4.1.5 클라이언트에서 roster 정보 요청

```
client : < iq id = "fullroster_4" type = "get" >
            < query xmlns = "jabber : iq : roster" />
         </ iq >

server : < iq id = 'fullroster_4'  type = 'result'
            from = 'tommy@203.250.77.118/myjabber' >
            < query xmlns = 'jabber:iq:roster'>
                < item jid = 'billy@203.250.77.118' name = 'billy'
                    subscription = 'none' ask = 'subscribe' />
                ... 생략 ...
            </query>
         </iq>
```

(그림 14) roster 정보 요청을 위한 클라이언트·서버 메시지 구조

① js_mapi_call()에서는 js_mapi_register()를 통해 등록된 event 에 대한 콜백함수를 수행(e_SESSION)
② mod_roster에서는 해당 유저의 등록된 roster 정보를

추출하여 반환해 줌

③ 클라이언트에게 roster 정보 전송

일반 roster 정보는 "jabber : iiq : roster"라는 Jabber Namespace를 사용하였으며, BioPlace 메신저 시스템에서는 BioPlace의 부서 및 사용자 정보를 표현하기 위해서 "jabber : iq : bioplace_roster"라는 Namespace 및 추가적인 Jabber XML 프로토콜을 기술하고, 이를 생성하기 위한 bioplace_roster라는 JSM 하부 모듈을 추가적으로 구현하였다.

4.1.6 클라이언트에서 bioplace_roster 정보 요청

```
client : <iq id = "fullroster_5" type = "get">
            <query xmlns = "jabber : iq : bioplace_roster" /> </ iq >

server : < iq id = 'fullroster_5' type = 'result'
            from = 'test2@203.250.77.118/myjabber' >
            < query xmlns = 'jabber : iq : bioplace_roster' >
              < item jid = atom@203.250.77.118 name = 'tom'
                    subscription = 'to'>
                  < group > my friends </ group >
                  < type > usergroup </ type >
              </ item >
                        ... 생략 ...
              < item jid = 'atom@203.250.77.118' subscription = 'from' />
              < group > study group </ group >
                    < type >workgroup</type></item>
                      ... 생략 ...
            </query>
        </iq>
```

(그림 15) bioplace roster 정보를 요청하는 클라이언트 · 서버 측메시지

① js_mapi_call()에서는 js_mapi_register()를 통해 등록된 event에 대한 콜백함수를 수행(e_SESSION)

② bioplace_roster에서는 BioPlace의 부서 및 팀작업장의 정보를 추출

③ 클라이언트에게 BioPlace의 부서 및 팀작업장의 정보 전송

4.1.7 이후의 인증 과정

이상이 본 논문에서 추가한 XML 프로토콜의 흐름과 각종 컴포넌트들의 동작 과정에 대한 기술이다. 이후의 인증 과정 및 세부 이벤트 핸들링은 다음과 같으며, 기존 Jabber 서버와 메신저 클라이언트 시스템과 동일하다.

① 클라이언트가 자신의 presence 정보를 서버로 전송

② 서버는 mod_presence 모듈을 통해 해당 클라이언트의 roster 정보 내의 모든 클라이언트들의 presence 정보를 파악하여 다른 클라이언트로 브로드캐스팅 메시지를 전송

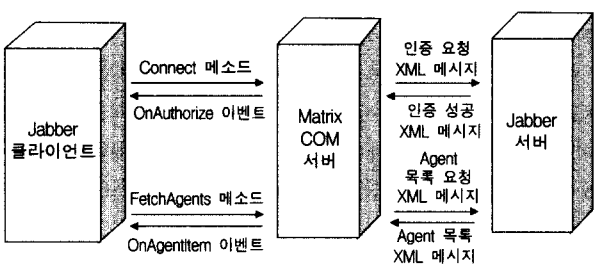
③ 서버측 mod_offline 모듈은 오프라인 중에 수신한 메시지를 클라이언트에게 전송

④ 일반적인 Jabber 시스템의 메시지 처리 및 로그기록 기능 수행

4.2 클라이언트측 구현

구현된 BioPlace의 메신저 클라이언트가 일반 Jabber 메신저 클라이언트와 구별되는 가장 큰 특징은 BioPlace 협업지원 시스템과의 연동을 효과적으로 지원하는 것이다.

BioPlace 메신저 클라이언트의 인터페이스는 협업지원 시스템이 가지고 있는 부서 및 팀작업장의 정보를 효과적으로 나타내 줄 수 있도록 설계되었다. BioPlace 메신저 클라이언트 시스템은 Microsoft Visual Basic 및 Jabber 클라이언트 개발용 라이브러리인 Matrix COM을 사용하여 작성되었다. Matrix COM은 소켓통신 및 각종 XML 메시지의 파싱을 담당하는 컴포넌트이다. (그림 16)은 Matrix COM의 구동방식을 나타낸 것이다. Jabber 클라이언트 프로그램에서 Matrix COM의 메서드를 수행하면 해당하는 동작을 수행한 후 결과를 이벤트 발생을 통하여 Jabber 클라이언트로 통보해 준다. Jabber 클라이언트는 이 이벤트의 종류에 따라 다음에 취해야 할 행동을 결정하여 다시 Matrix COM의 메서드를 수행하는 형태로 클라이언트 프로그램은 구동된다.



(그림 16) Matrix COM의 구동 원리

하지만 Matrix COM에는 일반적인 Jabber 클라이언트의 개발을 위한 기본적인 XML 생성과 파싱 작업만을 수행하도록 되어있다. 따라서 확장된 Jabber 서버와 상호작용하기 위해서는 다음과 같은 추가 확장이 필요하다.

① 새로운 BioPlace 인증 프로토콜을 생성하는 Create_BioPlace_Auth 메서드 추가

② 추가된 XML을 파싱하기 위한 ParseBioPlaceRoster-Items 메서드 추가

③ 부서 및 팀 정보를 표현하고, 보관하기 위한 클래스 모듈 cBioPlaceRosterItem 추가

④ 기타 BioPlace 프로토콜에 관련된 각종 정적 변수 및 이벤트 처리 루틴, 메서드 추가

위와 같은 사항들은 기존 모듈의 확장이므로 기존의 Matrix COM을 사용하는 클라이언트에게 아무런 영향을 끼치지 않으며, BioPlace와 관련된 추가된 내용을 충실히 지원해줄 수 있다.

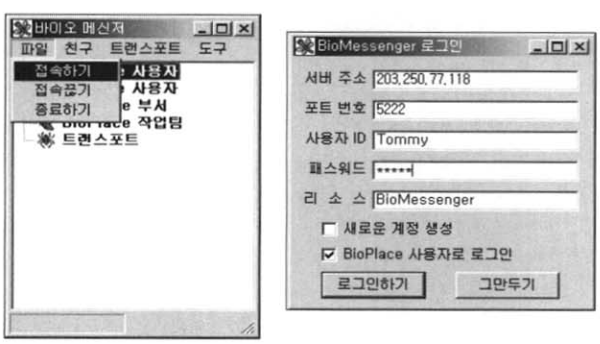
(그림 17)은 BioPlace 메신저의 클라이언트 인터페이스를 나타낸 것이다.

왼쪽 프레임에는 BioPlace 사용자, 일반 Jabber 사용자 및

(그림 17) BioPlace 메신저의 클라이언트 인터페이스

기타 메신저 사용자의 분류와 더불어 Jabber의 각종 서비스인 트랜스포트의 목록을 트리 형태의 인터페이스를 통해 보여준다. 이중 BioPlace 부서 및 작업팀의 트리 밑에는 BioPlace의 부서 및 팀작업장의 정보를 표시해 준다. 사용자는 이 트리에 나타나는 사용자의 이름을 직접 마우스로 클릭하여 대화창을 열어 메시지를 전달할 수 있다.

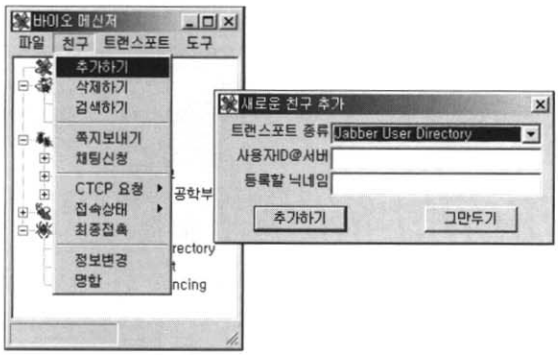
BioPlace 메신저 클라이언트는 일반 Jabber 사용자와 BioPlace 사용자의 두 가지 로그인 옵션을 제공한다. BioPlace 사용자의 경우, 일반 사용자와는 달리 추가로 사용자 부서·팀작업장에 대한 정보를 전달받은 후 트리 인터페이스를 통해 표시해 준다.



(그림 18) 로그인 인터페이스

(그림 18)은 BioPlace 메신저의 로그인 인터페이스를 설계한 것이다. 여기서 서버주소와 포트번호는 접속하고자 하는 Jabber 서버의 주소와 포트번호를 입력하며, 사용자ID 와 패스워드는 Jabber 인증시 필요한 ID와 패스워드를 입력한다. "새로운 계정 생성"을 체크하면 해당 서버에 입력한 내용의 새로운 사용자 계정을 생성하게 되며, "BioPlace 사용자로 로그인" 체크박스는 BioPlace 시스템으로의 로그인을 표시해 주기 위한 것이다.

(그림 19)는 새로운 친구를 추가하기 위한 대화상자인데, 메뉴에서 "친구추가"를 선택하면 된다. 여기에는 트랜스포트의 종류 및 사용자 ID와 서버, 그리고 자신의 인터페이스에서 보이게 될 닉네임을 지정하게 된다.



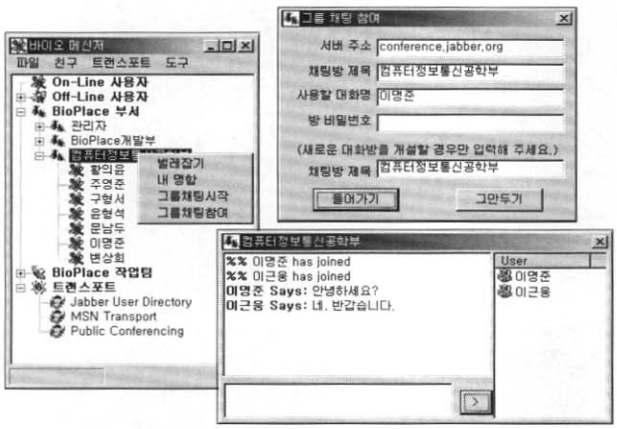
(그림 19) 새로운 친구 추가를 위한 인터페이스

일반적인 Jabber 클라이언트 시스템은 MSN, AOL, AIM, Yahoo 등의 다양한 기존의 메시징 시스템과도 연동하도록 되어 있으며, 본 논문에서 구현한 시스템 역시 이러한 기능을 갖추고 있다. (그림 20)은 MSN 트랜스포트의 속성을 설정 및 사용 예를 보인 것이다.

(그림 20) MSN 트랜스포트 설정 및 사용 예

트리구조의 "트랜스포트" 항목에서 원하는 Transport를 선택한 후, 마우스 우측버튼을 눌러 팝업메뉴에서 "속성설정" 메뉴를 선택하면 트랜스포트 속성 설정 창이 나오는데, 여기에 해당하는 자신의 메신저 계정과 사용할 닉네임을 표시해 준 후, "등록하기" 버튼을 누르면 해당 메신저의 친구정보가 전송되면서 로그인된다. 일단 해당 메신저의 계정으로 로그인 되고 나면, 일반 메신저처럼 채팅을 수행하거나 쪽지를 교환할 수 있다.

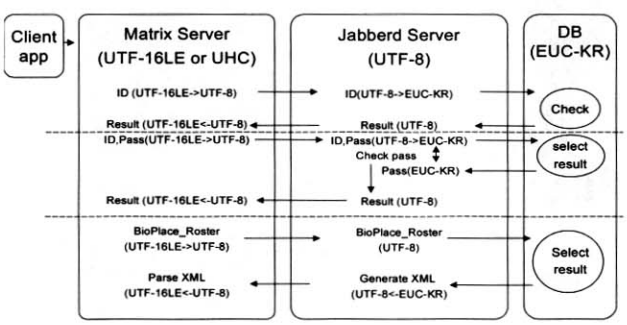
하지만 일부 메신저에만 적용되는 1 : 1 파일 전송이나 환경설정 등의 특수기능은 지원하지 않는다. 메신저 클라이언트는 같은 팀과 부서의 사용자들이 손쉽게 그룹 채팅방을 개설하여 온라인상에서 간단한 그룹미팅 등을 수행할 수 있는 기능을 제공한다. (그림 21)은 BioPlace 메신저 클라이언트에서 사용자 부서별 그룹채팅을 실행한 화면이다.



(그림 21) 사용자 부서별 그룹채팅 실행 화면

4.3 한글 정보의 처리

기본적인 Jabber 서버의 코드체계는 UTF-8이다. 하지만 본 시스템에서 개발한 클라이언트 프로그램은 윈도우즈 응용 프로그램으로 UTF-16LE 혹은 CP949로 불리우는 코드체계를 사용하고 있다. 또한 BioPlace에서 사용하는 Postgre SQL 내에 저장되는 데이터들은 EUC-KR의 코드체계를 사용한다. 따라서 순수 영문으로 저장된 데이터 이외의 특수문자나 한글 데이터들은 호환이 되지 않아서 오류가 발생하는 경우가 종종 발생한다. 더욱이 BioPlace의 사용자 ID는 대부분 한글로 된 것이 많아 BioPlace 사용자로서 로그인을 수행하거나 타 한글 ID 사용자에게 메시지를 전송하기 위해서는 이러한 문제의 해결은 필수적이다.



(그림 22) 한글 데이터의 처리 과정

본 논문에서는 이러한 문제를 해결하기 위하여 클라이언트측에 Matrix COM 내의 UTF-16LE와 UTF-8 사이의 코드체계를 바꿔주는 별도의 모듈을 작성하였다. Jabber 서버에서는 iconv라는 코드체계 변환 라이브러리를 사용하여 작성한 코드체계 변환 함수를 사용하여 클라이언트 측에서 보내온 문자열을 EUC-KR 문자열로 변환하여, 데이터베이스에 질의를 수행한다. 데이터베이스 질의 결과를 한 번 더 UTF-8 문자열로 변환하는 작업을 수행해준 후 이것을 클라이언트 측으로 전송하도록 구현함으로써 시스템 사이에 발생할 수 있는 한글코드 관련 문제들을 해결하였다. (그림 22)는 이러한 과정을 도식화하여 나타낸 것이다. 이러한 과정은 한글 데이터 뿐만 아니라 모든 영문 데이터와 특수문자 데이터에도 해당되며, 기존의 메신저 시스템과도 호환된다.

4.4 메신저 시스템 개발 현황

인터넷 인구의 급격한 확산과 더불어 전자메일과 같은 비동기적인 의사교환 방법보다 실시간적으로 메시지를 전달하고 자료를 교환할 수 있는 시스템에 대한 요구가 증대되었다. 그 중 대표적인 연구의 하나가 P2P 기술을 기반으로한 Jabber 실시간 메시징 시스템이다. 본 장에서는 개발된 Jabber 메시징 시스템과 기개발된 시스템과의 비교를 통하여 시스템이 가지는 장점과 특성을 비교 분석하고자 한다. <표 1>은 시스템이 지원하는 서비스 및 기능을 중심으로 상호비교한 표이다. 서비스 등급은 크게 세 가지로 나누어 비교하였으며, 운영 플랫폼은 윈도우와 리눅스 두 가지에 대해서만 조사하였다. 표에서 보는바와 같이 본 논문에서 개발된 메시징 시스템은 유일하게 협업 시스템과의 연동을 지원하는 메신저 시스템으로서, 그룹채팅, 파일 송수신 등 실시간 메시징 서비스를 위한 기본 기능을 또한 지원하고 있다.

〈표 1〉 Jabber 기반 시스템의 성능 비교

시스템명	그룹 채팅	Unicode 지원	SSL 지원	파일 송·수신	협업시스템 연동	Message History지원	운영 체제
*Jabber 메신저	○	○	×	○	○	○	wondows
Gabber	○	△	○	○	×	○	Linx
Jabberwana	○	○	×	○	×	×	wondows
Exodus	○	△	○	○	×	○	wondows
Yabber	△	○	×	○	×	○	wondows
JabberX	○	△	×	×	×	△	Linx

* : 본 논문에서 개발한 Jabber 기반 메시징 시스템.
○ : 완벽하게 지원, △ : 부분적으로 지원, × : 지원하지 않음.

5. 결론 및 추후 연구

본 논문에서는 Jabber 실시간 메시징 시스템이 BioPlace 협업지원 시스템에서 실시간 의사소통 수단으로 사용될 수 있도록 부서 정보와 팀작업장에 대한 정보를 기술하는 추가적인 Jabber XML 프로토콜을 설계하고, 이를 지원하는 확장된 Jabber 메신저 서버와 Jabber 메신저 클라이언트 시스템을 개발하였다.

Jabber 서버의 JSM 컴포넌트에 일반적인 Jabber 인증과는 별도로 BioPlace로의 인증을 담당하는 bioplace_auth 컴포넌트와 BioPlace의 부서 및 팀작업장의 정보를 XML 파일로 변환하여 이것을 클라이언트에게 전달해 주는 bioplace_roster 컴포넌트를 추가하였다. 메신저 클라이언트는 Matrix COM 동적 라이브러리를 확장하여 추가된 Jabber 프로토콜을 지원할 수 있도록 구현되었다.

BioPlace의 메신저 클라이언트는 기존의 Jabber 클라이언트에 BioPlace의 사용자 부서 및 팀작업장에 대한 정보를 효과적으로 표시해 주기 위한 사용자 중심의 인터페이스를 제공한다. 추가로, 기존 Jabber 서버가 한글로 된 정보의 처리를 제대로 하지 못하는 단점을 보완하였는데, 클라이언트측에서는 서버측에 전송할 XML 문자열을 생성한 후, UTF-16LE와 UTF-8 사이의 코드체계를 바꿔주는 별도의 모듈을 작성하였으며, Jabber 서버에서는 iconv라는 코드체계 변환 라이브러리를 사용하여 변환 함수를 구현함으로써, 서버-클라이언트-데이터베이스간의 서로 다른 코드체계를 매개하였다.

추후 공동작업장의 중요 문서 및 결과 문서에 대하여 보안기능을 추가한 공유방법을 지원함으로써 공동작업의 효율성을 높일 예정이다.

참 고 문 헌

[1] Myung-Joon Lee, Geon-Tae Ahn, Jin-Hong Kim, Keun-Woong Lee, Hyeong-Seo Koo, In-Seob Han, "BioPlace : a Web-based Collaborative Environment for Meeting of Korean SEffective Genome Research," Proceedings of the Annual ociety for Bioinformatics, Vol.1, pp.77-84, 2002.

[2] http://www.jabber.org/about/techover.html, Jabber Technology.

[3] http://www.jabber.org/protocol/, Jabber Protocol Review.

[4] D. J. Adams, "Programming Jabber," O'Rally, 2002.

[5] T. Muldowney, E. Landrum, "The Jabber Programmers Guide-A comprehensive Snapshot of jabber," Jabber Software Foundation, 2000.

[6] Li Gong, "JXTA : Technology Overview," Sun Microsystems, Inc.

[7] http://www.jxta.org/, Project JXTA.

[8] D. Brookshier, S. Li, B. Wilson, "JXTA : P2P Grows Up," Technical Articles in java.sun.com, 2002.

[9] J. Girard, "P2P Applications : New Internet Bandwidth Monsters," GartnerGroup Research Note Tactical Guidelines, December, 2000.

[10] Tyler Jewell, "EJB 2.0 specification release review," On-Java. com, May, 2001.

[11] Java2 Platform, Enterprise Edition Blueprints, "J2EE™ Design Patterns."

[12] http://www.orionserver.com/, Orion Application Server.

[13] http://postgresql.lerner.co.il/users-lounge/index.html, PostgreSQL.

[14] P. Saint-Andre, "XML : Messaging With Jabber," O'Reilly's Emerging Technology Conference, April, 2003.

[15] 문남두, 안건태, 김진홍, 한천용, 정명희, 이명준, "CoWare : 효과적인 공동작업을 위한 웹기반 그룹웨어", 정보처리학회 논문지B, 제8-B권 제3호, pp.269-282, 2001.

이 근 웅

e-mail : daredevl@mail.ulsan.ac.kr

2001년 울산대학교 컴퓨터정보통신 공학부 (공학사)

2003년 울산대학교 컴퓨터정보통신 공학 석사

관심분야 : P2P, 메신저 시스템, 협업지원 시스템 등

안 건 태

e-mail : java2u@mail.ulsan.ac.kr

1999년 울산대학교 전자계산학과 공학사

2001년 울산대학교 컴퓨터정보통신 공학 석사

2003년 현재 울산대학교 컴퓨터정보통신 공학부 박사과정

관심분야 : 협업지원시스템, 생물정보학, 웹 프로그래밍, 로직 프로그래밍 등

황 의 윤

e-mail : heyoon@mail.ulsan.ac.kr

2003년 울산대학교 컴퓨터정보통신 공학사

2003년 현재 울산대학교 컴퓨터정보통신 공학부 석사과정

관심분야 : 생물정보학, 웹 프로그래밍 등

김 진 홍

e-mail : karif99@mail.ulsan.ac.kr

1999년 울산대학교 전자계산학과 공학사

2001년 울산대학교 컴퓨터정보통신 공학 석사

2003년 현재 울산대학교 컴퓨터정보통신 공학부 박사과정

관심분야 : 생물정보학, 웹 프로그래밍, 로직프로그래밍, 협업 지원시스템 등

이 명 준

e-mail : mjlee@mail.ulsan.ac.kr

1980년 서울대학교 수학과(학사)

1982년 한국과학기술원 전산학과(석사)

1991년 한국과학기술원 전산학과(박사)

1993년~1994년 미국 버지니아대학 교환교수

1982년~현재 울산대학교 컴퓨터정보통신 공학부 교수

관심분야 : 프로그래밍언어, 분산 객체 프로그래밍 시스템, 병행 실시간 컴퓨팅, 인터넷 프로그래밍시스템 등