LINUX 기반 WAP 게이트웨이/서버 통합구조의 설계 및 구현

송 병 권† · 오 태 안††

요 약

최근 무선 인터넷 서비스에 대한 관심이 고조되면서 관련 기술 개발이 활발히 진행되고 있다. 현재 가장 유력한 국제표준 중 하나로 인식되어 가고 있는 WAP(Wireless Application Protocol)의 규격에 따르면 이동 단말과 WAP 서버는 WAP 게이트웨이를 통하여 통신하도록 되어있다. 본 논문은 리눅스 기반에서 WAP 게이트웨이와 서버가 통합적으로 지원되는 I(Intelligent) WAP 게이트웨이/서버 플랫폼의 설계 및 구현에 관한 것이다. 제안된 IWAP 플랫폼은 WAP 게이트웨이, JAVA기반의 서버 개발 환경, WML 툴킷 및 MUI(Management User Interface) 등 크게 4개의 모듈로 구성되고, 베어러(bearer)망으로 SMSC(Short Message Service Center)와 CSD(Circuit Switched Data) 라우터를 고려하였다.

A Design and Implementation of WAP Gateway/Server Integration Structure based on Linux

Byungkwen Song† · Taean Oh††

ABSTRACT

As the interest in the wireless internet services is increasing recently, the related technology development is in active progress. According to WAP (Wireless Application Protocol) specification which is currently considered as one of the most powerful international standardizations, mobile terminal and WAP server are supposed to communicate through WAP Gateway. This paper is about the design and implementation of IWAP platform where WAP Gateway and Server are integrated and supported based on Linux. The proposed IWAP platform broadly consists of four modules like WAP Gateway, JAVA based Server development environment, WML Tool-Kit, and MUL (Management User Interface) and for bearer network, SMSC (Short Message Service Center) and CSD (Circuit Switched Data) router are considered.

키워드 : IWAP 플랫폼(I(Intelligent) WAP Platform), WAP 게이트웨이/서버(WAP Gateway/Server), 베이러 망(Bear Network)

1. 서 론

세계적으로 이동통신 가입자가 급속히 증가하고 인터넷이 일반화되면서 휴대성과 이동성으로 대표되는 무선 인터넷 서비스가 보편화되고 있다. 현재 무선 인터넷 서비스를 지원하기 위한 표준화가 WAP 포럼, W3C(World Wide Web Consortium) 및 마이크로소프트사 등을 중심으로 진행되고 있다. 그 중에서 WAP은 에릭슨, 모토롤러, 노키아, 폰닷컴 등 유력한 이동통신 업체들을 중심으로 AT&T, 벨 사우스, IBM을 포함하여 200여 개의 통신 및 컴퓨터 업체들이 참여하여 무선 인터넷 관련 표준 프로토콜로 될 가능성이 높다. WAP은 무선 구간 상에서 인터넷 서비스를 이용할 때 발생되는 이동 단말기 상의 한계인 CPU 연산 능력의 제한, 낮은 메모리, 제한된 전력 소모, 제한된 사용자 인터페이스의 문제와 무선 구간 상의 문제점인 낮은 대역폭, 긴 지연율, 낮은 연결 안정성 등의 문제점들을 해결하기 위해 개발된 하나의 통신 규약으로 트랜잭션, 세션 그리고 응용 계층으로 구성된다. WAP 모델은 무선 환경의 문제점을 극복하기 위하여, 유선망 즉 핵심망에서는 인터넷 관련 표준 프로토콜을 사용하여 기존 인터넷 서버와 연결되고 무선 구간에서는 무선 환경에 적합하도록 설계된 프로토콜을 이용하여 핵심망과 연결되는 연결 분리(split connection) 구조를 채택하고 있다. 따라서, 이러한 분리된 연결 구조를 연동하기 위하여 WAP 게이트웨이가 필요하며, 이동 단말은 항상 WAP 게이트웨이를 통하여 기존 인터넷 및 WAP 서버와 연결된다[1].

기존에 발표된 대표적인 관련 연구로는 참고문헌 [20]에서 서술하고 있는 Kannel WAP 게이트웨이와 Phone.com [21]의 UP.Link 등이 있다. Kannel WAP 게이트웨이는 WSP (Wireless Session Protocol)/WTP(Wireless Transaction

† 종신회원 : 서경대학교 정보통신공학과 교수
†† 정 회 원 : 트라이콤텍(주)
논문접수 : 2002년 6월 5일, 심사완료 : 2002년 12월 21일

Protocol) 스택을 포함한 WAP 박스(box)와 베어러망과의 연결을 위한 베어러 박스로 구성된다. 베어러 박스는 이동 단말로부터 전달된 데이터그램을 WAP 박스로 적절히 부하를 분산시키는 기능을 수행하고, WAP 박스는 WSP/WTP 프로토콜 스택을 가지며 이동 단말로부터 WML(Wireless Markup Language) 문서 요구를 처리하기 위하여 해당 요구를 HTTP(HyperText Trasfer Protocol) 형태로 변경하여 서버로 전달하고, WML 서버로부터 전달된 WML 문서는 WML 컴파일러를 이용하여 이진 WML 형태로 변경한 후 베이러 박스에게 전달한다. 따라서, Kannel WAP 게이트웨이는 모든 데이터가 하나의 베이러 박스를 경유하여 송수신되는 구조이므로 동시에 수행하는 클라이언트 수가 일정 수준 이상으로 증가하면 베이러 박스에서 병목현상이 발생하여 전체 WAP 게이트웨이 센터의 처리 능력에 문제가 생길 수 있다. Phone.com의 UP.Link는 기본적으로 UP.Link 게이트웨이, UP.Link 서버 및 UP.Link 어플리케이션 등의 3가지 모듈로 구성되어 있다. UP.Link 게이트웨이는 WAP 규격에 따른 프로토콜을 지원하고 있으며 UP.Link 서버는 푸시(push) 서버, 팩스 서버, 과금 서버 및 컨텐츠 변환기 등으로 구성되고 UP.Link 어플리케이션은 다양한 형태의 응용 서비스 개발 환경을 제공한다.

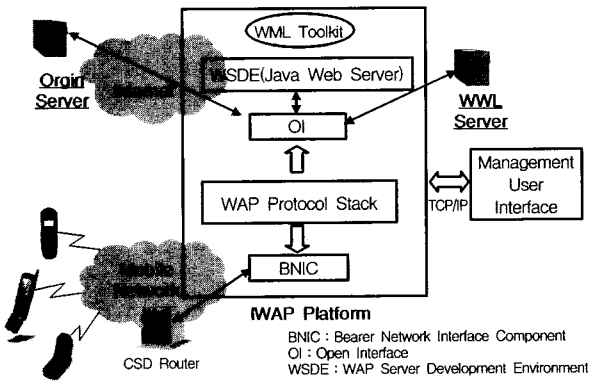
본 논문은 리눅스 기반에서 WAP 게이트웨이 및 서버가 통합적으로 지원되는 IWAP 플랫폼의 설계 및 구현에 관한 것이다. 따라서 IWAP 플랫폼은 이동 핵심 망 내에 존재하는 WAP 게이트웨이와의 연동 없이도 WML 기반의 WAP 서비스 개발자나 컨텐츠 제공자들에게 전자상거래, 인터넷 엑세스, 메일 서버와 같은 다양하고 독창적인 응용 서비스 개발을 용이하게 할 수 있는 장점을 제공한다. 또한 본 플랫폼은 베어러 망과의 연결을 위한 베어러 박스를 통하여 무선망에서의 서로 다른 두 방식의 메시지, 즉 CSD 라우터 및 SMSC 인터페이스를 통한 메시지를 모두 수용할 수 있도록 구현되었다.

제안된 IWAP 플랫폼[32]은 WAP 게이트웨이, WAP 서버 개발 환경을 제공하는 Java 기반의 웹 서버, WML 툴킷 및 MUI 등 크게 4개의 부분으로 구성되고, 베어러 망으로는 CSD 라우터 및 SMSC를 고려하였다.

본 논문의 구성은 다음과 같다. 2장에서는 IWAP 플랫폼 전체 구조를 기술하고 3장에서 IWAP 플랫폼 내부 구조를 살펴본다. 4장에서는 IWAP 시험 환경 및 현재 진행되고 있는 연구에 대하여 기술하고 5장에서 결론을 맺는다.

2. IWAP 플랫폼 전체 시스템 구조

제안된 IWAP 플랫폼은 WAP 게이트웨이, Java 기반 WAP 서버 개발 환경, WML 툴킷 및 MUI 등 크게 4개의 부분으로 구성된다. 또한 하부 베어러 망은 WAP 게이트웨이의 BNIC(Bearer Network Interface Component)를 통하여 CSD 라우터와 연동되고, OI(Open Interface) 모듈을 통하여 ① 기존 인터넷 서버, ② WML 및 WML 스크립트 서비스를 제공하는 WML 서버, ③ WAP 서버 개발 환경을 제공하는 Java 기반 Web 서버와 HTTP 데몬을 통하여 연결된다((그림 1) 참조).

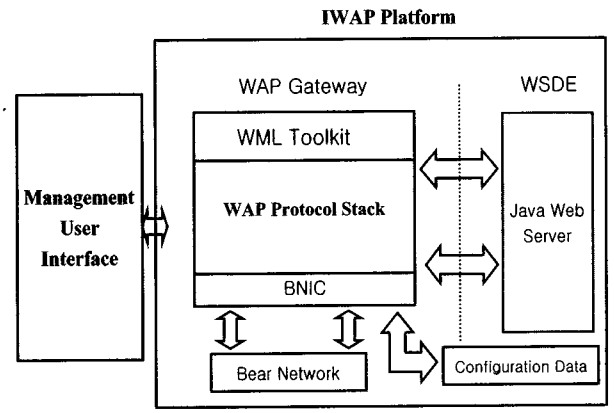


(그림 1) IWAP 플랫폼 전체 시스템 구조 및 망 구성도

IWAP 플랫폼의 서비스 수행 절차는 다음과 같이 요약될 수 있다. 무선망을 통해 전달된 WAP 단말기의 서비스 요구는 IWAP 플랫폼의 베어러 망 인터페이스인 BNIC를 통해서 WAP 프로토콜 스택으로 전달된다. WAP 프로토콜 스택은 WSP를 이용하여 WAP 단말기와 세션을 성립하고 WAP 단말기에서 요구한 서비스를 HTTP 형태로 변경한 후 OI 모듈을 통하여 인터넷상의 특정 서버에게 전달하거나 IWAP 플랫폼 내부에 포함된 Java 기반 WAP 서버 개발 환경(WSDE : WAP Server Development Environment)으로 전달한다.

3. IWAP 플랫폼 내부 구성 요소

IWAP 플랫폼은 WAP 게이트웨이, WSDE을 제공하는 Java 기반 Web 서버, WML 툴킷 및 MUI로 구성된다((그림 2) 참조).



(그림 2) IWAP 플랫폼 내부 구조

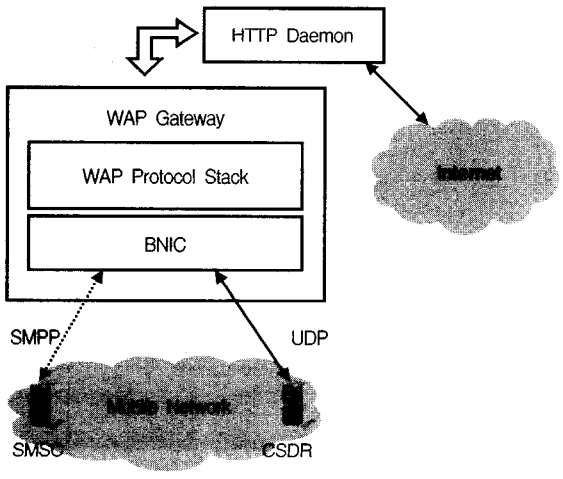
WAP 게이트웨이는[4-8]에 제안한 WAP 프로토콜 스택을 제공하고, WSDE는 Java 기반 WAP 관련 응용 서비스 개발 환경을 제공하며, WML 툴킷은 WML 소스 코드의 에러 및 문법 체크 기능과 해당 소스 코드를 바이트 코드 형태로 변환하는 기능을 수행한다. 그리고 MUI는 WAP 게이트웨이 및 WSDE를 시작 또는 종료하거나 알람(alarm) 모니터링과 같은 IWAP 플랫폼전체를 관리하는 기능을 수행한다. 특히 본 플랫폼 내부에서 WDSE 지원을 위한 Java 웹 서버는 Java 셔블릿(servlet)을 기반으로 설계되어져있기 때문에 많은 클라이언트의 요구사항을 신속히 처리할 수가 있으며, Java 기반의 다양한 응용 서비스를 개발할 수 있다. 셔블릿 서비스 수행절차 및 컨테이너 구동은 3.2절에서 자세히 소개한다.

IWAP 플랫폼의 각 부분별 기능은 다음과 같다.

3.1 WAP 게이트웨이 모듈

WAP 게이트웨이 모듈에서 지원되는 프로토콜은 다음과 같다.

- WDP(Wireless Datagram Protocol)
  [8]에서 권고한 CSD 라우터 인터페이스 기능 제공
- WTP(Wireless Transaction Protocol)
  [6]에서 권고한 Class 0, Class 1 및 Class 2 제공
- WSP(Wireless Session Protocol)
  [5]에서 권고한 연결 지향 및 비연결 지향형 서비스 제공
- WTLS(Wireless Transport Layer Security)
  널(null) 계층
- WAE
  [4]에서 권고한 WML 툴킷 기반의 WAP 애플리케이션 환경



(그림 3) IWAP 플랫폼의 하위 및 상위 연결구조

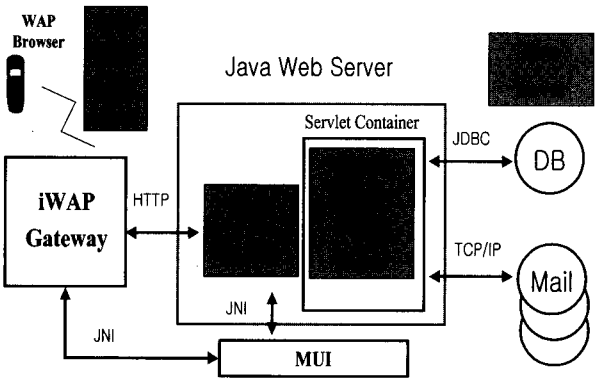
베이러 망과의 인터페이스로 UDP 프로토콜을 통한 CSD 라우터 인터페이스 및 SMSC와의 인터페이스를 위하여 SMPP (Short Message Peer to Peer Protocol)[22, 23] 모듈을 제공한다.

(그림 3)은 IWAP 플랫폼의 하위 및 상위 연결구조를 나타낸다.

3.2 WSDE(WAP Service Development Environment)

WSDE는 Java VM(Virtual Machine)에서 제공되는 셔블릿[24, 25] 기반의 스레드(thread)로 구현하였다. 따라서 기존 웹 서버에서 제공하는 CGI(Common Gateway Interface)의 단점인 클라이언트 증가에 따른 성능 저하 현상을 방지할 수 있다. 또한 셔블릿은 동적인 확장성을 지원하는 컴포넌트 형태의 클래스를 생성하므로 객체지향 특성에 따른 코드의 재사용 성이나 확장성 등을 효율적으로 지원할 수 있다.

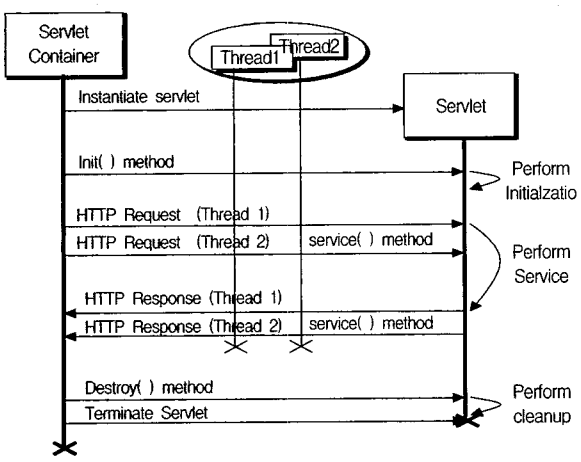
다음 (그림 4)는 WSDE 전체 구조를 나타낸다.



(그림 4) WSDE 전체 시스템 구조

WSDE의 셔블릿 서비스는 WAP 단말기로부터의 셔블릿 요구로부터 시작된다. 즉 WAP 단말기의 셔블릿 요구는 HTTP 형태로 Java Web 서버로 전달된다. 셔블릿 요구를 처리하기 위하여, Java Web 서버는 셔블릿 컨테이너(container)에 구동하여 셔블릿 인스턴스(instance)를 생성한다. 생성된 인스턴스는 SLC(Servlet Life Cycle)을 통해서 셔블릿 응답을 HTTP 형태로 WAP 단말기에게 전달한다.

(그림 5)는 WSDE에서 제공되는 셔블릿 서비스의 호출 및 실행흐름에 대한 것을 MSC(Message Sequence Chart) 형태로 나타낸 것이다. (그림 5)에서와 같이 셔블릿 요구에 대하여 셔블릿 컨테이너는 스레드와 셔블릿 인스턴스를 생성한다. 생성된 스레드는 셔블릿 요구를 SLC에 따라 처리하고, 또 다른 WAP 단말기가 서비스를 요청하면 서버는 셔블릿의 새로운 스레드를 생성해 해당 요청을 처리한다.



(그림 5) 셔블릿 서비스 흐름

SLC의 메쏘드(method)는 다음과 같다.

- **public void *init* (ServletConfig config) throws ServletException**

init 메쏘드는 ServletConfig 속성을 갖고 객체를 통하여 초기화 인자들을 읽을 수 있다. 서버가 셔블릿을 로드(load)할 때 init 메쏘드를 실행하고, 셔블릿 인스턴스 생성시 최초 단 한번만 실행한다.

- **public void *service*(ServletRequest request, ServletResponse response) throws ServletException, IOException**

service 메쏘드는 요청을 처리하기 위해 호출되거나 WAP 단말기로부터 "Get" 이나 "Post" 명령을 수행하며 그 예는 아래와 같다.

```
// Get
public void doGet (HttpServletRequest request, HttpServlet
                   Response response)
throws ServletException, IOException {
  ......
}
// Post
publice void doPost (HttpServletRequest request, HttpServlet
                     Response response)
throws ServletException, IOException {
  ......
}
```

- **public void *destroy*( )**

destroy이 메쏘드는 셔블릿 서비스가 끝나기 바로 전에 한번 호출된다.

3.3 MUI (Management User Interface)

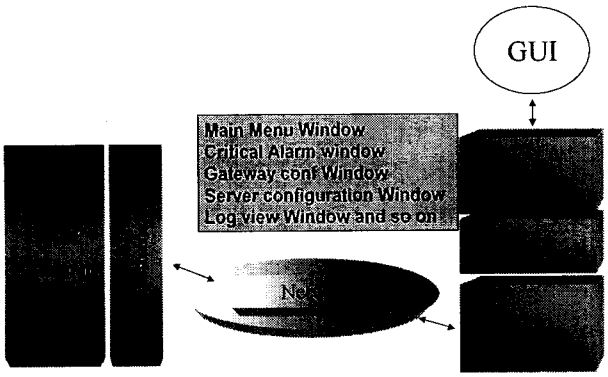
MUI는 운용자에게 IWAP 플랫폼 관리를 위하여 Java 기반의 GUI(Graphical User Interface)를 제공한다. MUI에서 제공하는 주요 기능들은 다음과 같다.

- IWAP 플랫폼 시작 및 종료
- IWAP 구성(configuration)
- 구성 데이터 접근 및 변경

MUI는 운영자에게 구성 데이터 관리 기능을 제공한다. 또한 운영자는 새로운 구성데이터를 다시 로드할 수 있다.

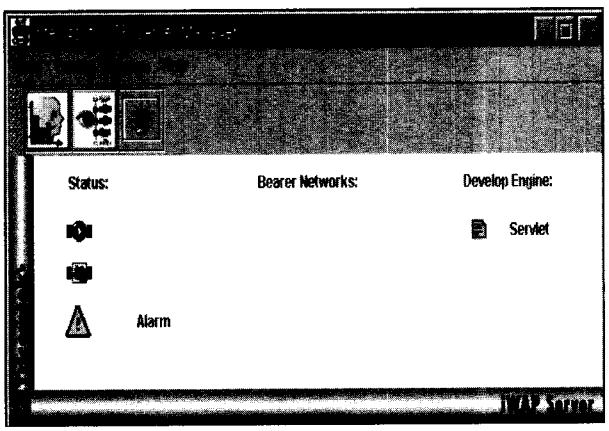
- 베이러 망 초기화 및 트레이스(trace) 단계 설정
- 도움말
- 알람 모니터링(monitoring)

(그림 6)은 MUI 시스템 내부 구조를 나타낸다. MUI 시스템은 Java 2 스윙(swing)[29]을 사용하고 TCP/IP를 이용하여 IWAP 플랫폼과 연결된다. IWAP 플랫폼은 C-언어로 구현되어있기 때문에 Java 클래스와 C-언어의 데이터 타입 변경을 위하여 JNI(Java Native Interface)[28] 모듈을 사용하였다.

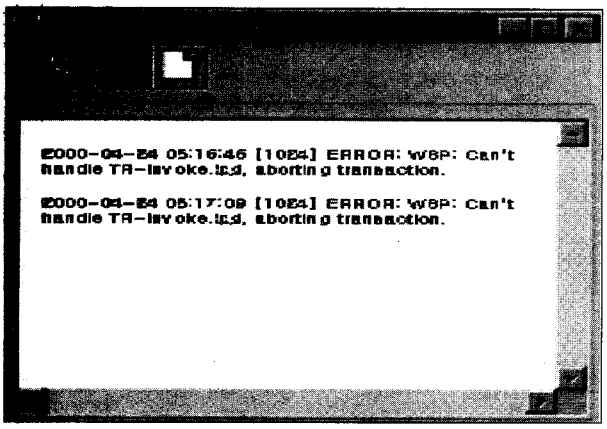


(그림 6) MUI 시스템 내부구조

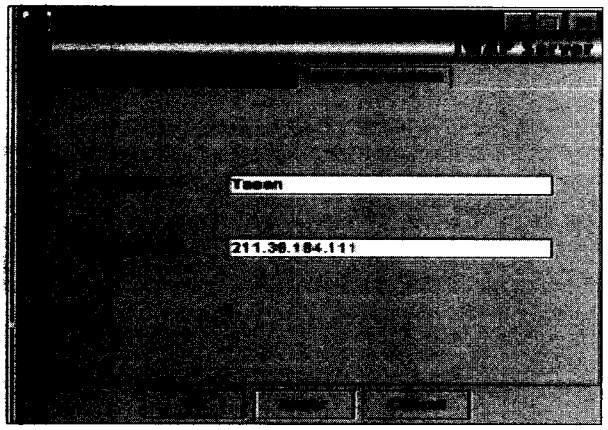
MUI 시스템의 운영자의 인터페이스는 여러 종류의 Java 클래스로 구성되어 있다. 아래 (그림 7)은 메인 메뉴 클래스에 대한 것이고, (그림 8)은 알람 및 구성 메뉴에 대한 예이다.



(그림 7) MUI의 메인 메뉴



(a) Critical Alarm window



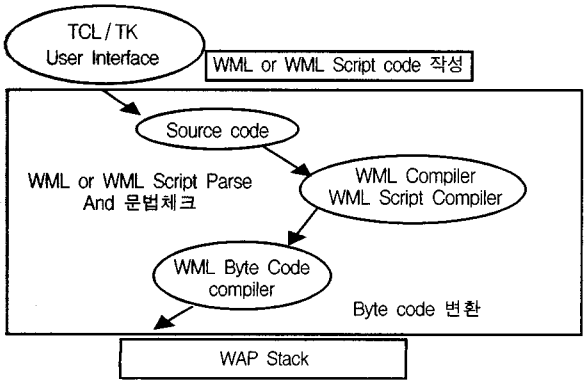
(b) iWAP Server configuration window

(그림 8) 알람 및 구성 메뉴

3.4 WML 툴킷

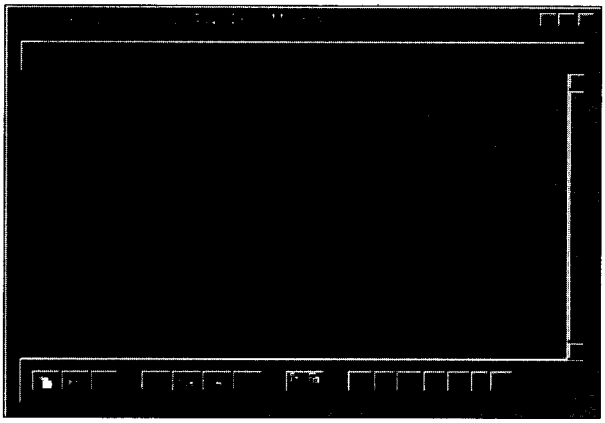
WML 툴킷은 WAE[4] 개발 환경을 제공한다. TCL/TK [31]로 구현된 사용자 인터페이스를 통해 작성된 WML 및 WML 스크립트 소스 코드는 WML 툴킷을 이용하여 에러 및 문법 체크된 후에 바이트 코드로 변환된 후 WAP 프로토콜 스택을 통해 WAP 단말기로 전달된다.

(그림 9)은 WML 툴킷의 실행구조를 나타낸다.



(그림 9) WML 내부 실행구조

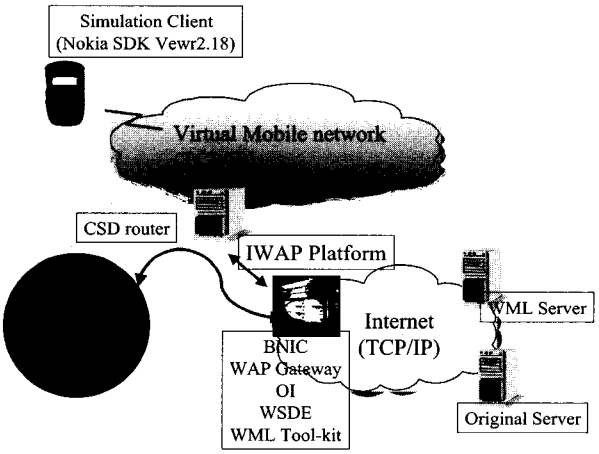
(그림 10)은 TCL/TK로 구현된 WML 툴킷의 사용자 인터페이스다.



(그림 10) WML 툴킷 사용자 인터페이스 예

4. 실험 및 토의

IWAP 플랫폼의 기능 실험은 시뮬레이션 WAP 클라이언트로 컨텐츠 전송을 위한 가상의 이동 망 환경 및 WAP 컨텐츠 서비스 제공을 위한 서버 환경을 각각의 TCP/IP network 구성을 통해서 구현하였다. 가상 이동 망 환경은 CSD 라우터를 시뮬레이션하기 위한 기존 라우터와 Nokia SDK Ver2.18[19]을 사용한 WAP 클라이언트로 구성하였고, WAP 컨텐츠 서비스 제공을 위한 서버 환경은 IWAP 플랫폼, WML 서버 및 기존 Web 서버로 구성하였다. (그림 11)은 두 개의 TCP/IP network으로 구성된 IWAP 플랫폼 테스트 환경에 관한 것이다.



(그림 11) IWAP 플랫폼 테스트 환경

IWAP 플랫폼의 기능 실험은 WAP 클라이언트에게 이미지 및 텍스트를 포함하는 컨텐츠 서비스 제공 실험으로 수행된다. 먼저 IWAP 플랫폼에서 제공되는 WML 툴킷을 사용한 index.wml 컨텐츠는 아래와 같다.

```
<?xml version = "1.0"?>
<!DOCTYPE wml PUBLIC "-// WAPFORUM // DTD WML
 1.1 // EN" "http://211.39.184.15/index.wml">
<!-- Source Generated by WML Deck Decoder -->
<wml>
  <template>
    <do type = "prev">
      <noop/>
    </do>
  </template>
  <card id = "first" ontimer = "# select"> <timer value = "30"/>
    <do type = "Options" label = "Reserved">
      <go href = "# copyright"/>
    </do>
  <p align = "center">
    <img src = "tri.wbmp" alt = "Tricomtek"/>
    <br/>   <br/>
    <b> WAP Ver1.0 </b>
  </p>
</card>
  <card id = "select" title = "Tricomtek">
    <do type = "Options" label = "Reserved" >
      <go href = "# copyright"/>
    </do>
      <p> <a href = "overview.wml"> Overview </a> <br/>
      <a href = "introduction.wml"> Introduction </a> <br/>
      <a href = "news.wml"> News </a> <br/>
      <a href = "tricomtek.wml"> Tricomtek </a> <br/>
      </p>
  </card>
  <card id = "copyright">
    <onevent type = "ontimer">   <prev/>
    </onevent>
    <timer value = "30"/>
    <p align = "center">
      <br/>
      <br/>
      <small> Copyright & # xA 9 ; 2000
        <br/>
        Tricomtek Co.
        <br/>
        All rights reserved. </small>
    </p>
  </card>
</wml>
```

초기화면인 index.wml은 로고(logo) 그림 파일 및 사용자에게 선택 메뉴를 제공하는 화면으로 디자인되었다. 사용자 메뉴에는 overview, introduction, news 및 tricomtek를 제공해준다. 물론 이러한 서비스 제공 메뉴도 WML 툴킷을 사용하여서 WML 코드로 디자인한다.

아래는 메뉴 중 기상정보 제공 실험을 위해서 생성된 news.wml 컨텐츠이다.

```
<?xml version = "1.0"?>
<!DOCTYPE wml PUBLIC "-// WAPFORUM //
 DTD WML 1.1 // EN" "http://www.wapforum.org/DTD/wml
 _1.1 xml">
  <wml>
    <template>
      <do type = "prev">
        <prev/>
      </do>
    </template>
      <card id = "card1" title = "Weather Forecast">
      <p>
      <table columns = "3" align = "LCC">
      <tr><td> Date </td> <td> F & apos ; cast </td> <td>
          T &# xB0 ; C </td>
      </tr>      <tr>
      <td> M 6/7 </td> <td> <img src = "rainy.wbmp"
          alt = "rain"/> </td>
      <td> 25 &# xB0 ; C </td>
      </tr>      <tr>
      <td> T 6/8 </td> <td> <img src = "partcldy.wbmp"
          alt = "part cldy"/> </td>
      <td> 27 &# xB0 ; C </td>
      </tr>      <tr>
      <td> W 6/9 </td> <td> <img src = "cloudy.wbmp"
          alt = "cloudy"/> </td>
      <td> 24 &# xB0 ; C </td>
      </tr>      <tr>
      <td> T 6/10 </td> <td> <img src = "rainy.wbmp"
          alt = "rainy"/> </td>
      <td> 28 &# xB0 ; C </td>
      </tr>      <tr>
      <td> F 6/11 </td> <td> <img src = "sunny.wbmp"
          alt = "sunny"/> </td>
      <td> 29 &# xB0 ; C </td>
      </tr>
      </table>
      </p>
      </card>
</wml>
```

이상과 같이 생성된 WML 컨텐츠들을 WSDE에 저장한다. 저장된 WML 컨텐츠의 서비스를 위해서 WSDE의 MINE 타입에 아래와 같은 WML 관련 서비스를 등록한다.

- .wml text/vnd.wap.wml
- .wbmp image/vnd.wap.wbmp
- .wmlc application/vnd.wap.wmlc
- .wmls text/vnd.wap.wmlscript
- .wmlsc application/vnd.wap.wmlscriptc

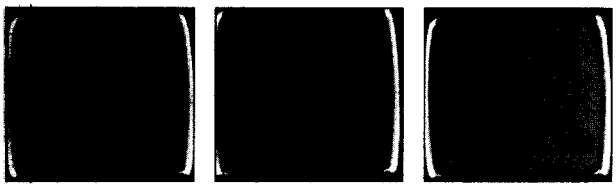
이렇게 저장된 컨텐츠는 WAP 클라이언트들에게 가상 이동 망으로 구성된 UDP 베어러 망을 통해서 서비스를 제공받을 수 있다. UDP 베어러 망을 사용하기 위해서 IWAP 플랫폼의 BNIC의 설정을 아래와 같이 UDP 서비스가 가능한 구성으로 설정한다.

- wdp-udp = wap (Bearer network UDP 설정)
- interface-name = 211.39.184.15
- wap-service = "wsp/wtp"

이상과 같이 IWAP 플랫폼의 컨텐츠를 생성, WSDE에 WML 관련 서비스 등록, UDP 베어러 망의 설정을 통해서 IWAP 플랫폼 기능실험을 위한 설정을 마치고, 시뮬레이션

WAP 클라이언트를 통해서 직접 기능시험 결과를 확인하였다.

(그림 12)는 WML 툴킷을 사용하여 index.wml 컨텐츠를 WSDE에 저장하여, UDP 베어러 망을 통하여 Nokia SDK가 구동되는 시뮬레이션 WAP 클라이언트 화면에 출력된 결과를 나타낸다. (그림 12)(a)는 시뮬레이션 WAP 클라이언트가 최초 IWAP 플랫폼에 접속한 초기화면이고, (그림 12)(b)는 카드(card)와 데크(deck)로 이루어진 선택 화면이며, (그림 12)(c)는 WBMP 파일을 WML 소스 파일에 포함하여서 출력한 결과를 나타낸다.



(a) 초기 연결 (b) 선 택 (c) WBMP 파일

(그림 12) IWAP 플랫폼 시뮬레이션 결과

또한 WSDE의 클라이언트 접속 로그 파일을 통해서 시뮬레이션 WAP 클라이언트로 컨텐츠가 정상적으로 전송된 실험 결과를 확인 할 수 있다.

```
211.39.184.15 unknown - [Mon Apr 24 14 : 04 : 41
   GMT + 09 : 00 2000 ]
GET /index.wml Content - Length = 0
211.39.184.15 unknown - [Mon Apr 24 14 : 04 : 44
   GMT + 09 : 00 2001 ]
GET /news.wml Content - Length = 0
211.39.184.15 unknown - [ Tue Apr 24 14 : 04 : 46 2001 ]
GET /rainy.wbmp Content - Length = 0
211.39.184.15 unknown - [ Tue Apr 24 14 : 04 : 47 2001 ]
GET /partcldy.wbmp Content - Length = 0
211.39.184.15 unknown - [ Tue Apr 24 14 : 04 : 48 2001 ]
GET /cloudy.wbmp Content - Length = 0
211.39.184.15 unknown - [ Tue Apr 24 14 : 04 : 48 2001 ]
GET /rainy.wbmp Content - Length = 0
211.39.184.15 unknown - [ Tue Apr 24 14 : 04 : 49 2001 ]
GET /sunny.wbmp Content - Length = 0
211.39.184.15 unknown - [ Tue Apr 24 14 : 05 : 01 2001 ]
GET /index.wml Content - Length = 0
```

현재 IWAP 플랫폼은 WAP 버전 2.0로 업그레이드 하고 있고, 망 관리를 위하여 SNMP(Simple Network Management Protocol) 에이전트 및 과금 처리 모듈을 첨가하는 작업을 수행 중에 있으며, 베어러 망으로 GPRS(General Packet Radio System)의 GGSN(Gateway GPRS Serving Node) 및 SGSN(Serving GPRS Service Node)과의 인터페이스를 위한 모듈 개발하고 있다.

5. 결 론

본 논문은 리눅스 기반에서 WAP 게이트웨이 및 서버가 통합적으로 지원되는 IWAP 플랫폼의 설계 및 구현에 것이다. 제안된 IWAP 플랫폼은 WAP 게이트웨이, WAP 서버 개발 환경을 제공하는 Java 기반의 웹 서버, WML 툴킷 및 MUI 등 크게 4개의 부분으로 구성되고, 베어러 망으로는 CSD 라우터 및 SMSC를 제공한다. 특히, 본 플랫폼은 Java 기반의 WAP 서버 개발 환경을 제공함으로서 WML 기반 WAP 서버 개발자나 컨텐츠 제공자에게 이동 망 내에 존재하는 실제 WAP 게이트웨이와의 연동 없이도 다양한 형태의 응용 서비스를 개발할 수 있는 환경을 제공할 수 있다.

IWAP 플랫폼은 버전 1.2를 기준으로 개발이 완료된 상태에 있고 현재 버전 2.0으로 업그레이드 진행 중에 있으며 망관리를 위한 SNMP 에이전트 모듈, CDR(Call Detailed Record)을 이용한 과금처리 모듈 및 GPRS 기반의 베어러 망 인터페이스 모듈 개발을 병행하고 있다.

WAP에서 규정하고 WALS(Wireless Application Layer Security), WTLS(Wireless Transport Layer Security), WPKI(WAP Public Key Infrastructure), WIM(WAP Identity Module) 등의 보안 메커니즘을 수용하기 위한 연구가 요구되어 지고, 향후 3세대 이동 통신망으로 하부 네트워크의 성능이 향상되고 그 위에 다양한 무선 멀티미디어 데이터 서비스를 지원하기 위해서는 WSP/WTP에 대한 기능 확장이나 성능 최적화에 대한 연구가 요구된다 하겠다.

참 고 문 헌

[1] WAP Forum, "Wireless Application Protocol Architecture Specification," Nov., 1999. URL : http://www.wapforum.org.

[2] WAP Forum, "Wireless Markup Language," Nov., 1999.

[3] WAP Forum, "Wireless Markup Language Script," Nov., 1999.

[4] WAP Forum, "Wireless Application Environment Specification," Nov., 1999.

[5] WAP Forum, "Wireless Session Protocol Specification," Nov., 1999.

[6] WAP Forum, "Wireless Transaction Protocol," Nov., 1999.

[7] WAP Forum, "Wireless Transport Protocol Security Protocol Specification," Nov., 1999.

[8] WAP Forum, "Wireless Datagram Protocol," Nov., 1999.

[9] WAP Forum, "Binary XML Content Format Specification," Nov., 1999.

[10] RFC 2068, "Hypertext Transfer Protocol-HTTP/1.1," Jan., 1997.

[12] A. Fasbender et al, "Any Network, Any Terminal, Anywhere," IEEE Personal Communications, April, 1999.

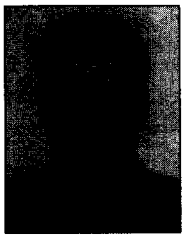
[14] D. Larner, "HTTP-NG Web Interface," IETF internet draft, August, 1988.

[15] Microsoft Mobile Explorer White Paper, http://www.microsoft.com/wireless.

[16] Danny Ayers et al, "Professional Java Server Programming," Wrox Press Ltd, 1999.

[17] Steve Mann et al, "Programming Application with the

Wireless Application Protocol," Willey & Sons Inc, 1999.
[18] Merlin Hughes et al, "Java Network Programming," MANNING, July, 1999.
[19] Nokia WAP Forum, http://www.forum.nokia.com.
[20] Kannel, http://www.kannel.org, January, 2001.
[21] Openware, http://www.phone.com.
[22] ETSI/GSM 03.40, Technical Realisation of Short Message Point to Point, 1999.
[23] Logica-Aldiscon, "Short Message Peer to Peer Protocol Specification v3.4 : Documentation Version : v1.0," 1999.
[24] http://java.sun.com/products/servlet/.
[25] http://www.novocode.com/doc/servlet-essentials/.
[26] Andrew Oram and Steve Talbott, "Managing Projects with make," O'Relly & Associates, Inc.
[27] Bil Lewis and Danielj Berg, "Multithreaded Programming with Pthreads," Sun Microsystems Press A Prentice Hall Title.
[28] Rob Gordon, "essential JNI JAVA NATIVE INTERFACE," Prentice Hall PTR.
[29] Ivor Horton, "Beginning Java 2," Wrox Press Ltd.
[30] http://xmlsoft.org/xml.html, The Gnome XML library.
[31] John K. Ousterhout, "TCL and The TK Toolkit," Addison-Wesley, 1999.



송 병 권

e-mail : bksong@skuniv.ac.kr
1984년 고려대학교 전자공학과(공학사)
1986년 고려대학교 대학원 전자공학과 (공학석사)
1995년 고려대학교 대학원 전자공학과 (공학박사)
1984년~1991년 삼성종합기술원 선임연구원
1995년~현재 서경대학교 정보통신공학과 조교수
관심분야 : 고속망 프로토콜, 분산처리시스템, 이동 컴퓨팅

오 태 안

e-mail : taean@tricomtek.com
1999년 명지대학교 전기전자공학부 정보통신공학과 졸업
1999년~현재 트라이콤텍 부설연구소 선임연구원
관심분야 : 무선어플리케이션, 메세징 프로토콜 등