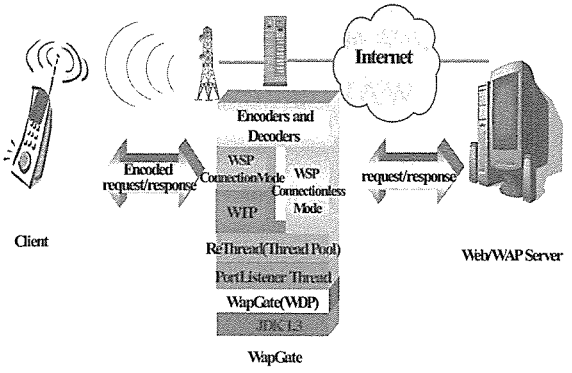
'Wap Gateway'

무선 인터넷 응용 서비스를 위한 WAP Gateway 프로그램

- 서경대학교 컴퓨터공학과 모바일 컴퓨팅 연구실 -

서경대학교 컴퓨터공학과 모바일 컴퓨팅 연구실의 이양선 교수 지도하에 장현익, 박완영이 공동 개발한 WAP Gateway 프로그램("WapGate")은 무선 인터넷 응용 서비스를 위한 미들웨어 시스템으로 사용자 단말기와 Web/WAP 서버 사이에 위치하여 유선 인터넷 환경에서 제공되는 각종 콘텐츠를 무선 인터넷 사용자에게 제공하기 위해 HTTP 프로토콜과 해당 WAP 프로토콜 사이의 변환 및 WML 문서의 인코딩 기능 등을 제공한다. 개발한 WapGate는 WAP 포럼에서 발표한 WAP 규격 1.2를 토대로 설계되었으며 Java 언어로 구현하여 운영체제나 아키텍처에 상관없이 플랫폼 독립적으로 운영될 수 있도록 하였다.



[서경대학교 'WapGate' 시스템 구성도]

WapGate 프로그램은 단말기와 같은 사용자 에이전트의 요청을 스레드로 처리하면서 재사용 가능한 스레드의 집합을 통하여 스레드 생성과 소멸 시 발생하는 오버헤드를 줄임으로서 전체적인 성능과 안정성을 높였다. 또한, 다중 포트 리스너(port listener)를 구현하여 WAP의 연결형과 비연결형 통신 기능 모두를 지원한다.

WapGate 프로그램은 현재 학교에서 무선 인터넷 프로그래밍을 위한 교육자료로 활용되고 있으며 기업에서는 무선 포탈 사이트를 통해 사용자에게 각종 콘텐츠를 제공하는데 활용될 수 있다. 또한, 앞으로 많은 수요가 예상되는 무선 인터넷 기술을 보유한 고급 인력의 양성과 무선 인터넷 컨텐츠 산업의 활성화에 긍정적인 효과가 기대된다.

정보통신부장관상

무선 인터넷 응용 서비스를 위한 WAP Gateway 프로그램

- 서경대학교 컴퓨터공학과 모바일 컴퓨팅 연구실 -

1. 작품명 : 무선 인터넷 응용 서비스를 위한 WAP Gateway 프로그램("WapGate")

2. 제작자 : 서경대학교 컴퓨터공학과 모바일 컴퓨팅 연구실

- 지도교수 : 이 양선
- 출 품 자 : 장 현 익, 박완영
- 연 락 처 : javapl@korea.com

3. 프로그램 개요

3.1 개발 배경

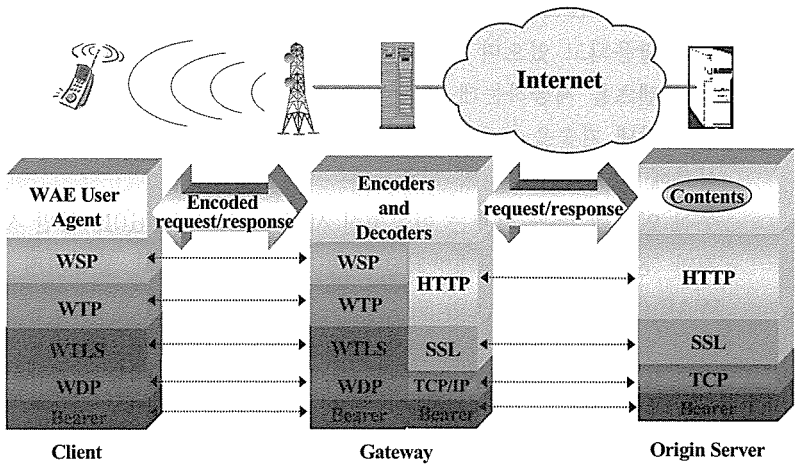
인터넷 비즈니스 시장은 물리적인 시간과 동간의 제약을 받는 유선 인터넷 환경의 시장 개념에서 시간과 공간의 제약을 해소하고 언제(any time), 어디서나(any where) 자유롭게 사용할 수 있는 무선 인터넷 시장 개념으로 점차 이동하고 있다.

그러나 아직까지 현재의 인터넷 환경은 단위 네트워크들을 케이블로 연결한 유선망에 기반을 두고 있기 때문에 시공간적 제한 상황들을 완전하게 해소해 주지 못하고 있는 상황이다. 이러한 이유로 무선 이동 단말기나 PDA와 같은 Post PC를 이용한 무선 인터넷에 관한 연구가 활발하게 진행되고 있는 상황에서 Phone.com, Ericsson, Nokia, Motorola 등의 기업이 WAP 포럼을 결성하여 무선 인터넷 규격인 WAP(Wireless Application Protocol)을 발표하였다. WAP의 구조는 크게 Application Layer(WAE), Session

Layer(WSP), Transaction Layer(WTP), Security Layer(WTLS), Transport Layer(WDP)로 구분할 수 있는데 이는 유선 인터넷상의 각각의 프로토콜 계층을 무선 환경으로 변환시킨 것이다.

기존 HTML 기반의 유선 인터넷 환경의 콘텐츠를 이동 단말기에서 서비스 받기 위해서는 유선 환경에 비해 상대적으로 열악한 전송 속도와 디스플레이 환경을 극복해야만 하는데 WAP 모델에서는 XML 기반의 WML(Wireless Markup Language)과 Gateway를 통해 인코딩된 바이너리 형태의 전송을 통하여 해결하였다. 이 중 WAP Gateway는 WDP(Wireless Transport Layer), WSP(Wireless Session Layer)와 같은 WAP 프로토콜 스택의 구현을 통한 HTTP 프로토콜과 해당 WAP 프로토콜 사이의 변환 및 WML 문서의 인코딩 기능 등을 제공한다.

3.2 시스템 개요



[그림 24] WAP 시스템 개요

무선 인터넷 응용 서비스를 위한 WAP Gateway 프로그램은 사용자 단말기(핸드폰, PDA등)와 Web/WAP 서버 사이에 위치하여 유선 인터넷 환경에서 제공되는 각종 콘텐츠를 무선 인터넷 사용자에게 제공하기 위해 HTTP 프로토콜과 해당 WAP(Wireless Application Protocol) 프로토콜 사이의 변환

및 WML(Wireless Markup Language) 문서의 인코딩(encoding) 기능 등을 제공하는 미들웨어 시스템이다.

구현된 WAP Gateway인 WapGate는 WAP 포럼에서 발표한 WAP 규격 1.2를 토대로 설계되었으며 Java 언어로 구현하여 운영체제나 아키텍처에 상관없이 플랫폼 독립적으로 운영될 수 있도록 하였다. 따라서, 프로그램을 별도로 수정하지 않고도 Windows, Linux, Unix에 모두 설치하여 수행할 수 있으며 이렇게 함으로써 프로그램을 개발하는 데 필요한 시간과 노력과 경비를 줄일 수 있게 하였다. 또한, 개발된 시스템을 Nokia Tool Kit로 테스트한 결과 Kannel 제품보다 성능이 우수한 것으로 나타났다.

WapGate는 단말기와 같은 User Agent의 요청을 쓰레드로서 처리하는데 재사용 가능한 Thread Pool인 ReThread Class의 사용을 통하여 쓰레드 생성과 파괴시 발생하는 오버헤드를 줄임으로서 전체적인 성능과 안정성을 높였습니다. 또한 Java 언어에서 제공되지 않는 버클리 소켓의 select() 함수와 같은 다중 포트 감시 기능을 하는 Port Listener를 구현하여 WAP의 연결형(Connection-Oriented)과 비연결형(Connectionless) 통신 기능 모두를 지원한다.

WapGate 프로그램은 현재 학교에서 무선 인터넷 프로그래밍을 위한 교육자료로 활용되고 있으며 기업에서는 무선 포탈 사이트를 통해 사용자에게 각종 콘텐츠를 제공하는데 활용될 수 있으므로 장래 많은 수요가 예상되는 무선 인터넷 기술을 보유한 고급 인력의 양성과 무선 인터넷 컨텐츠 산업의 활성화에 긍정적인 효과가 기대된다.

무선 인터넷 망을 현재는 이동통신 사업자(011,017,016,019)들만 사용하고 있지만 콘텐츠 업체에 개방되면 이동통신 사업자에게 종속되지 않는 자체 WAP Gateway를 이용한 다양한 무선 인터넷 컨텐츠의 제공이 예상되므로 국내 무선 인터넷 시장은 더욱 활성화될 것으로 판단된다.

3.3 시스템 특징

WAP Gateway 프로그램

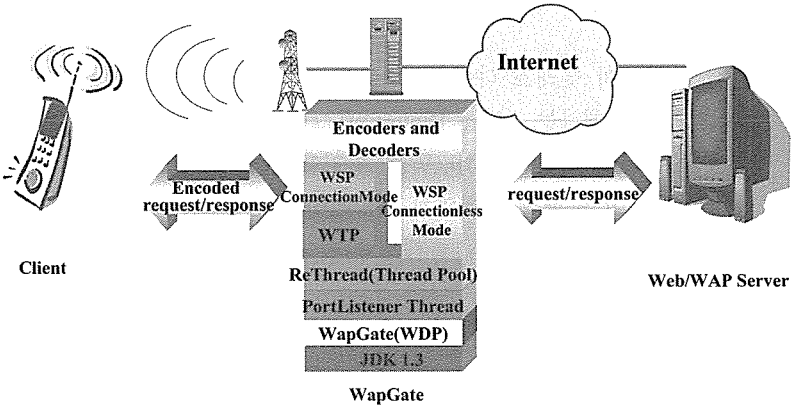
- WAP 규격 1.2를 기반으로 설계
- WAP Protocol Stack 구현
- WML 문서의 Encoding/Decoding 지원

ReThread 클래스
- Thread Pool을 이용한 스레드의 재사용으로 사용자 에이전트의 요구 때마다 새로 생성해야만 하는 스레드 생성시의 오버헤드와 메모리 낭비문제를 해결

DatagramSocketThread 클래스
- Berkley Socket에서는 지원하지만 Java 에서는 지원하지 않던 select() 함수와 같은 다중 포트감시 기능을 구현
- 연결형 통신과 비연결형 통신이 모두 가능해짐

Java 언어로 개발
- 플랫폼(운영체제, 아키텍처 등) 독립적 실행
- 객체 지향의 장점을 이용한 코드의 재사용

3.4 시스템 구성



[그림 25] WapGate 시스템 구성

WapGate는 크게 다음과 같은 다섯 부분으로 구성된다.

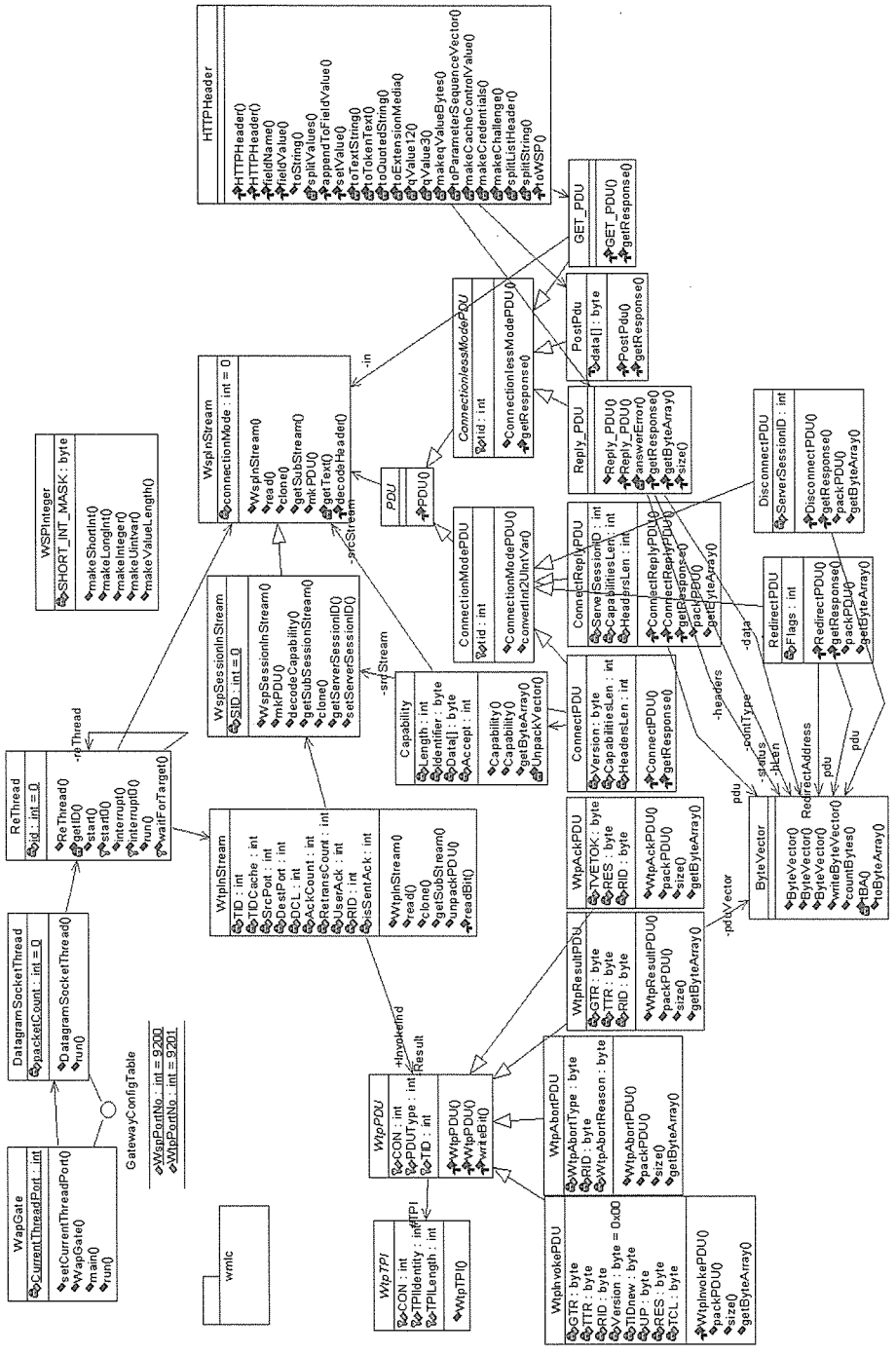
- JDK 1.3 (자바 패키지) : 전체 게이트웨이에서 사용하는 기본 라이브러리로 JDK(Java Development Kit) 1.3을 사용하며 실제로 사용되는 패키지는 java.net(DatagramSocket DatagramPacket, InetAddress),

java.io(InputStream, BufferedInputStream), java.util(Timer, Vector), java.lang(Thread, String, Exception)이다.

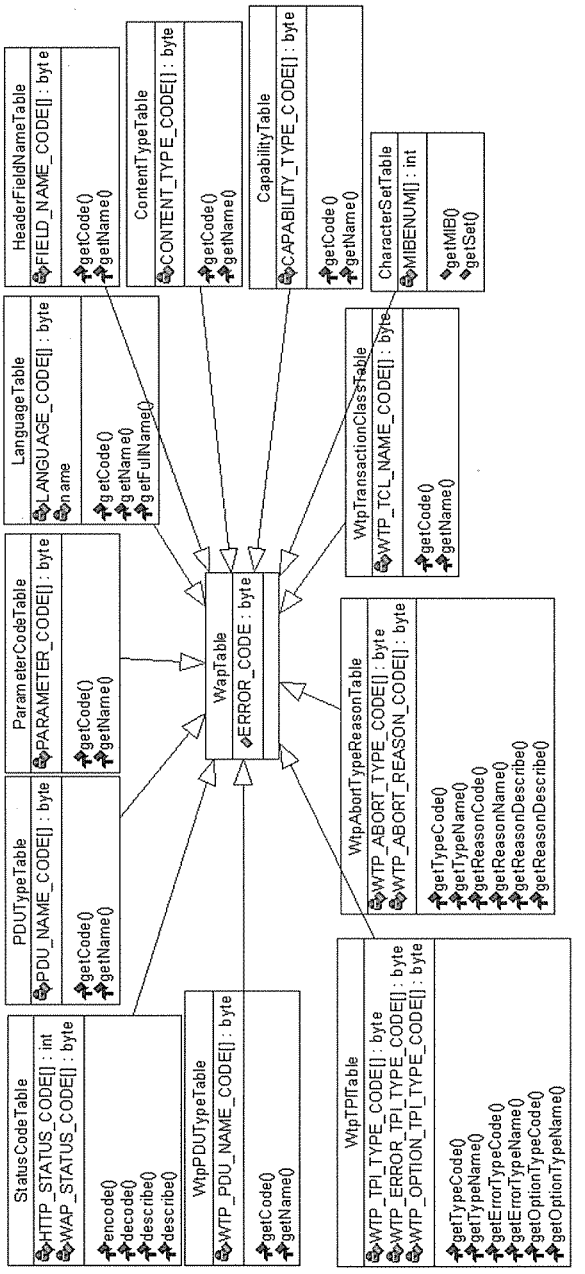
- WDP layer 및 관련 스레드 : WAP 클라이언트로부터 UDP 패킷을 송수신하는 부분으로 Runnable 인터페이스를 구현하여 스레드로 실행되는 메인 클래스인 WapGate와 동시에 여러 포트의 UDP 패킷을 감시하는 PortListener 스레드 그리고 Thread Pool인 ReThread로 구성된다.
- WTP layer - WSP 연결형 통신에 함께 사용되며 트랜잭션 관리 기능을 담당한다.
- WSP layer - WSP layer의 기능은 크게 연결형과 비연결형으로 구별되는데 전자는 WTP layer 함께 사용되어 WSP의 세션 관리 기능을 담당하고 후자는 WSP의 메소드 호출과 응답 그리고 웹 서버와의 통신을 담당한다.
- 인코더 & 디코더 - WML 문서를 무선 전송에 적합한 바이너리 형태의 WML로 상호변환하는 기능을 수행하며 별도의 WMLComplier 패키지로 구성된다.

3.5 프로그램 구성

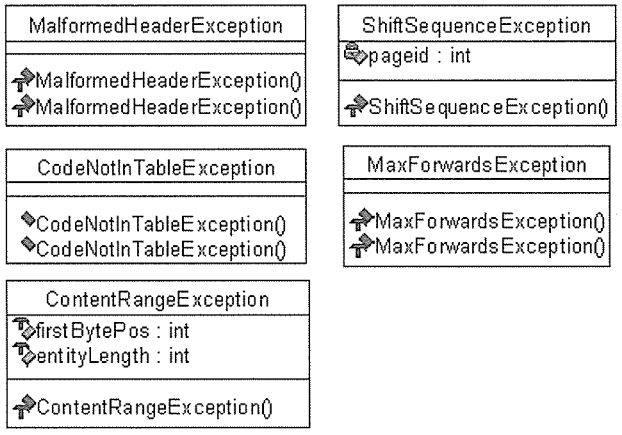
WapGate의 클래스 다이어그램(class diagram)은 4부분으로 구성된다. 주요 기능을 수행하는 Main Diagram([그림 3])과 각종 Code Table로 구성된 Table Diagram([그림 4]), WML처리를 위한 WML 컴파일러 Diagram([그림 5]), 그리고 수행상의 예외처리를 위한 Exception Diagram([그림 6])으로 이루어. 이를 합치면 전체 클래스 다이어그램이 된다.



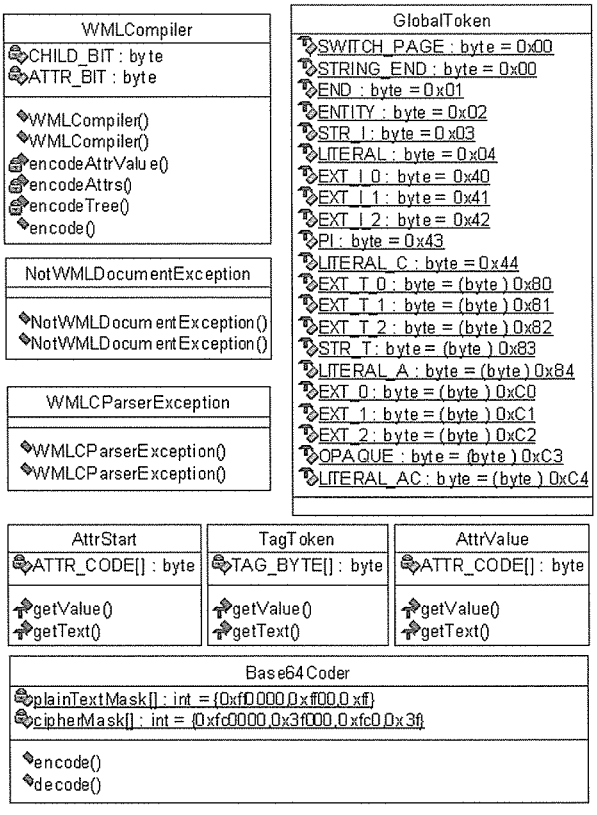
[그림 3] WapGate Main 클래스 다이어그램



[그림 4] WapGate Table 클래스 다이어그램



[그림 5] WapGate Exception 클래스 다이어그램



[그림 6] WML 컴파일러 클래스 다이어그램

4. 실험 및 결과

구현된 WapGate의 기능 및 안정성을 테스트하기 위해 기능 수행 실험과 성능 및 안정성 실험을 실시하였다. 기능 수행 실험은 게이트웨이 내부 처리를 실험하는 것이기 때문에 구현된 WAP 게이트웨이인 WapGate만을 대상으로 실험하였으며, 안정성 실험과 성능 실험은 비교 판단이 필요한 실험이므로 비교 대상을 C 언어로 작성된 리눅스용 WAP 게이트웨이인 Kannel로 설정하였다.

4.1 기능 수행 실험

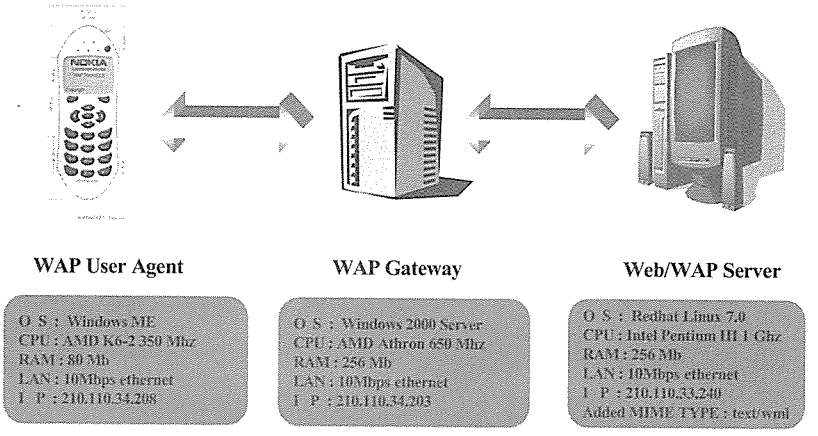
본 실험은 작성된 게이트웨이가 설계한대로 비연결형과 연결형 통신에서 모두 기능 수행이 가능한지 알아보는 실험이다. 때문에 비연결형 실험과 연결형 실험을 별도로 수행하였다.

실험 모델

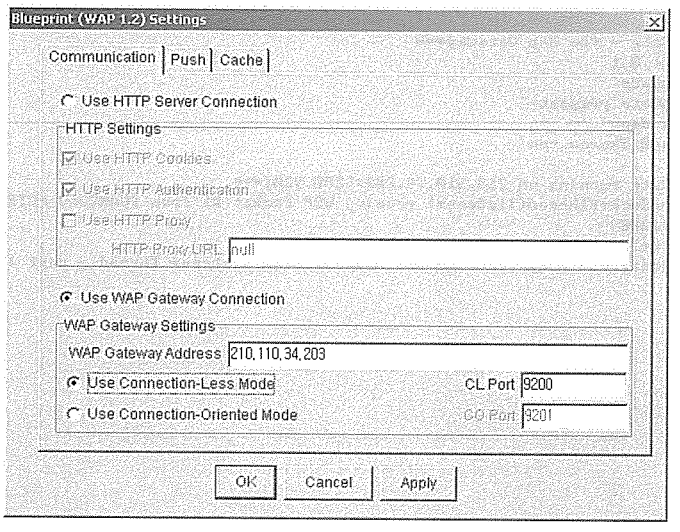
WAP 게이트웨이의 기능을 실험하기 위한 일반적인 실험 모델은 WAP 사용자 에이전트로서의 휴대용 단말기와 WAP 게이트웨이, 그리고 Web/WAP 서버가 상호 통신하는 것입니다. 하지만 휴대용 단말기를 WAP 사용자 에이전트로 사용하는 것은 무선 통신망 라우터인 CSD 라우터(Circuit Switch Datagram Router)를 이용해 직접 실험을 해야하는 어려운 점이 있다. 때문에 본 실험에서는 WAP 시뮬레이터인 노키아 WAP 툴킷 2.0을 WAP 사용자 에이전트로 사용하였다.

비연결형 통신 실험

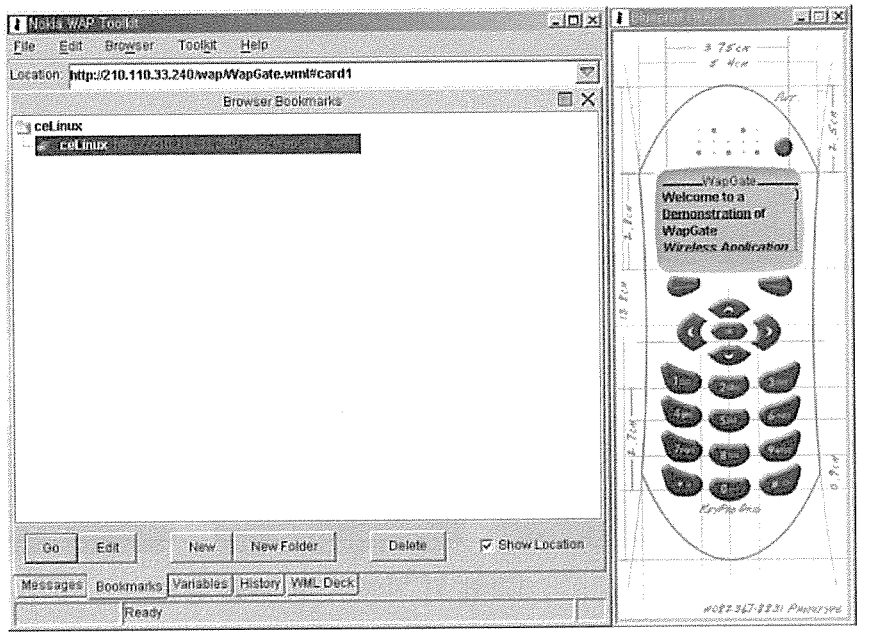
비연결형 통신을 실험하기 위해 아래의 [그림 8]과 같이 노키아 WAP 툴킷을 설치하였다. [그림 9]은 http://210.110.33.240 /wap/WapGate.wml 요청을 수행한 결과이며, [그림 10]은 위의 요청을 처리하는 WAP 게이트웨이의 수행 화면이다.



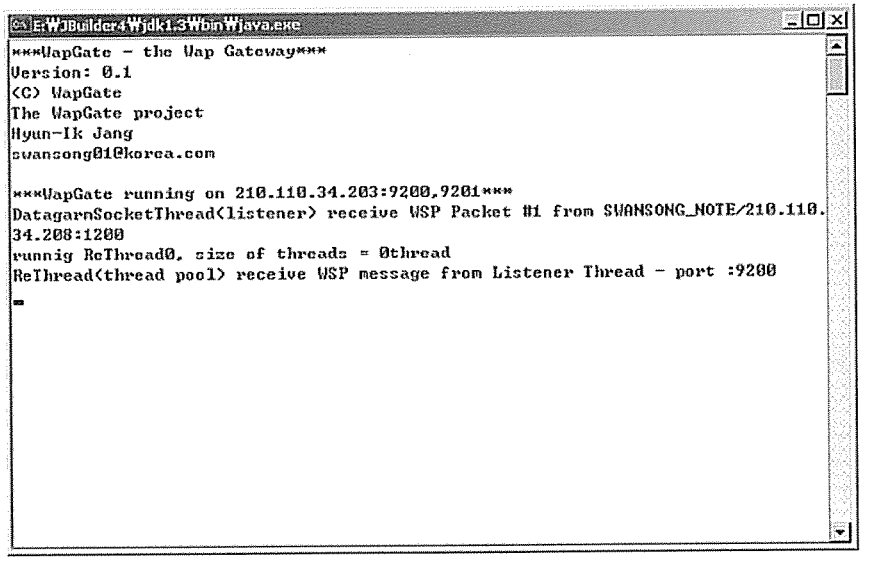
[그림 7] 기능 수행 실험 환경



[그림 31] 비연결형 설정

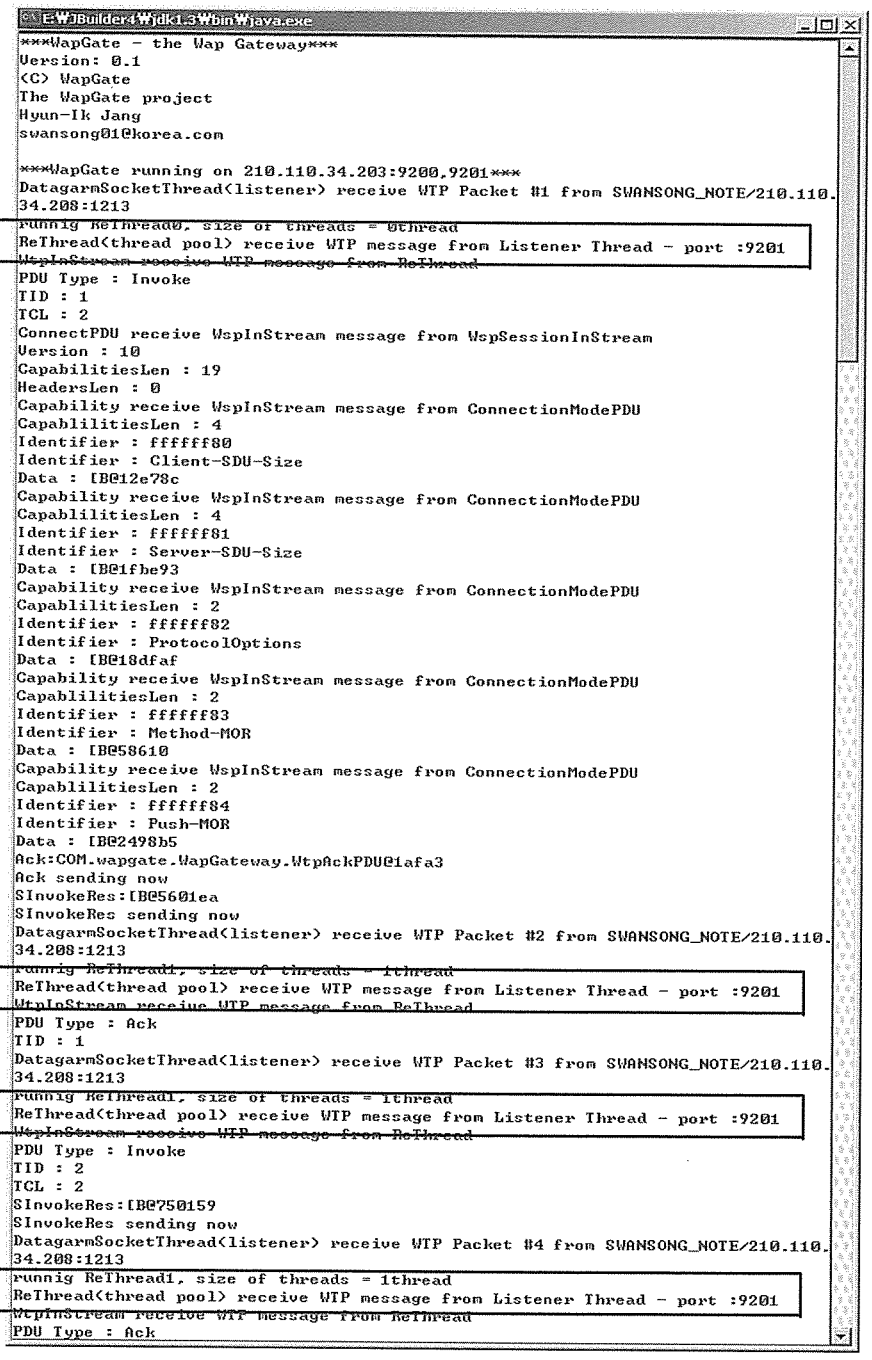


[그림 32] 비연결형 실험 수행 결과



[그림 33] 비연결형 요청에 대한 게이트웨이 처리 화면

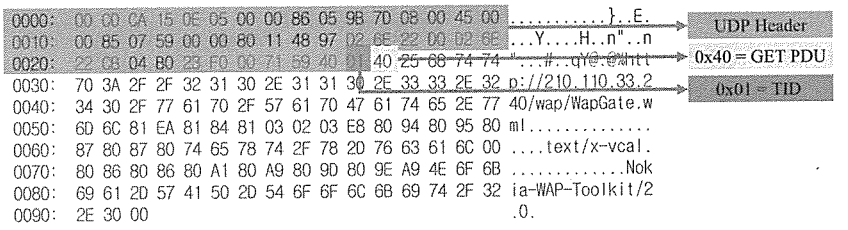
연결형 통신 실험



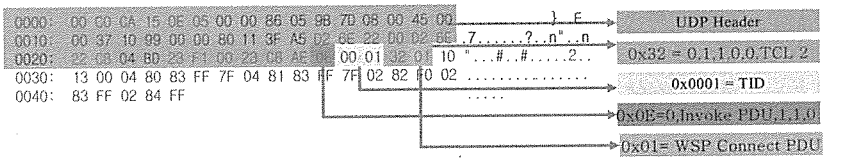
[그림 34] 연결형 요청에 대한 게이트웨이의 처리 화면

연결형 통신 실험은 노키아 WAP 툴킷의 설치를 [그림 8]에서 Use Connection-Oriented Mode(CO port 9201) 옵션을 선택하는 것을 제외하고는 비연결형 실험과 동일하게 진행되며 결과 또한 [그림 9]와 같다. [그림 11]은 연결형 요청에 대한 WAP 게이트웨이의 처리 화면이다.

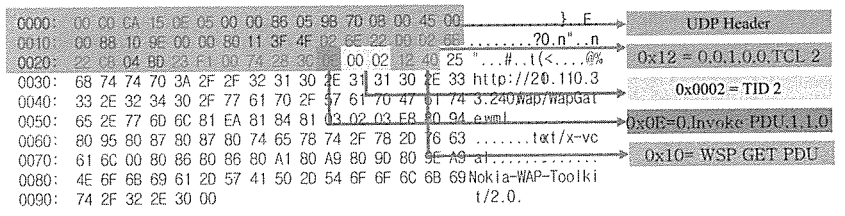
패킷 분석



[그림 12] 비연결형 통신 패킷

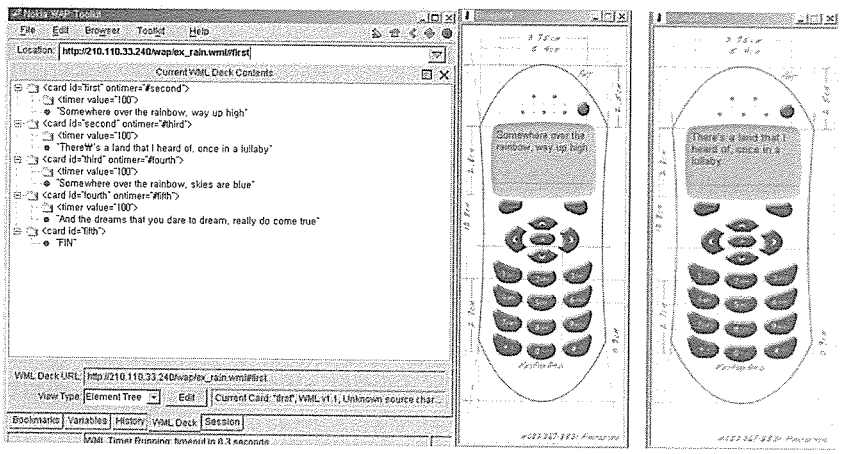


[그림 13] 연결형 Connect 패킷



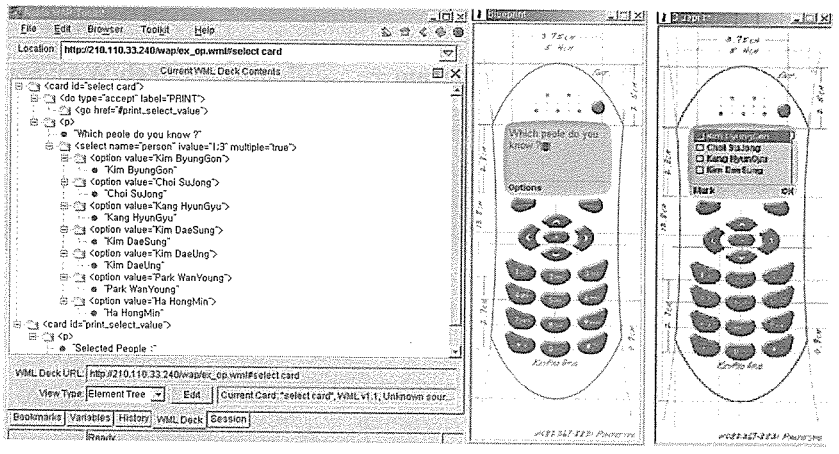
[그림 14] 연결형 Get 요청 패킷

Timer를 이용한 CARD 전환의 예제 실행 결과



[그림 15] Timer 예제

아이템을 선택하는 예제 실행 결과



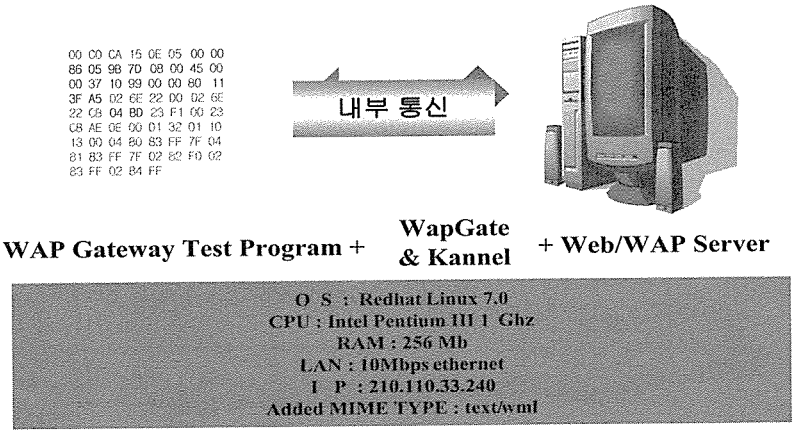
[그림 16] Option 선택 예제

4.2 성능 및 안정성 실험

성능 및 안정성 실험은 작성된 게이트웨이가 얼마나 많은 동시 요청을 안정적으로 처리할 수 있는가와 얼마나 빠른 시간에 사용자 에이전트의 요구에 응답을 할 수 있는가에 대한 실험이다.

일반적으로 서버 측 프로그램들은 적은 수의 요청에 대해서는 안정적으로 작동을 하지만 동시에 많은 요청이 발생할 경우에는 그 처리과정에서의 오류로 인해 프로그램의 기능이 중단되거나 혹은 프로그램 자체가 종료되는 경우가 많이 발생한다. 하지만 서버 측 프로그램들은 자신의 처리 한도 이상의 요청이 들어와서 그 요청을 일시적으로 처리하지 못하는 경우가 발생하는 것은 수용이 가능하나 그로 인해 서비스가 중단되어서는 안된다.

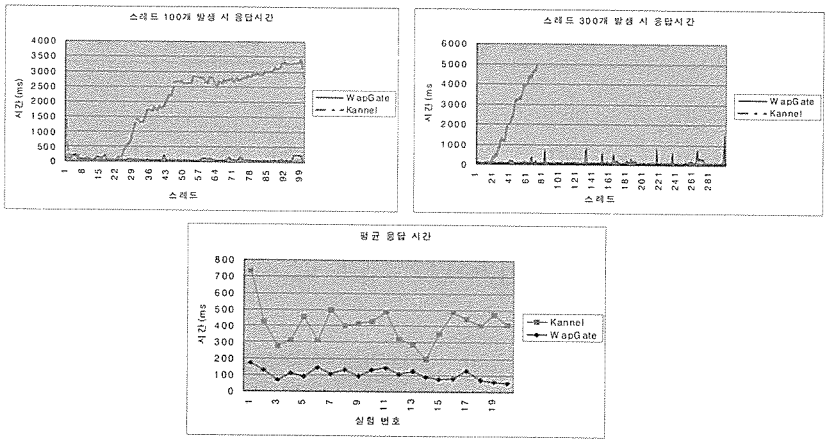
실험 모델



[그림 17] 성능 수행 실험 환경

실험 결과

다음은 성능 수행 실험을 하기 위한 환경과 비교 대상 Gateway인 Kannel과의 실험 결과를 그래프로 나타낸 것이다.



[그림 45] 실험 결과 (WapGate VS. Kannel)

실험 결과를 살펴보면 처음 100개 스레드를 대상으로 한 실험은 WapGate와 Kannel 모두 정상적인 수행이 가능했으며 응답 시간은 WapGate 쪽이 빠른 것을 알 수 있다. 또한, 200개 이상 스레드를 대상으로 한 실험들에서 Kannel 게이트웨이가 일부 스레드의 요청에 대해서만 응답한 후에 시스템이 다운되는 현상이 나타나 더 이상 비교하는 것이 의미가 없게 되었다. 스레드 100개 발생 실험을 20회 반복하여 게이트웨이별로 스레드별 평균 응답시간의 변화 추이를 그래프로 정리하면 3번째 그래프와 같다.

종합해서 살펴 보면 자바로 구현된 WAP Gateway인 WapGate의 경우 실험한 모든 부분에서 Kannel의 WAP Gateway 보나 나은 성능을 보여주었다.

5. 개발 환경, 언어 및 도구

- OS : Windows 2000 Server, Redhet Linux 7.0
- CPU : AMD Athron 650MHz, Intel Pentium III ·1GHz
- RAM : 256M LAN : 10Mbps Ethernet
- Compiler : JDK 1.3 (java.sun.com)
- Web Server : Apache 1.3.17 (www.apache.org)
- Client Test Tool : Nokia WAP Tool Kit 2.