2000년 한국정보처리학회 추계 학술발표논문집 제7권 제2호

WAP 게이트웨이에 대한 연구

강동우*, 박기현**
*계명대학교 컴퓨터공학과
**계명대학교 컴퓨터공학과
e-mail:khp@kmucc.kmu.ac.kr

A Study on a WAP Gateway

Dong-Woo Kang*, Kee-Hyun Park**
*Dept of Computer Engineering, Keimyung University
**Dept of Computer Engineering, Keimyung University

요약

최근 무선통신을 이용한 이동 컴퓨팅 환경에 대하여 점차 관심이 집중되고 있다. 그런데 이런 무선통신환경을 구축하기 위해서는 기존의 유선 HTML(Hyper Text Markup Language) 문서들을 WML(Wireless Markup Language) 문서등으로 변환하는 작업이 필요하다. 이러한 오버헤드를 줄이기 위한 노력의 일환으로 WAP(Wireless Application Protocol) 게이트웨이(Gateway)를 제작하여 사용할 수 있다. 본 논문에서는 기존의 HTML 서버의 내용을 이동 무선단말기에서도 검색할 수 있도록 하는 WAP 게이트웨이에 대하여 연구한다. WAP 서버, WEB 클라이언트, 프로토콜 변환기, WML 인코더/디코더 등으로 WAP 게이트웨이를 구성하며 현재 WTP, WSP 단계에서의 설계와 구현을 하고 있으며 본 논문에서는 현재까지의 중간결과를 설명한다.

1. 서론

기존 유선 인터넷 환경은 데스크톱 이상의 컴퓨터와 높은 대역을 제공할 수 있는 네트워크를 기반으로 하지만, 이동(Mobile) 인터넷 환경과 이동 무선 단말기는 현재의 데스트톱 기준으로 접근하기에는 아직 전력 소모량, 메모리 크기, 디스플레이 크기, 전송 속도, 안정성 등에서 많은 어려움이 있다[1]. 따라서 유선 인터넷의 표준을 그대로 사용하는데 한계가 있으므로, 기존 표준을 가능하면 따르면서도 무선 환경에 적합한 프로토콜을 만들려는 움직임이 일어났다.

이에 따라 여러 통신업체들이 모여 서로의 기술을 공개하고 협의하는 조직인 WAP(Wireless Application Protocol) 포럼을 만들어 WAP을 발표하였으며, 현재 WAP 스펙 버전 1.2이 발표되어 있다[1]. WAP의 목적은 디지털 셀룰러 전화(PCS 등)와 무선 터미널에서의 인터넷 서비스 및 다른 종류의 무선 통신망 기술에서 운용될 수 있는 무선 프로토콜 규격과 컨텐츠, 응용기술을 개발하는데 있다.

WAP 게이트웨이(Gateway)는 기존 HTTP/TCP/IP 기반의 패킷전송에 대해 WSP/WTP/UDP 기반의 패킷으로 상호 변환을 시켜주며 무선환경에 맞게 전송 패킷의 오버헤드를 줄여주기 위해 최적화를 시켜 준다. 또한 기존 웹서버의 HTML 문서를 WML 문서로의 자동변환을 해줌으로써 무선인터넷을 위한 새로운 컨텐츠의 작성 없이도 기존 HTML을 이용하여 서비스가 가능하도록 한다. 이것은 무선인터넷을 위한 컨텐츠 작성에 대한 시간과 인력을 최소화시켜 준다.

WAP 게이트웨이의 기본동작 원리는 WAP 프로토콜에 따라 요청 받은 서비스를 기존 인터넷 유선망을 통해 서버에 요청한다. 응답으로 인터넷 서버로부터 응답 메시지를 받고 서비스를 요청했던 이동 무선단말기로 WAP 프로토콜로 전송함으로써 과정이 이루어진다[1]. 구현된 게이트웨이가 올바른 동작을 하고 있는지 테스트하기 위해서 Nokia에서 제공하는 인터넷폰 에뮬레이터를 이용, 게이트웨이를 통

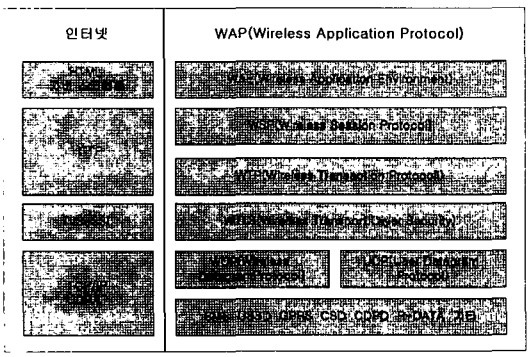
** 이 연구는 2000년도 중소기업기술혁신 개발사업비로 조성되었습니다.

과하게 하여 기존 웹서버에 요청을 하도록 하고, 응답 메시지를 인터넷폰 에뮬레이터에 다시 전송하여 올바른 동작이 이루어 졌는지를 테스트한다[7,9].

본 논문의 구성은 다음과 같다. 제 2장에서는 WAP의 구조에 대하여 알아보고, 제 3장에서는 WAP 게이트웨이의 구성과 각 구성요소들의 동작에 대하여 설명한다. 제 4장에서는 WSP 프로토콜의 동작과 WSP와 HTTP의 프로토콜 변환에 대한 구현 방법에 대한 설명을 하며, 마지막으로 제 5장에서는 본 논문에 대한 결론 및 향후 연구 계획에 대하여 기술한다.

2. WAP 구조

WAP의 구조는 <그림 1>과 같이 전체 프로토콜이 계층화된 구조를 가지고 있기 때문에 이동 무선 단말기의 애플리케이션 개발을 위해 확장이 가능한 환경을 제공하고 있다. 각 계층은 그보다 상위 계층에 의해 접근할 수 있고 다른 외부 서비스나 애플리케이션에 의해서도 직접 접근할 수 있다.[1,8,9]



<그림 1> WAP의 구조

1) WAE(Wireless Application Environment) : 웹과 이동 통신 기술의 통합에 기초한 일반적인 목적의 애플리케이션 환경이다. WAE의 주된 목적은 서비스 제공자와 개발자가 상호 대화할 수 있는 환경을 제공함으로써, 다양한 무선 플랫폼 위에서 작동할 수 있는 애플리케이션과 서비스를 능률적으로 구축할 수 있게 하는데 있다.

2) WSP(Wireless Session Layer) : 두 개의 세션 서비스를 위한 인터페이스로 구성되며, 트랜잭션 계층 프로토콜인 WTP(Wireless Transport Protocol) 위에 동작하는 연결형 서비스와 보안 또는 비보안 데이터그램 서비스인 WDP 위에서 동작되는 비연결형이 있다. HTTP의 세션을 맺는 기능과 동일하며, 세션 헤더의 교환과 케이퍼빌러티 협상 등 최적화된 컨텐츠 전달을 위한 상태를 제공한다.

3) WTP(Wireless Transaction Layer) : 데이터그램 서비스 위에서 실행되며, 인터넷 접속 전용 컴퓨터에서 비교적 간단한 트랜잭션을 위한 프로토콜이며, 3개의 클래스를 지원한다. 대역폭에서 벗어난 데이터가 들어올 때, 선택적으로 승인(acknowledge)하는 기능, 전송할 메시지의 수를 감소시키기 위한 PDU(Protocol Data Unit)로의 연결과 지연의 통지 및 비동기 트랜잭션을 수행한다.

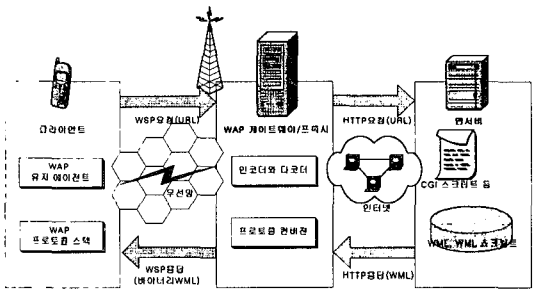
4) WTLS(Wireless Transport Layer Security) : 공식적으로 SSL(Secured Socket Layer)로 알려진 산업표준형 TLS(Transport Layer Security)에 기반을 둔 보안 프로토콜이다. WAP의 트랜스포트 프로토콜과 함께 사용하도록 설계되었고 좁은 대역의 통신채널에서 사용될 수 있도록 최적화되어 있다.

5) WDP(Wireless Datagram Protocol) : WDP 위에 존재하는 프로토콜 계층들이 다양하게 존재하는 물리적 네트워크에 상관없이 작동할 수 있도록 한다. WDP 프로토콜에서 상위계층 프로토콜에게 일관된 인터페이스를 제공하기 때문에 상위계층 프로토콜은 무선망의 구성에 관계없이 자신의 기능을 독립적으로 수행할 수 있다. 현재 가능한 WAP 프로토콜은 다양한 응용 서비스 위에서 작동할 수 있도록 설계되었다. 이를 위해 WDP에서 각 응용 서비스마다 어떤 동작을 할 것인지를 정해 놓았다. 앞으로 지원 가능한 베어러(Bearer) 수는 증가하게 될 것이며 현재는 GSM, CDMA, PHS, IS_136 등을 지원하고 있다[1,2,6].

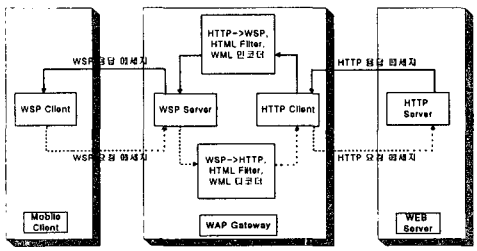
3. WAP 게이트웨이의 구성

WAP 게이트웨이를 이용한 전반적인 무선 인터넷의 구성은 <그림 2>와 같으며 기본적인 동작은 기존 유선 인터넷의 동작원리와 흡사하다. 클라이언트(이동 무선단말기)와 WAP 게이트웨이간은 무선망을 이용한 WAP 프로토콜로 통신이 이루어지며 웹서버와 WAP 게이트웨이간은 TCP/IP를 이용한 패킷전송이 이루어지게 된다[2,4].

<그림 3>은 WAP 게이트웨이의 구성요소로써 클라이언트(이동 무선단말기), WAP 게이트웨이, 웹서버의 구성요소들을 세부 모듈로 나누어 볼 수 있다.



<그림 2> WAP 게이트웨이를 이용한 무선인터넷



<그림 3> WAP 구성요소들의 각 모듈들

이동 무선단말기와 게이트웨이간의 데이터통신에서 이동 무선단말기의 브라우저(WSP 클라이언트)는 WSP와 같은 트랜스포트 프로토콜을 이용해 통신을 하게 된다. UDP 프로토콜은 패킷 교환 네트웍 IP 위에 위치하며 WSP와 같은 프로토콜은 UDP 위에 제공한다. 따라서 WAP 게이트웨이 시스템에서 WSP는 HTTP로 변환이 이루어지며, UDP는 TCP로 변환되어 인터넷을 통해 웹서버(HTTP 서버)로 전달된다. 웹서버에서의 응답은 WAP 게이트웨이에서 위의 과정의 역순으로 이동 무선단말기로 전달이 이루어진다.

WAP 게이트웨이는 프로토콜 변환(HTTP와 WSP간의 상호 변환)과 WSP 서버, HTTP 클라이언트, HTML 필터(HTML Filter - HTML 문서에서 WML 문서로의 변환)의 모듈로 구성된다.

1) WSP 서버 : WAP 브라우저로부터의 요청을 받아 메시지의 헤더 분석과 케이퍼빌러티(Capability) 협상 등을 수행한다. 메시지 분석을 마친 WSP 서버는 HTTP 서버에서 자원을 가져와야 할 경우 프로토콜 변환기로 메시지를 전송하여 프로토콜 변환을 수행한 후 HTTP 클라이언트에 메시지를 보낸다.

2) HTTP 클라이언트 : HTTP 프로토콜을 이용하여 HTTP 서버로 요청과 응답을 수행한다.[3] 또한 HTTP 클라이언트는 자체 에러검색 기능을 주어 불필요한 메시지 교환이 이루어지지 않도록 한다.

3) HTML 필터 : HTML 문서를 WML 문서로 변환시킨다. 이동 무선단말기의 액정 크기와 메모리의 한계 등으로 기존 HTML 문서를 원본 그대로 변환하기에는 어려움이 따르며 자체적인 기준에 의해 변환을 하게 된다.

4) WML 인코더/디코더 : 무선데이터 전송의 효율을 높이기 위한 것으로 잘 알려진 태그들과 문자들은 WAP 사양[1]에서 정의되어진 테이블을 참조하여 인코딩을 수행한다. WSP 클라이언트 측에서는 이 인코딩된 WML을 다시 디코딩하여 이동 무선단말기 브라우저에 출력을 하게 된다.

4. WSP 프로토콜의 동작과 프로토콜 변환

WSP는 상위 또는 하위 단계에서 발생하는 이벤트와 이 이벤트에 의해 발생되어 전송되는 PDU를 통하여 세션의 성립, 메소드의 처리, 다음에 행할 동작 등이 정의되어진다. WSP 이벤트의 종류에는 세션의 연결, 종료, 중지, 재개, 메소드발생, 메소드결과 등이 있으며 이벤트의 정의의 예는 <표 1>과 같다..

```
struct S-Connect.ind
{
    AddressType   Address,
    HeaderType   Client Headers,
    CapabilityType   Requsted Capabilities,
    integer   session ID
}

struct S-Disconnect.ind
{
    integer   Reason code,
    integer   Redirect Security,
    AddressType    Redirect Addresses,
    HeaderType   Error Headers
    String   Error Body,
    integer   session ID
}
```

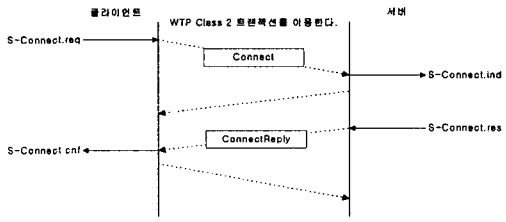
<표 1> WSP 이벤트 정의예

이벤트에 의해 전송되어지는 PDU(Packet Data Unit)의 정의의 예는 <표 2>이며 Connect PDU를 보여주고 있다.

이름	타입	데이타값
Type	uint8	PDU 타입정의값
Version	uint8	프로토콜 버전
CapabilityLen	uintvar	Capability 필드의 길이
HeadersLen	uintvar	Header 필드의 길이
Capabilities	octets	S-Connect.req 이벤트의 요청 Capabiltity의 값
Headers	octets	S-Connect.req 의 Client Headers의 값

<표 2> WSP PDU의 정의 예

이벤트와 PDU에 의한 일반적인 WSP의 동작을 <그림 4>에서 보여주고 있다. 클라이언트에서 발생한 연결 이벤트에 의하여 connect PDU가 생성되어 서버로 전송이 이루어지며, 서버에서는 connect PDU가 전송되었다는 이벤트를 받게 된다. 서버는 다시 클라이언트로 connectReply PDU를 전송하게 된다.



<그림 4> WSP의 일반적인 세션 생성

<그림 4>와 같은 일반적인 세션 생성의 경우에 대한 알고리즘은 <표 3>과 같다.

1. WSP는 NULL state 에서 event 발생을 기다림.
2. TR-Invoke.ind event 발생.
3. 발생된 event(TR-Invoke.ind)를 분석함.
(condition과 각 요소들을 분석)
4. TR-Invoke.res 를 생성하여 하위 layer로 보냄.
5. N_Methods 를 0으로 초기화시킴.
6. S-Connect.ind를 생성하여 상위 layer로 보냄.
(Method state의 경우 프로토콜 변환기로 데이터를 넘겨주고 다시 프로토콜 변환기에서 데이터를 받아야 함).
7. WSP는 State를 connecting 상태로 변환시킴.
8. WSP는 connecting state 에서 event 발생을 기다림.
9. S-Connect.res event 발생.
10. 이 주소로 생성되어 있는 다른 모든 session의 연결을 종료시킴.
11. Session_ID를 부여함.
12. ConnectReply PDU(S-Connect.res를 분석하여) 를 생성하고 TR-Result.req를 만들어 보냄.
13. 이 session과 관련되어 HOLDING 상태에 있는 모든 method transaction들을 Release 시킴.
14. WSP는 State를 connecting_2 상태로 변환시킴.
15. WSP는 connecting_2 state 에서 event 발생을 기다림.
16. TR-Result.cnf event 발생.
17. WSP는 State를 connected 상태로 변환되면서 연결이 이루어짐. 다음 이벤트를 기다림.

<표 3> WSP 세션 생성 알고리즘

위의 과정을 통해 생성된 PDU는 프로토콜 변환기를 통하여 HTTP 프로토콜로의 변환과정을 거쳐 웹서버로 전송이 되어야 하며, 반대로 웹서버에서 온 응답 메시지도 프로토콜 변환기를 거쳐 WSP 서버로 전송이 이루어진다. WSP 프로토콜과 HTTP 프로토콜은 서로 유사한 형식으로 되어 있으므로 프로토콜 변환은 맵핑 테이블을 이용한 변환방법과 파싱을 이용한 헤더 정보의 첨가, 수정, 삭제 방법을 이용하여 프로토콜을 재구성한다.

4. 결론 및 향후계획

WAP 게이트웨이를 이용한다면 유선 HTML 문서들을 WML 문서 등으로 변환하는 작업의 오버헤드를 줄일 수 있고 무선 인터넷과 유선 인터넷과의 효율적인 데이터 통신이 이루어질 수 있는 등, WAP 게이트웨이를 이용한 여러 가지 응용분야가 생길 수 있다. 본 논문은 이러한 WAP 게이트웨이의 구축과정에서 수행된 중간결과를 설명하였다.

향후 연구과제로서 WTP, WSP 단계에서의 WAP 게이트웨이의 구현을 완료하고자 한다. 또한 WML을 지원하는 웹서버, WAP 게이트웨이, Nokia 에뮬레이터를 이용한 테스트 환경을 구축한 후, 가상 시나리오를 설정하여 WAP 게이트웨이의 동작을 분석하고자 한다.

참고문헌

[1] WAP Forum "Wireless Application Protocol spec 1.2", 1999. 12.

[2] Charles E.Perkins "Mobile IP Design Principles and Practices" Addison-wesley, 1998.

[3] RFC 2068 : Hypertext Transfer Protocol HTTP/1.1, 1997, 1.

[4] RFC 2002 : IP Mobility surpport, 1996. 10.

[5] W. Richard Stevens "Unix Network Programing" Prentice Hall, 1991.

[6] Douglas E. Comer " Computer Networks And Internets" Prentice Hall, 1997.

[7] Nokia Group, "Nokia WAP Toolkit 2.0", 2000.

[8] Charles Arehart "Professional WAP" WROX Press, 2000.7.

[9] "Programming Applications with WAP", Wiley, 2000. 3.