XML기반 전역 Peer-to-Peer 엔진 설계 및 구현

정회원 권 태 숙*, 이 일 수**, 이 승 룡***

Design and Implementation of XML based Global Peer-to-Peer Engine

Tae-suk Kwon*, Il-su Lee**, Sung-young Lee

요 약

본 논문에서는 다양한 종류의 서비스 지원이 가능하며, PC, 웹, 모바일 환경을 연동 할 수 있는 새로운 개념의 XML 기반 글로벌 P2P 엔진을 제안하고 이에 대한 설계 및 구현 경험을 소개한다. 제안된 P2P 엔진은 모든 메시지 교환 시 텍스트 기반의 XML을 사용함으로써 웹 연동 및 이기종간 데이터 교환이 가능하며, 다중 수준의 보안레벨과 여러 보안 알고리즘을 적용할 수 있는 기능도 제공한다. 이를 위하여 제안된 시스템은 모든 메시지를 스케쥴링, 필터링 하는 Message Dispatcher, 보안 기능을 지원하는 보안 관리자와 전송을 담당하는 전송 관리자를 포함하는 SecureNet Manager, 피어를 검색하여 피어 네트워크 환경을 구성하는 Discovery Manager, 그리고 XML 문서처리 기능을 포함하는 데이터 관리자인 Repository Manager 모듈로 구성되어있다. 본 논문에서 제안된 시스템의 가용성 평가를 위해 커뮤니케이션 서비스인 채팅과 협업 중 공동 저작 도구로서 화이트보드 그리고 파일 공유서비스를 각각 구현하고, 기존의 타 시스템과의 성능 비교 평가를 하였다.

Key Words : P2P, Peer-to-Peer, XML

ABSTRACT

SIn this paper, we introduce our experience for designing and implementing new concept of a global XML-based Peer-to-Peer (P2P) engine to support various P2P applications, and interconnection among PC, Web and mobile computing environments. The proposed P2P engine can support to heterogeneous data exchanges and web interconnection by facilitating with the text-base XML while message exchange are necessary. It is also to provide multi-level security functions as well as to apply different types of security algorithms. The system consist of four modules; a message dispatcher to scheduling and filtering the message, a SecureNet to providing security services and data transmission, a Discovery Manager to constructing peer-to-peer networking, and a Repository Manager to processing data management including XML documents. As a feasibility test, we implement various P2P services such as chatting as a communication service, white-board as an authoring tool set during collaborative working, and a file system as a file sharing service. We also compared the proposed system to a Gnutella in order to measure performance of the systems.

* 경희대학교 컴퓨터공학과 실시간&멀티미디어 연구실(ssuki@oslab.khu.ac.kr),
** 경희대학교 컴퓨터공학과 실시간&멀티미디어 연구실(primelee1@hotmail.com),
*** 경희대학교 컴퓨터공학과 교수(sylee@oslab.khu.ac.kr)
논문번호 : 020215-0504, 접수일자 : 2002년 2월 15일

I. 서 론

컴퓨터 시스템의 빠른 발전과 네트워크 속도의 향상으로 서버로부터 서비스를 받기만 하던 클라이언트들이 서버의 역할을 대행할 수 있을 정도로 클라이언트와 서버의 역할 구분이 모호하게됨에 따라 P2P 환경이 일반화되어가고 있다. 이러한 P2P 네트워크 환경의 응용분야는 크게 파일 공유, 협업, 분산 컴퓨팅, 저장소 공유, 커뮤니케이션 등으로 나눌 수 있다[1]. 그러나, 다양한 응용분야에도 불구하고 현재 서비스되고 있는 P2P 솔루션들은 각각의 서비스만을 지원할 수 있도록 설계되어 추가적인 서비스의 확장이 쉽지 않다. 또한, 멀티미디어 데이터와 같은 다양한 데이터의 저장 및 검색 기능과 보안 기능이 미약하며, 다양한 네트워크 환경을 지원할 수 있는 투명성도 부족하다.

따라서 본 논문에서는 기존의 여러 P2P 응용 서비스를 지원할 뿐만 아니라, 웹 표준이 될 XML을 기반으로 하여 웹 연동을 지원하도록 하고, 각각의 피어간에 암/복호화를 위해 키를 교환할 때 인증서 뿐만 아니라 직접 공개키를 공유할 수 있는 다양한 네트워크 프로토콜을 지원하는 XML 기반 글로벌 Peer-to-Peer(P2P) 엔진의 설계 및 구현 경험에 대해 소개한다. 제안하는 P2P 엔진은 여러 서비스를 지원할 수 있도록 하기 위해 P2P 서비스들이 공통적으로 필요로 하는 요구사항을 분석하여 설계한다. 제안하는 P2P 엔진은 여러 P2P 서비스를 통합 지원하고, 모든 메시지 교환에 텍스트 기반의 XML을 사용함으로써 웹 연동 및 이기종간 데이터 교환, 그리고 XML 메타데이터를 지원하며, 서비스가 필요한 수준의 보안레벨을 적용할 수 있는 보안 구조를 제공한다.

본 논문의 구성은 다음과 같다. 2장에서 관련 연구를 소개하고 P2P 서비스들의 공통 요구사항을 분석하고, 3장에서는 이러한 요구사항을 충족하는 P2P 엔진 설계에 대해 설명하며, 4장에서는 시스템 구현에 대해 기술한 후, 5장에서 구현한 시스템에 대해 성능을 평가하고, 마지막 6장에서 결론을 내린다.

II. 관련 연구

본 장에서는 P2P 연구로서 대표적인 파일공유 시스템인 누텔라와 프리넷, 그리고 P2P 협업 플랫폼인 그루브에 대해 살펴보겠다. 현재 서비스되고 있는 대부분의 P2P 솔루션들은 누텔라의 구조를 따르고 있다. 그리고 프리넷은 P2P를 활용한 분산 스토리지에 대한 연구이며 그루브는 최초로 P2P 네트워크를 이용해 협업을 구현한 플랫폼이다.

누텔라(Gnutella)[10]는 Winamp로 NullSoft가 만들어 공개되었으며 현재 서비스되고 있는 대부분의 P2P 솔루션의 개발에 활용되고 있다[10]. 누텔라는 하이브리드 형태의 냅스터와 달리 이용자간에 디렉토리 정보를 릴레이식으로 중개해 주는 서버가 필요 없는 순수한 형태의 P2P 방식으로 거의 모든 플랫폼에서 개발되어 있어 플랫폼에 독립적인 파일공유 솔루션이다. 누텔라에서 특징적인 것은 서버가 없는 상태에서 다른 피어를 검색하여 피어 네트워킹을 구축하는 방법이 제시된 프로토콜이다. 누텔라의 프로토콜 중 헤더는 Message identifier는 메시지에 대한 고유 ID로 GUID를 사용한다. Payload descriptor는 메시지의 종류를 나타내는데, 메시지 종류는 Ping, Pong, Push Request, Query, Query hits의 5가지로 이루어져 있다. TTL(Time To Live)은 메시지의 전달 깊이를 지정하는 값이며, Hops는 메시지가 전달된 시간을 나타내고 Payload length는 길이를 나타낸다. 누텔라는 Ping-Pong, 검색(Query), 다운로드의 세 가지 기능을 가지며, 작동 원리는 다음과 같다. 우선 A는 어떤 PC에서 누텔라가 실행되는지 *Ping* 메시지를 자신이 관리하는 피어 B로 전달하면, 누텔라가 실행되고 있는 다른 피어들에서 Pong 메시지를 A에 보내게 되고 A는 이를 이용해 누텔라가 실행되어 있는 B의 정보를 얻을 수 있다. B는 다시 연결되어 있는 다른 피어들에 Ping 메시지를 보내고 그 결과를 A에 알린다. 이런 방식은 무한히 반복될 수도 있는 위험이 있으므로 메시지는 TTL 만큼 메시지를 전달하도록 제어한다. 따라서 누텔라의 피어 검색 방법은 최초로 메시지를 전달할 자신이 관리하는 B의 IP를 가지고 있어야 한다는 가정이 따른다. 검색(Query)은 호스트의 최소 속도와 검색 키워드로 구성된 *Query* 메시지를 가지고 있는 IP들에 보내면 다운로드받을 수 있는 IP 주소와 포트, 속도, 파일정보, 응답 호스트의 GUID로 구성된 Query Hits 메시지를 보내게 되고, 이 정보를 이용해 직접 접속하여 다운로드받게 된다. 누텔라의 메시지는 TCP/IP 프로토콜을 사용하며 다운로드에 관해서는 HTTP를 사용한다. 누텔라에서 호스트는 피어의 IP 리스트만을 관리함으로써 부하

를 줄인다. 누텔라는 순수한 P2P 네트워킹을 구축할 수 있는 방법을 제시하였으며, 오픈 소스로 거의 모든 플랫폼에서 개발되어 있다는 장점을 가진다. 그러나, 최초의 피어 IP를 가지고 있어야 하며, 브로드캐스트를 이용한 피어 탐색 방법은 전달하는 데이터 양에 따라 많은 부하를 발생할 수 있다. 누텔라는 자체적으로 보안이 적용되는 부분은 없으며, 누텔라의 구조를 활용하고, 보안을 적용하는 솔루션들이 현재 개발되어지고 있다.

프리넷[11]은 Ian Clarke의 논문[12][13]을 바탕으로 개발된 P2P 시스템으로 익명성을 완전히 보장한 상태에서 분산된 정보의 저장과 검색을 하려는데 목적이 있다. 프리넷은 데이터의 검색, 저장, 관리의 세 부분으로 구성되어 있으며, 기본적으로 프리넷은 누텔라의 브로드캐스팅 검색을 하지 않고 냅스터의 중앙집중식의 인덱스를 포함하지 않는다. 그리고 데이터가 자동적으로 정보의 인기도에 따라서 네트웍을 따라서 복사, 이동, 삭제된다. 누텔라의 Ping-Pong과 다른 점은 누텔라의 경우, 모든 노드에서 다른 노드에 Data Request 메시지를 보내는 방법으로 브로드캐스팅을 이용하지만 프리넷에서는 각각 노드로 메시지 요청과 응답을 유니캐스트 하고 받는다는 점이다. 브로드캐스팅은 동시에 여러 노드로부터 응답을 받을 수 있어 빠르다는 장점이 있는 반면에 데이터 양이 많아지면 네트워크 부하가 커진다는 단점이 있다. 그리고 프리넷의 방법은 네트워크 부하가 적은 대신 각각의 노드에게 메시지를 주고받아야 하므로 노드수가 많을수록 결과를 얻는 시간이 많이 걸린다는 단점이 있다. 따라서 본 논문에서는 초기 P2P 네트워크 생성을 위한 초기화를 위한 통신에서는 데이터 양이 적으므로 빠른 응답을 위해 브로드캐스팅을 사용하고, 데이터 전송을 포함하는 메시지의 경우에는 프리넷의 유니캐스팅을 사용하도록 하였다.

MagiTM은 현재 나온 P2P 솔루션에 관한 개념 중 가장 통합적이라고 할 수 있는 endtech의 솔루션이다. MagiTM은 기존의 웹을 적극 활용하고, 소프트웨어를 컴포넌트 형식과 이벤트 형식으로 묶고, 플랫폼의 개념으로 접근하여 피어 자체의 독립성을 최대한 확보하는 4가지 개념으로 접근하고 있는 시스템이다[21][22]. MagiTM에서는 암호화를 위해 SSL(Secure Sockets Layer)를 사용하는데, 암호화를 위한 키로써 SSL 인증서로 교환된 공개키를 사용하며, 웹과 인터넷에서 일반적으로 네 가지의 인증 정책을 지원한다[21].

그루브(Groove)[6]는 로터스노츠의 레이오지가 협업 서비스 지원을 목적으로 개발한 상용 P2P 컴퓨팅 플랫폼으로 이를 이용해 네트워크 상에 세션을 만들어 문서 저작 등의 공동작업을 할 수 있다. 그루브의 플랫폼은 XML 기반으로 확장성이 높고 커스터마이징이 쉽다. 작업시 전송되는 정보는 자동으로 암호화되며 인증처리는 사용자가 관여하지 않고 로봇이 수행한다. 누텔라처럼 부하가 심한 브로드캐스트 대신 냅스터와 같이 중앙서버를 중개자로서 이용하여 병목현상을 없애는 동시에 유동 IP를 사용하는 호스트들에 접근할 수 있도록 하였다. 그루브에서는 첫째, 사용자의 계정을 보호하기 위해 passphrases를 사용한다. 저장되는 모든 그루브 데이터는 이 passphrases로부터 얻어진 키로 암호화되어 보호된다. 둘째, 다른 사용자들의 신원을 인증하고 자신의 신원을 다른 사용자들에게 인증받기 위해서 Digital fingerprint를 사용한다. 이 Digital fingerprint는 그루브에서 사용되는 두 개의 공개키의 해쉬 값이고, 기본적인 사용시마다 그루브에 의해서 계산되어 검증될 수 있다. 마지막으로, role과 permission을 할당함으로써 공유공간의 멤버들의 정보 접근을 관리한다. 그루브에서는 공유공간에 다른 사람을 참여시키기 위해서 초대 메시지를 보내는데 보안이 확보되는 송신방법으로서 PGP나 S/MIME과 같은 보안 E-mail을 사용한다. 그러나 PGP나 S/MIME은 그루브와는 별도로 설치하여야 하며, 환경설정이 어려운 단점이 있다. 또한, 그루브는 P2P 상에 협업을 할 수 있는 환경을 잘 구축하였으나, 상업용 플랫폼으로 공개가 되어 있지 않으며 피어간 공유 정보를 동기화 하는데 시스템 부하가 크다는 단점이 있다.

[표 1]은 기존의 P2P 시스템과 제안하려는 P2P 엔진과의 지원 기능 차이를 보여준다. 누텔라와 프리넷은 오픈소스 프로젝트로 소스가 공개되어 있는 파일 공유 프로그램으로 현재 서비스되고 있는 파일 공유 프로그램은 대부분은 이 소스의 응용이다. 그루브는 상업용 플랫폼으로 공개되어 있지 않고 P2P 네트워크 상에 협업을 구현한 P2P 협업 플랫폼이다. 그리고 누텔라와 프리넷은 중앙 서버가 없는 순수 P2P 모델을 기반으로 하며 XML을 지원하지 않고 그루브는 중앙의 관리 서버가 있는 하이브리드 모델을 지원한다. 제안한 P2P 엔진은 순수 모델과 하이브리드 모델을 서비스의 필요에 따라 지원하도록 하였으며, 그루브와 같이 XML 기반이다. 마지막으로 누텔라는 보안기능을 지원하지 않으며 암호화와 인증은 프리넷과 그루브, 그리고 제안된 P2P 엔진에서 지원한다.

표 1. 관련 연구와 제안한 P2P 엔진 비교

비교대상 / 비교항목		Gnutella	Freenet	Groove	제안한 P2P엔진
오픈소스 여부		공개	공개	비공개	-
지원서비스	File Sharing	O	O	O	O
	Collaboration	X	X	O	O
	Communication	X	X	O	O
P2P 네트워크 모델		Pure	Pure	Hybrid	Pure, Hybrid
XML 지원여부		X	X	O	O

표 2. P2P 서비스 기능 분석

대상 서비스 / 비교 항목		국내		국외		
피어 탐색 기능		K-Tella (Gnutella 응용)	나리지언[14]	Napster	Groove	MagiTM[15]
데이터 전송 기능		O	O	O	O	O
저장소		O	O	O	O	O
보안	암호화	X	O	X	O	O
	인증	X	X	X	O	O

Ⅲ. P2P 엔진 설계

1. P2P 서비스 지원을 위한 요구사항

P2P 서비스들이 공통적으로 필요로 하는 핵심 기능으로는 첫째, 다른 피어들의 존재 유무를 검색하고 피어 네트워킹 환경을 만드는 기능을 필요로 한다. 둘째로 여러 데이터를 전송하기 위한 기능으로 전송 데이터는 필요에 따라 암호화되어 보안이 적용되어야 한다. 셋째로 피어가 가지는 여러 가지 형태의 데이터를 저장 및 관리하는 기능을 필요로 한다. 현재의 P2P 공유 대상은 멀티미디어 파일로 한정되어 있지만 앞으로 P2P의 적용영역이 웹으로 확장되면 멀티미디어 파일뿐만 아니라 훈글, MS워드, HTML 등의 문서를 포함하는 여러 가지 형태의 데이터에 대한 정보를 저장하고 검색할 수 있어야 한다. 그러나 현재의 웹 문서 전체에 대한 키워드 검색은 부하가 많고 검색하는데 많은 시간을 낭비하며, 파일 이름에 대한 검색은 의미론적 검색을 할 수 없다. 그리고, P2P 시스템은 연결된 피어의 수가 많을수록 더욱 많은 리소스를 지닌 거대한 네트워크를 형성하기 때문에, 많은 사용자들이 이용하기 위해서는 경량화 되어야 한다. 마지막으로 서비스와 서비스, 서비스와 모듈, 그리고 모듈과 모듈간의 메시지들을 스케줄링하고 관리하는 관리자가 필요하다. [표 2]는 몇몇 대표적인 P2P 서비스들의 기능의 요구사항을 보여준다.

누텔라는 파일 공유 서비스 위주로 설계되었으며 브로드캐스트를 이용하여 피어를 탐색하는 Ping-Pong 방식은 데이터 양에 따라 피어의 네트워크에 많은 부하를 준다. 그리고 프리넷의 방법은 네트워크 부하가 적은 대신 각각의 노드에게 메시지를 주고받아야 하므로 노드수가 많을수록 결과를 얻는 시간이 많이 걸린다는 단점이 있다. 따라서 본 논문에서는 초기 P2P 네트워크 생성을 위한 초기화를 위한 통신에서는 데이터 양이 적으므로 빠른 응답을 위해 브로드캐스팅을 사용하고, 데이터 전송을 포함하는 메시지의 경우에는 프리넷의 유니캐스팅을 사용하도록 하였다. 그리고 데이터 전송 기능에 암/복호화를 결합하여 서비스 또는 사용자가 설정한 보안레벨에 따라 전송단에서의 암/복호화를 수행하도록 하였고 네트워크 인터페이스를 제공하여 투명성을 보장하였다. 여러 가지 데이터를 저장, 관리해야 하는 저장소는 데이터에 대한 XML 메타데이터를 생성함으로써 해결하였다. 이때 텍스트 기반의 XML 형식으로 만들어진 메타데이터는 타 시스템이나 웹에서 재활용될 수 있다. 메시지 관리자는 서비스와 서비스, 서비스와 모듈, 모듈과 모듈간의 모든 메시지를 우선순위 큐를 이용해 스케줄링하고 메시지를 필터링하여 처리해야할 모듈에 전달한다. 그리고 모든 메시지는 XML 포맷으로 만들어 확장성, 이식성을 제공하도록 하였다.

2. 제안한 P2P 기본 모델

제안한 시스템은 기존의 여러 P2P 응용 서비스를 지원할 뿐만 아니라, 새로운 웹 표준인 XML을 지원하며, 여러 보안레벨 및 알고리즘을 지원하는 XML 기반 글로벌 P2P 엔진의 설계 및 구현을 목표로 하며, 이에 대한 전체 시스템의 구조는 [그림 1]과 같다.

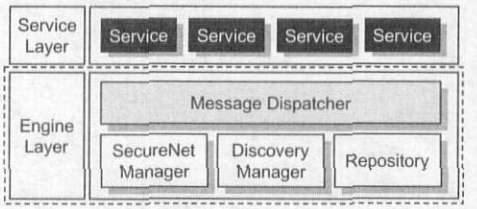


그림 1. XML 기반 글로벌 P2P 엔진 구조

시스템은 크게 서비스 계층과 엔진 계층으로 나누어진다. 서비스 계층에는 여러 가지 서비스를 구축할 수 있도록 하여 융통성을 제공한다. 본 연구에서는 성능 평가를 위해 화이트보드(협업), 채팅/쪽지(커뮤니케이션), 음악파일공유(파일공유) 서비스를 구현하여 본 연구의 테스트용으로써 사용한다. 엔진 계층은 P2P 서비스들의 공통으로 가지는 기능에 따라 네 개의 모듈로 구성된다. 우선, 다른 피어의 존재유무 및 정보를 검색하는 기능은 Discovery Manager, 암복호화 기능을 포함하는 전송 모듈은 SecureNet Manager, 그리고 저장소인 Repository와 모듈간 메시지를 중개하고 스케쥴링하는 Message Dispatcher로 구성하였다.

모듈간에 통신을 위한 모든 메시지는 [표 3]의 기본 구조를 갖는 XML 포맷으로 구성되며, [그림 2]는 이를 전송 패킷 형태로 나타낸 것이다.

표 3. 메시지 기본 포맷

```
<?xml version="1.0" encoding="euc-kr" ?>
<BODY>
  <MSG>
1:    <COMMAND>SEND</COMMAND)
2:    <PROTOCOL>TCP</PROTOCOL>
3:    <FROM>111.111.111.111</FROM>
4:    <TO>222.222.222.222</TO>
5:    <APP>WHITEBOARD</APP>
6:
7:  <CONTENT>msg_contents</CONTENT>
  </MSG>
</BODY>
```

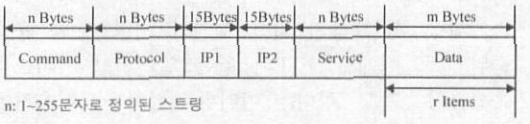


그림 2. 메시지 포맷의 패킷형태

<MSG> 태그로 둘러 쌓인 부분이 모듈간 통신에 사용되는 부분이다. 모두 6가지의 요소로 구성하였다. 메시지 내에 사용되는 정의된 문자열들은 임의의 파일에 저장되어 있으며 텍스트를 추가함으로써 새로운 항목을 추가할 수 있다. 2라인의 <COMMAND>는 메시지가 수행할 작업을 나타낸다. [표 4]는 기본적으로 정의되어 있는 커맨드와 각각의 작업을 보여준다. 3라인의 <PROTOCOL>은 메시지 전송 시 사용할 프로토콜의 종류로 TCP, UDP, HTTP 키워드를 사용한다. 4, 5라인의 <FROM>, <TO>의 IP는 각각 메시지를 보내는 피어의 IP와 받을 피어의 IP를 나타낸다. 6 라인의 <APP>는 이 메시지가 해당하는 서비스의 종류를 나타낸다. 7라인의 <CONTENT>는 서비스에 전달할 정보에 해당하는데 경우에 따라 암호화 될 수 있으며 메시지를 받은 서비스는 이 부분을 다시 필터링 하여 서비스의 세부 작업을 수행한다.

표 4. 기본 커맨드 정의

커맨드	수행작업
SEND	메시지의 <CONTENT>를 다른 피어로 전송한다.
SEARCH	<CONTENT>의 쿼리에 해당하는 정보를 연결되어 있는 피어 시스템에서 검색한다.
DOWNLOAD	검색한 파일을 다운로드 받는다.

3. 세부 모듈 설계

제안한 P2P 엔진의 세부모듈은 [그림 3]에서 보여준다. 시스템은 크게 상단의 서비스층과 메시지 Message Dispatcher, SecureNet Manager, Repository, Discovery Manager로 구성되어 있는 엔진층으로 나누어지며 서비스층은 다수의 서비스를 추가할 수 있도록 하여 확장성을 제공한다. 그리고 엔진층은 P2P 서비스가 필요로 하는 최소한의 기능들로 구성하여 경량화 하였으며, 모듈간 통신 메시지를 XML 포맷으로 만들어 확장성을 제공하고 웹에서 재활용할 수 있도록 하였다.

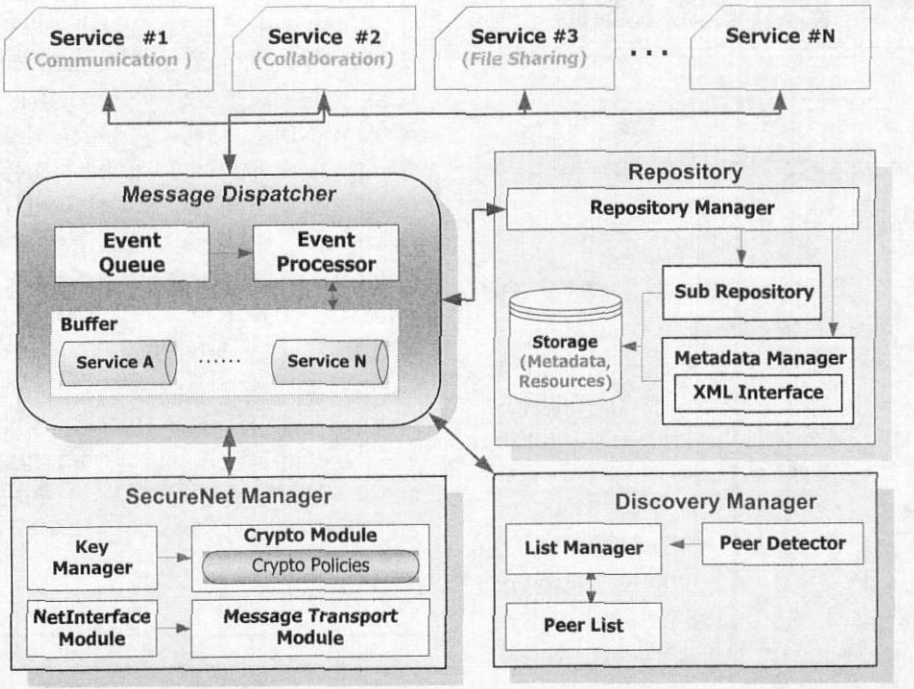


그림 3. XML 기반 글로벌 P2P 엔진 세부 모듈 구조

• Message Dispatcher

메시지 디스패쳐는 [그림 4]에서 보는 바와 같이 기능에 따라 Message Queue, Message Processor, Service Buffer 모듈로 구성되어져 있다.

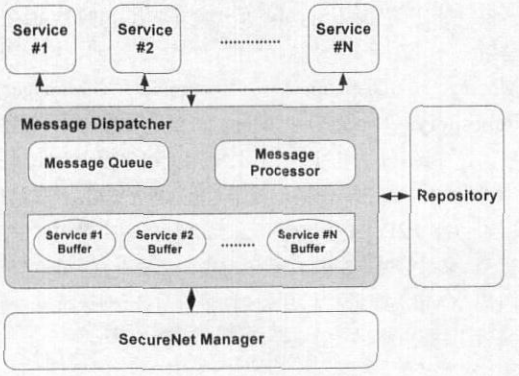


그림 4. 메시지 디스패쳐의 설계도

Message Queue는 P2P 엔진에서 발생하는 메시지를 큐에 배치하는 역할을 한다. 이 Message Queue의 핵심은 P2P 엔진에서 발생되는 메시지들을 어떠한 정책에 의하여 큐에 배치 할 것인가 하는 문제이다. 본 논문에서 언급하고 있는 P2P 엔진은 다양한 서비스를 지원하며 각 서비스들은 실시간 특성을 지닌 서비스와 그렇지 않은 서비스로 구분할 수 있다[표 5].

표 5. 메시지 레벨 구분

레벨	특성	서비스 종류
1	실시간 서비스	VOD, 채팅, 협력작업, etc
2	일반적인 서비스	e-mail, ftp, etc

따라서 본 논문에서는 메시지의 우선순위를 정하고 우선순위가 높은 것을 큐의 앞쪽에 배치하며, 우선 순위가 같을 경우에는 메시지의 마감시간을 비교하여 마감시간이 짧은 것들을 앞쪽에 배치하는 형태의 우선순위 큐를 사용하여 메시지들을 스케줄링 하도록 한다. 메시지들의 우선순위는 VOD 서비스, 채팅, 협업 등과 같은 실시간 서비스들을 우선순위 1로, 이메일, FTP와 같은 일반적인 서비스들 우선순위 2로 한다. 이러한 정책을 이용하여 Message Queue는 메시지의 우선순위와 마감시간을 검사하고, 각 메시지의 우선순위와 마감시간에 따라서 메시지들을 큐에 배열하는 작업을 수행한다.

Message Processor은 Message Queue에 배치되어 있는 메시지들을 순차적으로 처리하는 모듈로써 XML 파서를 이용하여 메시지들을 분석하고 각 모듈로 메시지를 전달한다. Message Processor가 각 메시지들을 처리하는 방법은 [표 6]와 같다. Message Processor의 메시지 처리방법은 먼저 전달되어진 메시지를 XML 파서를 이용하여 분석한다(1라인). 분석이 끝난 메시지는 그 메시지에 해당하는 버퍼에 저장된다(2라인). 다음으로 메시지의 종류에 따라 다른 처리과정을 거친다. 첫 번째로 다른 서비스에게 전달되어지는 메시지라면 해당하는 서비스에 이벤트를 발생시키고, 이벤트가 접수된 서비스는 버퍼로부터 메시지를 가져가게 된다(3라인). 두 번째로 다른 피어에게 전달되는 메시지라면 다른 피어의 네트워크정보와 보안정보를 보안네트워크관리자에게 보내주고 버퍼에서 메시지를 가져가도록 한다(4라인). 세 번째로 레파지토리에 전달되는 메시지라면 메시지 타입에 맞게 메시지를 저장하도록 메시지의 타입을 레파지토리에게 알려주고 레파지토리에 저장한다(5라인).

Service Buffer는 각 서비스 또는 모듈별로 메시지 처리에 대한 결과물을 얻기 위한 메시지 디스패처와 공유된 버퍼로써, 서비스들이 엔진에 추가되면서 각 서비스 별로 한 개씩의 버퍼를 생성한다. 또한 이 버퍼들은 다양한 서비스들의 특성을 고려하여, [16]의 Push와 Pull 기능을 동시에 지원하는 버퍼관리기법을 적용하여, Push와 Pull 방식을 모두 지원하는 버퍼이다.

표 6. Message Processor의 메시지 처리방법

```
1: Analysis Message by use XML Parser
2: Store Message in Buffer
3: if send other Service then
       send Event in Service
4: else if send other Peer then
       send other Peer information in SecureNet Manager
       send Message in SecureNet Manager
5: else if send Repository then
       send Message type in Repository
       store Message
```

- SecureNet Manager

보안 네트워크 관리자는 피어간의 데이터를 송수신하는데 있어 보안을 지원하는 모듈과 전송에 관계된 모듈로 구성된다. 보안을 지원하기 위해서 다양한 보안등급(Security Level)을 두어 다양한 데이터의 전송을 지원하기 위한 전송 모듈로 구성된다. [그림 5]는 Key Manager, Crypto Module, NetInterface Module, Message Transport Module로 구성되는 보안 네트워크 관리자의 구성도이다.

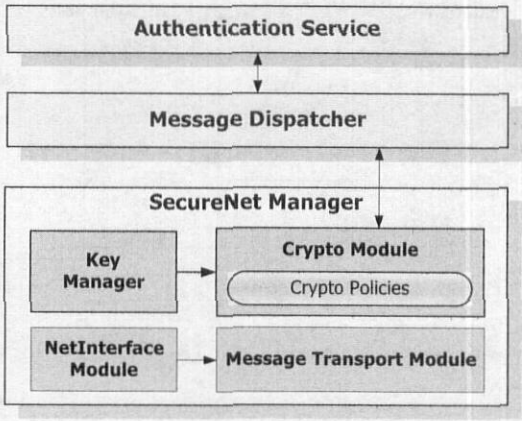


그림 5. SecureNet Manager의 구성도

Key Manager는 사용되는 세션키 및 그룹키의 분배, 관리를 담당한다. Crypto Module은 전송 및 수신시에 XML문서를 암호화/복호화 하는 과정을 수행한다. 이에 대한 정책을 수행하기 위하여 Crypto Module은 보안 정책를 포함하고 있다. 보안 정책은 암·복호화 하기 위한 암호화 방식 및 암호화 알고리즘의 선택과 암호화 키의 사용에 따른 정책을 포함한다. 이러한 보안 정책에서는 데이터의 중요도에 따라 보안레벨(Secure Level)을 두어 차별화된 알고리즘 및 키를 적용하는데, 이를 위해 XML 서명, 대칭키/공개키 암호 알고리즘, 해쉬 알고리즘 등 다양한 암호 알고리즘을 지원한다. NetInterface는 다양한 형태의 데이터의 전송, 그리고 TCP, UDP의 여러 네트워크 환경을 지원할 수 있도록 투명성을 제공하는 데이터 전송 모듈이다. Message Transport 모듈은 NetInterface를 이용하여 다른 피어에게 데이터를 전송하는 역할을 하며, 데이터의 종류에 따라서 전송 방식을 선택할 수 있는 정책을 가지고 있다.

[그림 6]은 메시지 전송 과정에 대한 그림이다. Message Dispatcher에서 전송 요청에 대한 XML 포맷의 메시지를 CryptoModule이 받으면 우선 보안등급을 확인하여 암호화 필요성을 판단한다. 암호화해야할 메시지라면 Key Manager에 Key 값을 요

청하고 Key Manager는 해당 키값을 생성하여 CryptoModule에 넘겨준다. CryptoModule은 이 키값을 이용해 메시지를 암호화하고 메시지 전송 모듈에 전송을 요청한다. 전송 모듈은 전송결과를 Message Dispatcher에 알린다.

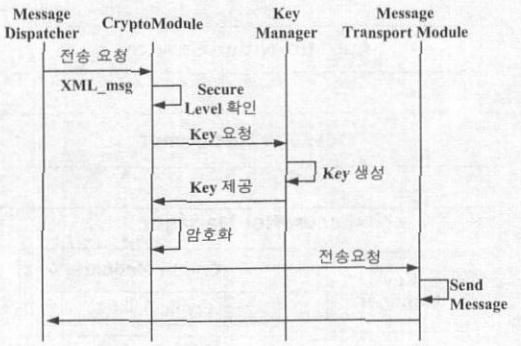


그림 6. 메시지 전송시 처리과정에 대한 시퀀스 다이어그램

• Discovery Manager

P2P 네트워크 형태에는 하이브리드 모델과 순수 모델의 두 가지 형태가 있다. 하이브리드 모델은 클라이언트/서버와 P2P의 중간적인 형태로 중앙에 서버가 존재하기는 하지만 데이터는 피어가 가지고 있으며 서버는 피어의 데이터에 대한 인덱스를 가진다. 순수 모델은 서버가 존재하지 않는 익명성 지향의 모델이다. 위의 두 가지 모델의 P2P에서는 피어들을 탐색할 방법을 필요로 한다. 초기 P2P 네트워크 생성을 위한 초기화를 위한 통신에서는 데이터 양이 적으므로 빠른 응답을 위해 브로드캐스팅을 사용하고, 데이터 전송을 포함하는 메시지의 경우에는 프리넷의 유니캐스팅을 사용하도록 하였다

디스커버리 관리자는 P2P 엔진의 네트워크 상에서 새로이 접속된 다른 노드들의 존재를 감지하여야 하며, 감지된 노드들의 연결 상태를 확인해야 한다. 이를 위하여 디스커버리 관리자는 [그림 7]에서와 같이 Dectect Manager, List Manager, peerList의 세 부분으로 모듈을 구성을 하였다.

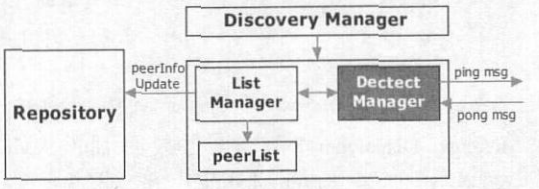


그림 7. Discovery Manager의 세부 구조

디스커버리 관리자의 각 모듈의 역할은 다음과 같다. Dectect Manager는 P2P 네트워크에 접속하기 위하여 현재 네트워크에 접속되어 있는 다른 노드들을 감지하는 역할을 한다. 이를 위하여 Dectect Manager는 P2P엔진 가동시 초기화를 위해 브로드캐스팅을 하여 현재 구동되고 있는 다른 P2P 네트워크[그림 8]를 감지하게 된다. 피어를 감지하게 되면 노드에 접속을 함으로써 감지한 노드가 참여하고 있는 P2P 네트워크에 참여할 수 있다[그림 9].

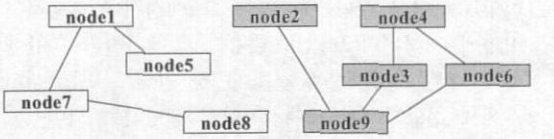


그림 8. 초기 P2P 네트워크 상태

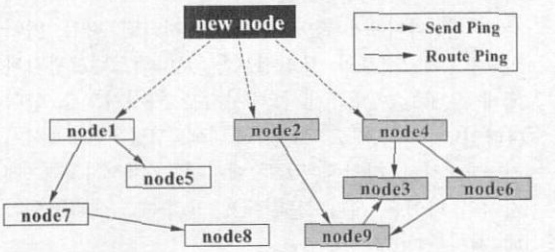


그림 9. Ping 메시지의 노드간 전달 방식

또한, Dectect Manager는 P2P 네트워크에 참여한 후에 다른 노드에게 Ping을 보내어 자신의 네트워크 참여를 알리고, 다른 노드들의 정보를 알기 위하여 Pong을 요청하게 된다. 여기서 한번 보내어진 Ping 메시지가 계속해서 브로드캐스팅이 되면 P2P 엔진의 네트워크는 Ping 메시지에 대한 트래픽이 발생되기 때문에 이를 위하여 Ping 메시지는 TTL(Time To Live)의 값을 갖도록 한다.

ListManager는 다른 노드들의 정보를 저장하기 위한 모듈이다. ListManager는 Detect Manager에서 감지된 새로운 노드들의 정보를 peerList와 Repository에 저장을 하고, Detect Manager에서 주기적으로 보내는 Ping 메시지에 대한 결과를 실시간으로 업데이트를 하여, 현재 P2P 네트워크 상황에 대한 정보를 실시간으로 가질 수 있다. peerList는 Repository에 저장하기 전에 Discovery Manager에서 노드들에 대한 정보를 저장하기 위한 저장소이다. 주기적으로 검색되어 업데이트 되어지는 P2P 네트워크 상에서의 다른 노드들의 정보를 Detect Manager가 확인하는 주기에 따라 Repository에 저장을 할 경우 발생되는 오버헤드를 줄이기 위한 임시 저장소이다.

• Repository

레파지토리가 지원하는 기능은 크게 P2P 엔진에서 생성되는 데이터들을 저장하는 기능과 공유 리소스에 대한 검색 기능, 그리고 XML 문서를 생성하고 파싱하여 처리하는 기능을 수행한다. [그림 10]의 세부 모듈 구조도에서 보여주는 것과 같이 크게 Repository Manager, Sub Repository, Metadata Manager 모듈로 구성되어 있다.

Repsitory Manager는 메시지 이동 계층의 최상위에서 Message Dispatcher, SecureNet Manager, Discovery Manager부터 오는 메시지에 대해 레파지토리의 세부 모듈들에게 작업을 분배하는 중앙 관리자의 역할을 한다. 즉, 저장소를 오고가는 모든 메시지들은 Repository Manager를 통과하게 된다. 그리고 P2P 엔진에서 생성되는 데이터를 저장하는 역할은 Sub Repository가 담당한다. Sub Repository의 ServiceInfo 는 저장해야할 서비스에 대한 정보의 저장 및 관리를 담당하고, SecurityInfo는 암호화 키와 같은 보안에 관련된 SecureNet Manager의 정보를 관리하며, PeerInfo는 Discovery Manager에서 검색한 피어들의 정보를 관리한다.

Metadata Manager는 리소스에 대한 메타데이터를 생성, 저장, 검색하는 기능을 수행한다. XML 파서로는 Microsoft사의 MSXML3를 이용하였고, XML 요소에 접근하고 메시지 포맷에 따라 XML 문서를 생성하는 XML 인터페이스를 DOM을 이용하여 구현하였다. DOM(Document Object Model)[17]은 XML 문서를 생성, 탐색, 추가, 수정, 삭제하는 MSXML에서 지원하는 API이다.

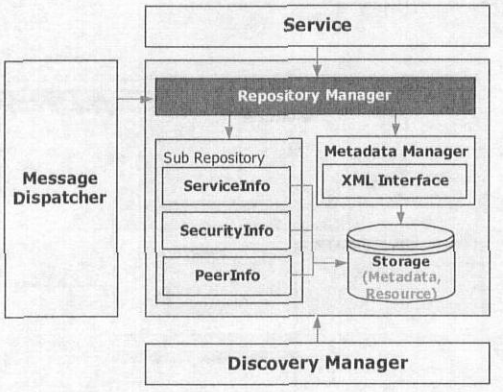


그림 9. Repository의 세부 모듈 구조도

메타데이터를 생성하여 검색하면 검색의 효율을 높일 수 있을 것이다. 그러나 메타데이터는 리소스에 대한 묘사를 따로 작성해야 하는 불편함이 있다. 그러므로 본 연구에서는 메타데이터 표준 프레임워크를 적용함으로써 한번 생성한 메타데이터를 여러 시스템에서 활용할 수 있는 융통성을 제공하였다. 메타데이터 요소는 메타데이터 표준 공개 포럼인 DCMI(Dublin Core Metadata Intiative: DCMI)의 메타데이터 요소 집합(Dubline Core Metadata Element Set Version 1.1)과 XML을 활용한 W3C의 메타데이터 표준 프레임워크인 RDF(Resource Description Framework)[18]를 적용하였다. DCMI의 메타데이터 요소 집합은 모든 웹 자원을 기술하는 데 사용될 수 있다[19]. RDF의 Dublin Core 메타데이터 15개 요소는 [표 7]과 같다[20].

요 소	정 의
표제 (Title)	그 자원에 부여된 제목
제작자 (Creator)	자원의 내용에 주된 책임을 가진 개체
주제 (Subject)	자원의 내용이 지닌 주제
설명 (Description)	자원의 내용에 대한 설명
발행처 (Publisher)	자원을 현재의 형태로 이용 가능하게 만든 실체
기타 제작자 (Contributor)	제작자 요소에 명시된 개체 이외에, 자원의 내용에 기여한 책임이 있는 기타 개체
날짜 (Date)	자원의 존재 기간 동안 어떠한 사건이 발생한 날짜
자료 유형 (Type)	자원 내용의 성격 또는 장르
형식 (Format)	자원의 물리적 표현형식 및 디지털 표현형식
식별자 (Identifier)	자원을 가리키는 지시자. 주어진 형태만으로는 그 의미를 알 수 없음
출처 (Source)	현재 자원의 출처가 되는 원 정보자원으로의 참조
언어 (Language)	자원의 지적인 내용을 기술하고 있는 언어 (1766 [RFC1766] 권장)
관련자료 (Relation)	관련 자원들로의 참조
내용범위 (Coverage)	자원의 내용 범위
이용조건 (Rights)	그 자원에 대해서 갖고 있는 권리에 관한 정보

표 7. Dublin Core의 15개 메타데이터 요소

[그림 11]은 P2P 엔진이 실행된 후, Discovery Manager가 다른 피어들을 검색하여, 피어의 정보를

PeerInfo 저장소에 저장하는 과정을 나타내며, [그림 15]는 저장소 정보의 업데이트 과정을 나타낸다. P2P 엔진이 작동되면 Discovery Manager는 P2P 네트워크 환경 세팅을 위해서 엔진이 동작중인 다른 피어들을 검색한다. 검색한 피어 정보는 IP, 피어의 시스템 정보, 엔진에서 작동중인 서비스 정보로 구성되며 이 정보를 Repository Manager에게 넘기면 해당 Sub Repository인 PeerInfo 저장소에 정보가 저장되고 저장 결과를 메인 관리자를 통해 Discovery Manager에 알리게 된다.

1) Initialization(with Discovery)

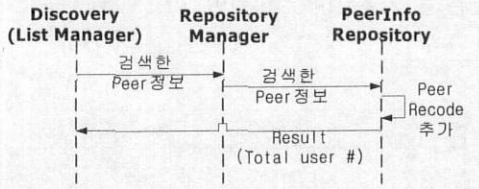


그림 11. 피어 정보 저장소의 정보 추가 시퀀스 다이어그램

그리고, 엔진 동작중 저장소는 디스패처, 시큐어넷 관리자, 디스커버리 관리자, 그리고 서비스로부터 데이터를 저장 및 검색을 요청받게 된다. [그림 12]는 메타데이터를 이용한 검색을 통한 데이터 업데이트 예제의 시퀀스 다이어그램을 간단히 보여준다. 중앙 메시지 처리 모듈인 Message Dispatcher에서는 메시지 필터링을 통해 저장소 정보의 업데이트에 관한 쿼리라는 것을 알아내어 저장소에 쿼리를 보낸다. 쿼리를 받은 Repository Manager는 메타데이터를 이용해 업데이트 할 데이터를 가지는 Sub Repository를 검색하고 쿼리를 해당 서브 저장소에 보낸다. Sub Repository는 데이터를 업데이트하고 결과를 Repository Manager를 통해 Message Dispatcher에 보낸다.

2) Update

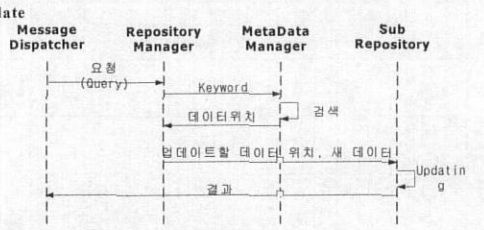


그림 12. 업데이트 처리 시퀀스 다이어그램

Ⅳ. P2P 엔진 구현

이 장에서는 제안한 P2P 엔진과 채팅, 화이트보드, 파일공유 어플리케이션을 구현함으로써 커뮤니케이션, 협업, 파일공유의 세 가지 서비스의 구현해 본다. 개발한 P2P 엔진은 C++을 이용하여 윈도우 2000에서 Visual C++ 6.0 도구를 사용하여 개발하였고, XML 파서로는 Microsoft사의 MSXML3를 사용하였다. 그리고 지원하는 전송 프로토콜은 TCP/IP, UDP, HTTP을 지원하며 서비스 어플리케이션은 인터페이스 구성이 쉬운 MFC(Microsoft Foundation Class)를 활용하였다[표 8].

표 8. 글로벌 P2P 엔진 동작환경

운영체제	Windows(Win98/2000/CE)
지원프로토콜	TCP/IP, UDP, HTTP
개발언어	C++
개발도구	Visual C++ 6.0

[그림 13]은 구현된 채팅, 화이트보드, 파일 공유 서비스들의 이용화면이다. 그림에서 화면 ❶의 창은 서비스 메인 화면으로 사용할 수 있는 서비스들의 실행 버튼이 있으며 익명상태에서 자신을 알릴 수 있는 임의의 아이디를 설정할 수 있다. 화면 ❷는 Discovery Manager가 발견한 피어들의 아이디 목록을 보여준다. 그리고 화면 ❸은 협업중 공동 저작의 예로 화이트보드 서비스를 보여주고 있으며, 화면 ❹는 대표적인 커뮤니케이션 서비스인 채팅 서비스의 사용 화면이다. 마지막으로 화면 ❺는 파일 공유 서비스이며, 사용자는 설정 버튼을 이용하여 공유하고자 하는 폴더의 리소스만을 공개하도록 제한할 수 있다.

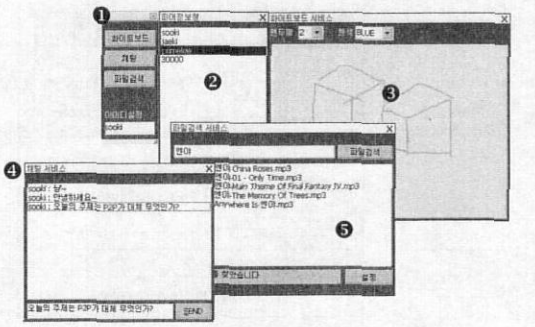


그림 13. 실행 화면 예

시스템에서 모든 메시지는 DOM을 이용해 구현한 XMLInterface를 이용하여 XML로 문서로 포맷

팅 되어 전달되어진다. [그림 14]은 채팅 서비스에서 전달되는 XML 메시지를 나타낸다.

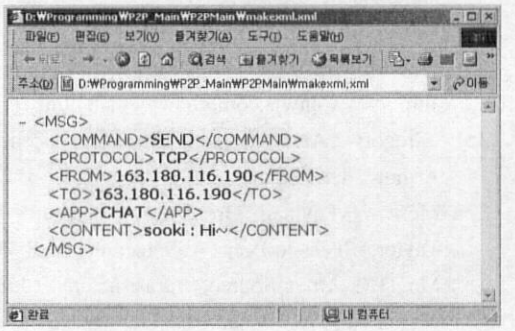
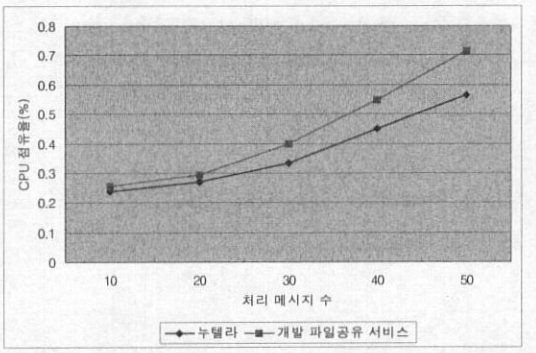


그림 14. 채팅시 교환되는 XML 메시지

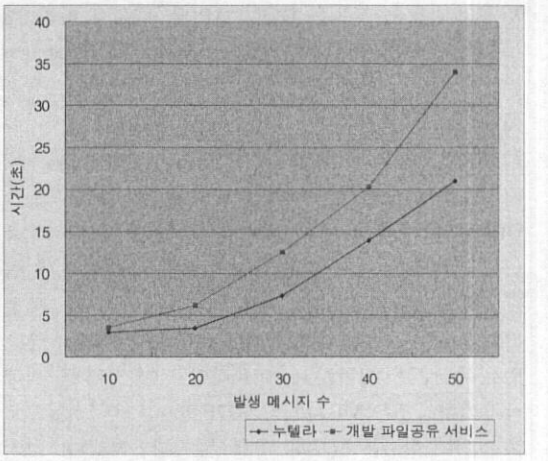
V. 성능 평가

P2P 엔진의 성능을 평가하기 위해서 대표적인 세 가지 서비스인 커뮤니케이션 서비스로서 채팅, 협업 중 공동저작의 화이트보드, 그리고 파일 공유 서비스를 구현하였다. P2P 엔진의 성능을 평가하기 위해 사용된 시스템은 Pentium II 600Mhz와 128MB 시스템에서 Windows 2000 OS이며, 100Mbps LAN 환경에서 메시지 수에 따른 처리 시간을 측정하여 누텔라와 비교하였으며, 처리 메시지 수에 따른 CPU 점유율을 측정하여 보았다. 처리 메시지는 [그림 14]와 같은 XML 포맷으로 이루어진 평균 0.3KBytes의 텍스트이며 갖는다. [그림 15]는 처리 메시지 수에 따른 CPU 점유율을 나타낸다.



[그림 15] 처리 메시지 수에 따른 CPU 점유율

[그림 16]은 메시지 수에 따라 처리시간이 비례로 증가하는 것을 알 수 있다. 또한 누텔라에 비해 메시지의 처리 시간이 길고, 처리해야할 메시지의 수가 많을수록 더욱 처리시간이 지연되는 것을 알 수 있다.



[그림 16] 발생 메시지 수에 따른 처리시간

위 결과에서 구현한 시스템이 XML 기반으로 모듈간 메시지를 만드는데 XML 포맷을 활용함으로써 확장성과 융통성을 제공할 수 있었으나 각 모듈에서 메시지의 내용을 분석하기 위해 문서를 파싱하는 추가작업으로 인해 누텔라에 비해 처리시간 지연을 보였다. 그러나 XML을 지원함으로써 얻을 수 있는 장점에 비해 성능이 크게 떨어지지 않으며 향후 속도가 향상된 XML 파서의 연구로 해결될 수 있을 것으로 생각된다.

VI. 결론

본 논문에서는 다양한 P2P 응용 서비스를 지원할 수 있는 XML 기반의 P2P 엔진을 설계하고 구현하였다. 제안된 P2P 엔진은 모든 메시지 교환에 텍스트 기반의 XML을 사용함으로써 웹 연동 및 이기종간 데이터 교환, 그리고 XML 메타데이터를 지원하며, 서비스가 필요한 수준의 보안레벨과 여러 보안 알고리즘을 적용할 수 있는 보안 구조를 제공한다. 엔진의 경량화를 위해서 여러 서비스들에서 공통으로 지원하는 핵심기능들을 위주로 Message Dispatcher, SecureNet Manager, Discovery Manager, Repository를 설계하였다. Message Dispatcher는 모듈 중간에서 모든 메시지들을 스케쥴링하고 관리하는 역할을 하는 모듈로 우선순위 큐를 이용하여 메시지의 우선순위에 따라

먼저 처리하도록 하였다. 그리고 XML로 구성되어진 메시지를 필터링하여 해당 모듈로 전달하여 준다. 이때 XML 메시지의 분석에 이용하는 XMLInterface 모듈은 DOM API를 필요에 따른 기능으로 구성하여 단순화하였다. SecureNet Manager는 데이터를 전송하는 모듈과 암·복호화 모듈을 결합하여 간소화하였고, 데이터의 전송에는 이기종 플랫폼과 TCP, UDP 프로토콜을 지원하는 네트워크 인터페이스를 구현하였다. 그리고 보안모듈은 DES등의 여러 가지 암호화 알고리즘을 지원하도록 하였다. Discovery Manager는 현재 공개되어 있는 누텔라와 프리넷의 피어 검색 알고리즘들 중 부하가 적은 프리넷의 알고리즘을 이용하였고, Repository는 정확하고 효율적인 검색지원을 위해 메타데이터를 XML기반의 RDF에 따라 구현함으로써 재사용할 수 있도록 융통성을 제공하였다. 그리고, 엔진 테스트와 시뮬레이션을 위해서 커뮤니케이션 서비스인 채팅과 협업중 공동 저작으로서 화이트보드, 그리고 파일 공유서비스를 각각 구현하고 시뮬레이션에서 성능을 측정해 보았다.

참 고 문 헌

[1] 김길동, Napster, http://www.napster.com

[2] O'Reilly, OpenP2P.com, http://www.openp2p.com

[3] Intel, PtP Working Group, http://www.peer-to-peerwg.org

[4] Ktella, http://www.ktella.com

[5] 소리바다, http://www.soribada.com

[6] Groove, http://www.groove.net

[7] SETI@home Project, http://seti.or.kr/

[8] Open4u, http://osweb1.open4u.co.kr

[9] 조용중, P2P Networking, http://www.xpert.co.kr/main/html/welcome/p2p.html

[10] Gnutella, http://www.gnutelladev.com

[11] Freenet, http://freenet.sourceforge.net

[12] I. Clarke, O. Sandberg, B. Wiley, and T.W. Hong, Freenet: A Distributed Anonymous Information Storage and Retrieval System in Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability, LNCS 2009, ed. by H. Federrath. Springer: New York, 2001.

[13] I. Clarke, A Distributed Decentralised Information Storage and Retrieval System, unpublished report, Division of Informatics, University of Edinburgh, 1999.

[14] 삼성 SDS EnWiz, 나리지언, http://www.enwiz.com/knowledgian.html

[15] Gregory Alan Boleer, Michael Gorlick, Arthur S. Hitomi, Peter Kammer, Brian Morrow, Peyman Oreizy, and Richard N. Taylor, "Peer-to-Peer Architectures and the MagiTM Open-Source Infrastructure", End-eavors http://www.endtech.com, December 6, 2000

[16] 정찬균, 이승룡, "멀티미디어 통신시스템을 위한 클라이언트에서의 Push/Pull 버퍼관리기법", 정보처리학회 멀티미디어 특집 논문집, pp. 721-732, 2000년 3월

[17] XML DOM Guide, http://www.microsoft.com/korea/msdn/xml/articles/beginner.asp

[18] RDF Spec, http://www.w3.org/RDF/

[19] Oram, Andy 저, 김필우 외 역, 'Peer-to-Peer', 13장 「Metadata」 page 340

[20] DCMI, Dublin Core·Metadata Element Set, Version 1.1: Reference Description, http://dublincore.org/documents/dces/

[21] Gregory Alan Bolcer, Michael Gorlick, Arthur S. Hitomi, Peter Kammer, Brian Morrow, Peyman Oreizy, Richard N. Taylor "Peer-to-Peer Architectures and the Magi Open-Source Infrastructure", December 6, 2000

[22] EndTech, http://www.endteck.com

권 태 숙(Tae-suk Kwon) 준회원

2000년 2월 : 한신대학교 정보통신학과 졸업
2002년 2월 : 경희대학교 컴퓨터공학과 석사
2003년 3월~현재: 경희대학교 컴퓨터공학과 실시간&멀티미디어 연구실 Research Staff

<관심분야> 인공지능, 게임 엔진, 협력시스템, P2P, 유비쿼터스

이 일 수(Il-su Lee) 준회원

1997년 2월 : 충북대학교 전기공학과 학사
2002년 2월 : 경희대학교 컴퓨터공학과 석사
2003년 7월~현재: Univ. of Southern California Computer Science 석사과정

<관심분야> 인터넷 및 웹 보안, Computer Network, Software Engineering, 협력 시스템

이 승 룡(Sung-young Lee) 정회원

1978년 2월 고려대학교 재료공학과 공학사
1987년 12월 Illinois Institute of Technology 전산학과 석사
1991년 12월 Illinois Institute of Technology 전산학과 박사
1992년 8월 Governors State Univ., Illinois 전임강사
1993년 8월 Governors State Univ., Illinois 조교수
1996년 9월 경희대학교 전자계산공학과 조교수
2001년 9월 경희대학교 전자계산공학과 부교수
2001년 10월~현재 경희대학교 전자계산공학과 교수

<관심분야> 실시간 시스템, 실시간 고장허용시스템, 멀티미디어 시스템