



NEAR EAST UNIVERSITY

Faculty of Engineering

**Department of Electrical and Electronic
Engineering**

THEORY AND APPLICATIONS OF WAP

**Graduation Project
EE – 400**

Student: Muhammad Atif Malik (971059)

Student: M. Haroon Khan (20001575)

**Supervisor: Assoc. Prof.
Dogan Ibrahim**

Nicosia - 2001

ACKNOWLEDGMENTS

First I want to thank Assoc. Prof. Dogan Ibrahim to be my supervisor. Under his guidance, I successfully overcome many difficulties and learn a lot about WAP. In each discussion, he explained my questions patiently, and I felt my quick progress from his advises. He always helps me a lot either in my study or my life. I asked him many questions in Computer Networks and Communication and he always answered my questions quickly and in detail.

Thanks to Faculty of Engineering for having such a good computational environment.

I also want to thank my friends in 'NEU': M. Arshad Khan, M. Jawad Khan, my housemates and close friends. Being with them made my 4 years in 'NEU' full of fun.

Finally, I want to thank my family, especially my parents. Without their endless support and love for me. I would never achieve my current position.

CONTENTS

ACKNOWLEDGMENT	i
ABSTRACT	iv
INTRODUCTION	1
1. WHY WE HAVE TO USE WAP	2
1.1 Why wap	2
1.2 Wap model	3
1.3 Wap architecture	4
1.3.1 Wireless application environment	4
1.3.2 Wireless session protocol	4
1.3.3 Wireless transaction protocol	5
1.3.4 Wireless transport layer security	5
1.3.5 wireless datagram protocol	6
2. HOW TO CREATE OUR WAP SITE	7
2.1 Example to change the title of the card	10
2.2 WAP phones	12
2.2.1 Connecting to wap sites with our phone	12
2.3 Scripting languages like ASP,JSP,PHP&Perl	13
2.3.1 ASP,JSP,PHP& perl	14
2.3.2 Page cached by the phone	14
2.3.3 Possible to do with wap	15
3. WAP-AN EXTENSION OF THE INTERNET MODEL	16
3.1 Wap-an extension of the internet model	16
3.2 What is the syntax of WML	18
3.3 Blue tooth technology	20
4. WAP FIT IN THE WIRELESS COMPUTING APPLICATION	23
4.1 WAP fit in the wireless computing application	23
4.2 WAP micro-browser	23

4.2.1 Warnings about GPRS and WAP hype – study	23
4.3 Nokia and WAP	25
4.4 WAP applications and examples	27
4.5 Interesting applications of WAP	28
4.6 The problem areas	30
5. WAP COMMUNICATION PROTOCOL AND ITS COMPONENTS	31
5.1 WAE & WML	31
5.2 Some editors we can use for creating WAP pages	34
5.3 WML script	34
5.4 WBMP	35
5.5 WTLS	36
5.6 WTP	37
5.7 WAP address limitations of wireless internet	39
6. WHAT ARE THE WAP PRODUCTS	
MICROBROWSER, WAP GATEWAY, WAP SERVERS	41
6.1 WAP gateway	41
6.2 WAP server	43
6.3 Gateways	44
6.4 Sending a wireless text – message with Java	44
6.5 Challenges of sending an SMS message from Java	45
6.6 Using SMTP to send a text – message	46
6.7 Using a third – party wireless platform to send a text – message	52
6.8 The players in this WAP game	54
6.9 I – mode	54
Conclusion	56
References	57

Abstract

Wireless application protocol (WAP) is an application environment and set of communication protocols for wireless devices designed to enable manufacturer-, vendor-, and technology-independent access to the Internet and advanced telephony services.

WAP bridges the gap between the mobile world and the Internet as well as corporate intranets and offers the ability to deliver an unlimited range of mobile value-added services to subscribers—independent of their network, bearer, and terminal. Mobile subscribers can access the same wealth of information from a pocket-sized device as they can from the desktop.

WAP is a global standard and is not controlled by any single company. Ericsson, Nokia, Motorola, and Unwired Planet founded the WAP Forum in the summer of 1997 with the initial purpose of defining an industry-wide specification for developing applications over wireless communications networks. The WAP specifications define a set of protocols in application, session, transaction, security, and transport layers, which enable operators, manufacturers, and applications providers to meet the challenges in advanced wireless service differentiation and fast/flexible service creation. There are now over one hundred members representing terminal and infrastructure manufacturers, operators, carriers, service providers, software houses, content providers, and companies developing services and applications for mobile devices. For more information, visit the WAP Forum at <http://www.wapforum.org/>.

WAP also defines a wireless application environment (WAE) aimed at enabling operators, manufacturers, and content developers to develop advanced differentiating services and applications including a microbrowser, scripting facilities, e-mail, World Wide Web (WWW)-to-mobile-handset messaging, and mobile-to-telefax access.

The WAP specifications continue to be developed by contributing members, who, through interoperability testing, have brought WAP into the limelight of the mobile data marketplace with fully functional WAP-enabled devices.

Introduction

WAP bridges the gap between the mobile world and the Internet as well as corporate intranets and offers the ability to deliver an unlimited range of mobile value-added services to subscribers—independent of their network, bearer, and terminal. Mobile subscribers can access the same wealth of information from a pocket-sized device as they can from the desktop.

WAP is a global standard and is not controlled by any single company. Ericsson, Nokia, Motorola, and Unwired Planet founded the WAP Forum in the summer of 1997 with the initial purpose of defining an industry-wide specification for developing applications over wireless communications networks. The WAP specifications define a set of protocols in application, session, transaction, security, and transport layers, which enable operators, manufacturers, and applications providers to meet the challenges in advanced wireless service differentiation and fast/flexible service creation. There are now over one hundred members representing terminal and infrastructure manufacturers, operators, carriers, service providers, software houses, content providers, and companies developing services and applications for mobile devices.

WAP also defines a wireless application environment (WAE) aimed at enabling operators, manufacturers, and content developers to develop advanced differentiating services and applications including a microbrowser, scripting facilities, e-mail, World Wide Web (WWW)—to-mobile-handset messaging, and mobile-to-telefax access.

The WAP specifications continue to be developed by contributing members, who, through interoperability testing, have brought WAP into the limelight of the mobile data marketplace with fully functional WAP-enabled devices.

CHAPTER ONE

1. WHY WE HAVE TO USE WAP

1.1 Why WAP

WAP provides mobile to internet connectivity. In theory WAP enabled devices like Mobile Phones, PDA's and other hand-held devices should be able to access web content as desktops do. But since minimal web content is html content and small devices have limitation for rendering this type of content, a different language called Wireless Markup Language (WML) has been adopted for use. WML is designed keeping in mind small display area of hand-held devices like mobile phones. Besides this problem, mobile devices have some other limitations too, which make WAP important. Here is a list of a few.

- **Less powerful CPUs**
- **Less memory (ROM and RAM)**
- **Restricted power consumption**
- **Smaller displays**
- **Different input devices (eg, a phone keypad)**

It is not only the limitations of mobile devices but problems in wireless data networks too had the impact on designing Wireless Application Protocol. Issues with wireless data networks are

- **Less bandwidth** : The GSM network user has maximum bandwidth of 9.6 kbps available to him/her. GPRS (General Packet Radio Service) will increase the bandwidth allocated per user to 160 kbps.
- **More latency**
- **Less connection stability**
- **Less predictable availability**

1.2 WAP Model

This diagram shows how the WAP model works. WAP Client sends an encoded request to the WAP server. WAP client can be a mobile phone, Personal digital assistant or any other WAP enabled device. The request is encoded in a compact form because of the limitations of the wireless data networks. WAP gateway decodes this request to HTTP request and sends this to the web server. The web server responds with the wml content. WAP gateway prepares encoded response and sends it to client server. the figure 1.1 shows how wap works.

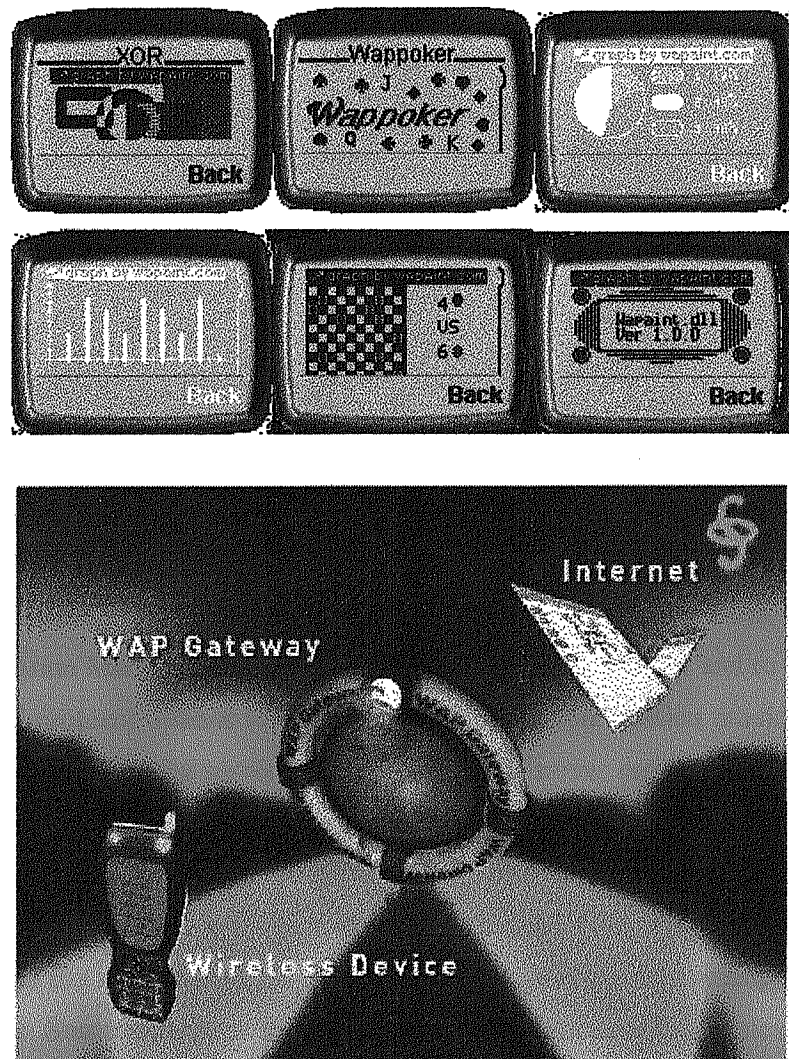


Figure 1.1

1.3 WAP Architecture

WAP has a layered architecture. The entire protocol stack consists of five layers.

- Wireless Application Environment
- Wireless Session Protocol
- Wireless Transaction Protocol
- Wireless Transport Layer Security
- Wireless Datagram Protocol

1.3.1 Wireless Application Environment

Wireless Application Environment (WAE) provides the environment for the development of applications and services which operate over wireless communication networks.

1.3.2 Wireless Session Protocol

WSP provides application layer with two types of services. First is Connection-Oriented service and the other is Connection-Less service. Connection-Oriented service operates above WTP layer. This mode of service provides facilities like Session Management, Method Invocation, Confirmed Push, Non-Confirmed Push and Suspend Resume. Session Management allows a client to connect to a server and to agree on the facilities and protocol options to be used during the session. Session provides a context between the client and the server. Both the client and server can terminate this session. Method Invocation allows a client to ask the server to execute an operation and return the result. Push facility permits the server to send unsolicited information to the client. This can be confirmed or non-confirmed. In confirmed push the client acknowledges the receipt of the information, but in non-confirmed push no acknowledgement is sent to the server. Suspend Resume facility allows the suspension of the session, so that the state of the session is preserved. In this state both the peers know that the further communication is not possible until the session is resumed. Session layer provides the functionality of encoding and decoding of data. Using these encoding techniques, the size of the data over the wireless network is reduced.

1.3.3 Wireless Transaction Protocol

WTP runs on top of a data gram service and optionally a security service. WTP has been defined as a lightweight transaction oriented protocol that is suitable for implementation in "thin" clients (mobile stations) and operates efficiently over wireless data gram networks. The benefits of using WTP include:

1. Improved reliability over data gram services. WTP relieves the upper layer from re-transmissions and acknowledgements, which are necessary if data gram services are used.
2. Improved efficiency over connection oriented services. WTP has no explicit connection set up or teardown phases.
3. WTP is message oriented and designed for services oriented towards transactions, such as "browsing".

WTP provides three classes of transactions.

⊕ **Class 0** Unreliable one way requests. Request from one peer to another is not followed by an acknowledgement of the receipt of request.

⊕ **Class 1** Reliable one way requests. Request from one peer to another is followed by an acknowledgement of the receipt of request.

⊕ **Class 2** Reliable two way request-reply transactions. Request from one peer is acknowledged by the other, which is followed by a reply and that is also acknowledged.

1.3.4 Wireless Transport Layer Security

The Security layer protocol in the WAP architecture is called the Wireless Transport Layer Security, WTLS. The WTLS layer operates above the transport protocol layer. The WTLS layer is modular and it depends on the required security level of the given application whether it is used or not. WTLS provides the upper-level layer of WAP with a secure transport service interface that preserves the transport service interface below it. In addition, WTLS provides an interface for managing (eg, creating and terminating) secure connections. The primary goal of the WTLS layer is to

provide privacy, data integrity and authentication between two communicating applications. The WTLS protocol is optimised for low-bandwidth bearer networks with relatively long latency.

1.3.5 Wireless Datagram Protocol

The WDP layer operates above the data capable bearer services supported by the various network types. As a general datagram service, WDP offers a consistent service to the upper layer protocol (Security, Transaction and Session) of WAP and communicate transparently over one of the available bearer services. Since the WDP protocols provide a common interface to the upper layer protocols the Security, Session and Application layers are able to function independently of the underlying wireless network. This is accomplished by adapting the transport layer to specific features of the underlying bearer. By keeping the transport layer interface and the basic features consistent, global interoperability can be achieved using mediating gateways.

CHAPTER TWO

2. HOW TO CREATE OUR WAP SITE

In a broad, general view, there's not much difference between "the web" and WAP. Actually, WAP is just as much part of "the web" as our typical PC based browser such as Netscape and Internet Explorer. For this introduction we will call normal web servers and web browsers "the web".

If we think of "the web" as a concept, with its specifications and protocols explaining how things must operate in order to do what we want them to do, we can think of WAP as a concept too. WAP is a set of specifications and protocols that explain how things should and must operate in order to be WAP. The difference between "the web" and WAP is of course that WAP is meant for wireless devices, including but not limited to mobile phones. In the wireless environment everything is simpler and smaller. We have very narrow bandwidths, very little power from batteries to drive very small processors. Everything must be smaller and simpler. If we've followed the hype on WAP we will probably have heard that WAP will die when faster connections become available, such as GPRS, but this just isn't the case. These higher bandwidth technologies will only improve the services that will become available through WAP.

Unfortunately the WAP concept is plagued by much hype. Many of we will no doubt think of WAP as the same as a "mobile phone from which we can surf the web", and this in a way true, but not as we think. WAP is very much in it's infancy, and expecting things that will blow we socks off today is probably going to end in disappointment. WAP is more or less at the evolutionary stage that "the web" was at about five years ago. In short, the limits of WAP today is plain text with some simple styling like **bold** and *italics*, but only on some devices. Monochrome images on displays that are typical 100 by 40 pixels. No audio, and no fancy animations. But this is today, and many companies are working hard on exciting new technologies to make WAP as saucy as anything we can imagine.

In order to get a grasp on WAP, there are a few things we need to know about how it works. On "the web", a "web page" is a file (document) stored on a web server,

and this file is sent to the web browser when we request it. In WAP, the documents are called decks, and they are also stored on web servers. Each deck contains one or more cards, and the content of each card is what is normally displayed in the WAP browser window, although the display window is scrollable. On "the web" the markup language for documents is HTML. In WAP the markup language is WML. The two might look very similar, and for this we'll pretend that they are very similar (but there are some rather important differences).

On "the web" our web browser will usually talk to a web server directly to ask for a certain page. In WAP, there is another device involved which is called a WAP gateway. This device sits somewhere on the network between our WAP device and the web server and helps the web server send data to the WAP device and vice versa. Also, although a normal web server is commonly used to serve WML decks, the web server will in most cases need to be slightly reconfigured in order to serve the WML decks correctly. This involves setting the so called MIME types on the server, and for more information on this, we can have a look at [this document](#). The person who owns the web server we are using should be able to help us with this.

A very simple HTML page may look like this:

```
<H1>Hello world!
```

The HTML code above may not be correctly formatted. First of all the end tag `</H1>` is missing, and there should really be a `<HEAD>` and `<BODY>` section etc etc. However, all normal web browsers will be able to display the page above correctly. In WML it's a different story. The WML code is much more strict, meaning that we must follow the formatting rules, or we simply won't be able to display it. There's no room for error. The reason for this is that the so called WAP gateway, with very little intelligence, will convert the WML code to a "compressed" format before it is sent to the WAP browser to save time.

The same as above written in WML would look something like this:

```
<?xml version="1.0"?>
```

```
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1/EN"
```

```
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="mycard" title="My first card">
<p>Hello world!</p>
</card>
</wml>
```

Most important is the header. The absolutely first character that is received by the WAP gateway, must be the < character that starts the line <?xml version="1.0">. Any other character (even a space or a carriage return or other will break the card).

If we want, we can now copy the code above and paste it into our favourite text editor, for instance Notepad. Save the file on our computer and call it *world.wml*. Notice the *.wml* file extension. This is the common file extension for WML files. By now, we'll want to look at this file in a WAP browser. Go to this page and pick a software based browser that can run on our computer. Stay away from the SDKs and the Toolkits for now. These are development tools, and really overkill for our very first WML deck. WinWAP is a good starter application, but as we get more advanced, the UP.SDK is a recommended choice. WinWAP can also open and display WML decks that are stored on our local computer, which means that we can test them without having to place them on a webserver.

Since the WML code above is extremely simple and should work on the first try, run WinWAP and open the file. If we see Hello world! in the WinWAP display, we've just made our very first WAP application. The WAP components in the chain between we and the web server, the WAP gateway and the WML browser are as previously mentioned very strict when it comes to syntax errors. If we get any problems, strip the code down to the bare minimum. This is especially true for the first two lines of code which are the so called XML and DTD definitions, and if we type the code in manually there is lots of room for error. If we're using a "software" WML browser like WinWAP or WapMan, which do not involve the WAP gateway, to test the code, we might find that it works on these even with some errors, but on a real WAP device it will not. The WAP gateway is much more strict than the emulators. Play around with the code above. Change the title of the card by setting the `title` parameter

in the `card` tag to something else. Change the text in the card and add another paragraph of text. For instance to something like this:

2.1 Example to change the title of the card

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
    "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="mycard" title="My second card">
<p>Hello again world!</p>
<p>I'm back again with another WML deck created from scratch!</p>
</card>
</wml>
```

Notice the `<p>` and `</p>` at the beginning and end of each line of text. The `<p>` tags are typically used to begin a new paragraph of text, and as previously mentioned, all WML tags must be closed, in this case with `</p>`. Actually, that's not quite true. Some WML tags do not have a closing tag. Examples are `
` (break) which is used to break a line of text, and `` which is used to insert an image in the text.

Let's try to use both these tags right away. But wait! As we already know if we've paid attention so far, WAP devices currently only support monochrome images. Further, they only support images in a special wireless image format called WBMP. Not GIFs, not JPEGs. To test the `` tag we'll obviously need a WBMP image. For now we'll use an already made WBMP image. The image is what we see below. In GIF format on the left, and in WBMP format on the right.

Finally! Let's use the image in a WML deck.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
    "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="mycard" title="My girlfriend">
<p></p>
<p>Check out my girlfriend!<br/>(I wish :-)</p>
</card>
</wml>
```

First of all notice that there is no closing tag for the `` tag, but as we can see, the tag ends with a / (slash) unlike the tags that need a closing tag. In short the `` tag does not need a closing tag. Then see the `alt` parameter to the ``. This parameter is required because not all WAP devices can display images. The text inside the `alt` parameter will be shown instead.

The next reasonable step would be to play around with the WML decks we have made so far. Then download the [WML Reference Specifications](#) which lists all the WML tags and what they do. (Adobe Acrobat Reader is required to read this and many other specifications related to WAP. Better [download](#) it right away). Most WAP specification documents are unfortunately pretty boring reading for beginners, and lots of caffeine is required to read them from beginning to end. Unlike this article, they need to be exact and contain no errors or omissions. As we get more advanced, the [WAP Specifications](#) from [WAP Forum](#) are required reading. WAP Forum are after all the group who develop WAP. For a slightly more human approach to getting into WAP.

Don't be afraid to ask others for help. Join mailing lists that cover WAP. We recommend the [WML and WMLscript Programmers](#) as it has a high number of people who are both good programmers and willing to help.

2.2 WAP phones

1. **Nokia 6210.** GSM 900/1800 supports EGSM 900.
2. **Siemens C35i.** GSM 900/1800.
3. **Siemens S35i.** GSM 900/1800.
4. **Siemens M35i.** GSM 900/1800.
5. **Motorola T2288.** GSM 900/1800.
6. **Motorola v.2288.** GSM 900/1800.
7. **Motorola Timeport P7389.** GSM 900/1800/1900.
8. **Ericsson R320s.** GSM 900/1800.
9. **Ericsson R380.** GSM 900/1800 supports EGSM 900.
10. **Neopoint 1000.** PCS CDMA.
11. **Panasonic GD93.** GSM 900/1800.
12. **Mitsubishi Trium Geo WAP.** GSM 900/1800.
13. **Samsung SGH-A110.** GSM 900/1800.
14. **Samsung SGH-810.** GSM 900.
15. **Sanyo SCP-4000.** CDMA.
16. **Sony CMD-Z5.** GSM. Supports both WML and HTML.
17. **Nokia 9110i Communicator.** GSM 900. Also HTML browser

2.2.1 Connecting to WAP sites with our phone

When we buy a WAP phone, it will probably be configured by an operator for immediate use. Unless, we have to enter some information in the phone before we are able to connect to WAP sites. The setup of a WAP phone will vary, depending on which model we have, but generally there is a menu called 'Internet' or 'Services'. In this menu, we should have the possibility to choose 'Setup', 'Access' or something similar. The information we need to know will be available from our operator, visit their web page or ask the staff in the shop where we bought the phone.

Necessary information is:

- Dial-up number
- Call type (either analogue or ISDN)
- IP address (4 X 3 numbers separated by dots, e.g. 010.010.020.22)
- Log-in name (sometimes required)

- Password (sometimes required)
- Bearer (data or SMS)
- Idle time before disconnecting (this is very valuable if we forget to disconnect, then the phone will break the connection after xx seconds, saving we money).

We will also be asked to enter the address to our 'Start-up page', here we can just write `http://wap.com`

or any other WAP address we know.

Then we are ready to surf the wireless Internet. Many phones let we enter an address, with others we must choose the 'Go to location' service provided by some WAP sites, like `http://wap.com`

2.3 Scripting languages like ASP, JSP, PHP and Perl

If our server side scripting language has been installed and is working fine with HTML pages, it should work just as well with WML. Just make sure to output the correct content-type as part of the header. If we are using PHP, this is how we should do it:

```
<?php
header("Content-Type: text/vnd.wap.wml");
?>
```

Here is how we do it with ASP:

```
<%
Response.ContentType = "text/vnd.wap.wml"
%>
```

and very similar with JSP:

```
<%@ page contentType="text/vnd.wap.wml" %>
```

2.3.1 Scripting languages like ASP, JSP, PHP and Perl

Yes, absolutely. If our server side scripting language has been installed and is working fine with HTML pages, it should work just as well with WML. Just make sure to output the correct content-type as part of the header.

If we are using PHP, this is how we should do it:

```
<?php
header("Content-Type: text/vnd.wap.wml");
?>
```

Here is how we do it with ASP:

```
<%
Response.ContentType = "text/vnd.wap.wml"
%>
```

and very similar with JSP:

```
<%@ page contentType="text/vnd.wap.wml" %>
```

2.3.2 Page cached by the phone

In order for our phone not to cache a page, we have to pass the correct information as part of the http header. This is possible through the use of server side scripting languages such as PHP, ASP or Perl. If we're using PHP, the following would do the trick:

```
<?php
header("Content-type: text/vnd.wap.wml"); // set the correct MIME type
header("Expires: Mon, 26 Jul 1997 05:00:00 GMT"); // expires in the past
header("Last-Modified: " . gmdate("D, d M Y H:i:s") . " GMT"); // Last modified, right now
header("Cache-Control: no-cache, must-revalidate"); // Prevent caching, HTTP/1.1
header("Pragma: no-cache"); // Prevent caching, HTTP/1.0
?>
```

Here's an example using ASP:

```
<%
```

```
Response.ContentType = "text/vnd.wap.wml"  
Response.Expires = -1  
Response.AddHeader "Pragma", "no-cache"  
Response.AddHeader "Cache-Control", "no-cache,must-revalidate"
```

2.3.3 Possible to do with WAP

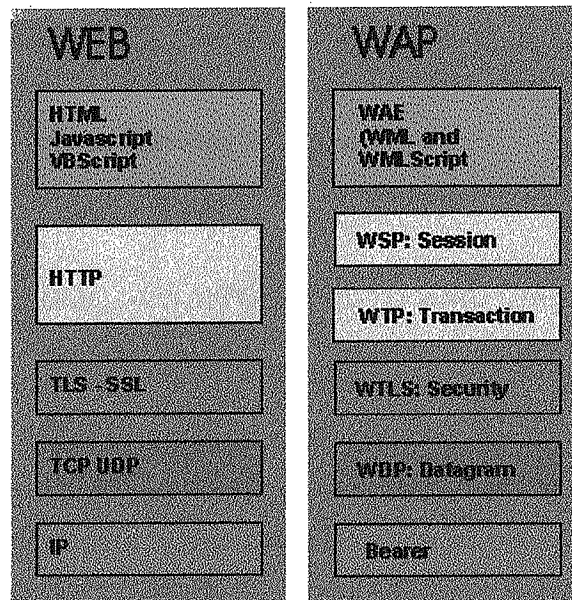
The probably most significant feature of the WAP 1.2 standard is the ability to push content from a server to one or many clients through a Push Proxy Gateway. Unlike an SMS, which also is pushed to the clients, a phone will only accept content from a 'trusted' source. Pushing will significantly increase the usability of WAP, and make it more flexible than the Internet. A new 'User Agent Profile' specification lets the WAP devices tell more about their features and capabilities than is possible with WAP 1.1. That opens for more tailor-made content providing. The WML and WMLScript specifications will have some slight modifications.

CHAPTER THREE

3. WAP- AN EXTENSION OF THE INTERNET MODEL

3.1 WAP- An Extension of the Internet Model

The WAP model closely resembles the Internet model of working. In Internet a WWW client requests a resource stored on a web server by identifying it using a unique URL, that is, a text string constituting an address to that resource. Standard communication protocols, like HTTP and Transmission Control Protocol/Internet Protocol (TCP/IP) manage these requests and transfer of data between the two ends. The content that is transferred can either be static like html pages or dynamic like Active Server Pages (ASP), Common Gateway Interface (CGI), and Servlets. The figure3.1 helps draw a parallel to the Internet protocols. We can see how WAP extends or reuses Internet protocols to achieve mobile Internet access. The strength of WAP (some call it the problem with WAP) lies on the fact that it very closely resembles the Internet model. In order to accommodate wireless access to the information space offered by the WWW, WAP is based on well-known Internet technology that has been optimized to meet the constraints of a wireless environment. Corresponding to HTML, WAP specifies a markup language adapted to the constraints of low bandwidth available with the usual mobile data bearers and the limited display capabilities of mobile devices - the Wireless Markup Language (WML). WML offers a navigation model designed for devices with small displays and limited input facilities (no mouse and limited keyboard). WAP also provides a means for supporting more advanced tasks, comparable to those solved by using for example JavaScript in HTML. The solution in WAP is called WML Script. We will find more information about these later in the coming sections. The figure3.2 will give a clear understanding of the WAP model.



(Figure 3.1)

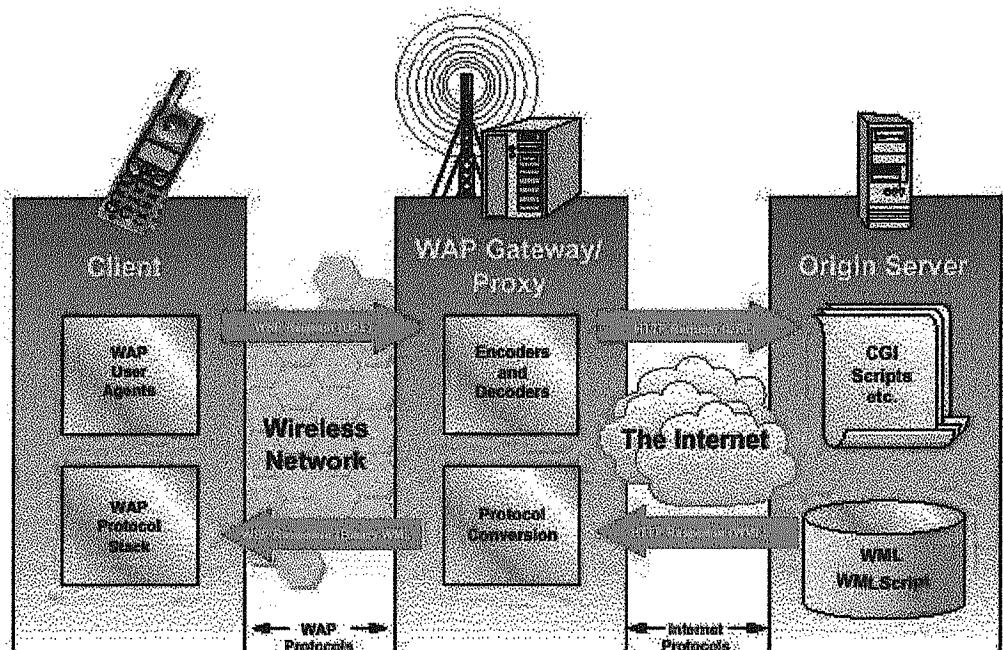


Figure (3.2) Source: Au systems white paper

3.2 What is the syntax of WML

A complete syntax reference is beyond the scope of this FAQ, but a general explanation will follow.

All decks must start with the following XML header:

```
<?xml version="1.0"?>  
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"  
"http://www.wapforum.org/DTD/wml_1.1.xml">
```

The first line gives the version number of the XML version used. The second line shows the URL to the Document Type Definition provided by W3C

Those two lines are convenient to have easily accessible, so you can cut and paste, no developer remembers all this stuff. The toolkits will help you there, with prewritten headers in every new file.

The body of a WML file starts with `<wml>`.

Between this and the end tag `</wml>` come the cards.

A card is defined like this:

```
<card id="card1" title="WML example">
```

Where you choose which id and title to give. The 'id' must be unique inside each deck, and is used for navigation between cards in a deck, we will come to this further down. The title is displayed on the top of the window in most WAP browsers, but not all, so do not display important info only in the title.

In order to display text inside a card, you must write a paragraph start and end tag and apply the text inside them:

```
<p>  
Here is the text!  
</p>
```

If you want the text to appear centered in the browser window, add the following attribute to the <p> tag:

`align="center"`

You can substitute 'center' with either 'left' or 'right' and watch the difference.

A line breaking tag is very useful, and looks like this:

`
`

Because WML uses XML syntax rules, tags that do not delimit content must have a '/' at the end.

Here is the source code for a simple WML deck which uses elements we have gone through so far:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="only" title="Test">
<p align="center">
This is the <br/>
only<br/>
card in this deck.
</p>
</card>
</wml>
```

Links can either be written like this:

`Linktext`

or like this:

```
<anchor><go href="http://domain.com/wmlfile.wml"/> Linktext</anchor>
```

Linking to another card in the same deck:

```
<a href="#cardID">Linktext</a>
```

A nice feature of WML is the timer. You can specify how long a card is displayed before the browser loads another card.

In this example, we want to navigate to the card named 'main' after 4 seconds:

```
<card id="intro" title="" ontimer="#main">
```

```
<timer value="40"/>
```

```
<p>
```

Welcome to us!

```
</p>
```

```
</card>
```

```
<card id="main" title="Main page">
```

```
<p>
```

This is our company WAP page.

```
</p>
```

```
</card>
```

The image tag is almost identical to HTML:

```

```

3.3 Bluetooth Technology

Automatic communication between various devices within a small area in a house or an office makes it possible to provide unique and innovative services to a professional worker or a small group of workers using portable devices. Bluetooth

technology has this potential and is coming along fast and quick. It will replace clumsy wires, make information transfer automatic without synchronization cradles and introduce many new applications. Technology visionaries hope that it will do what infra red could not do in past six years.

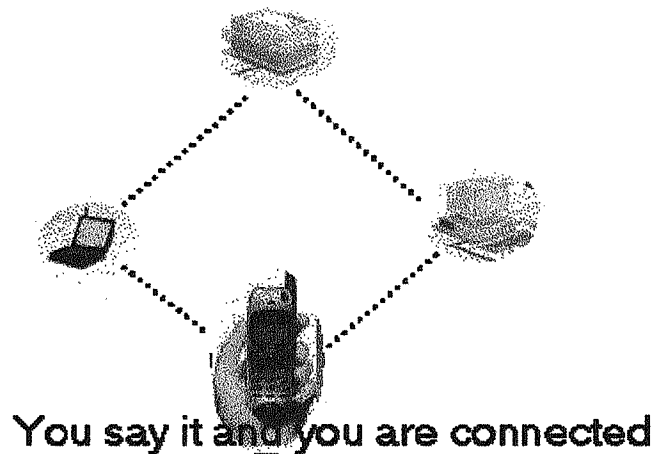


Figure 3.3

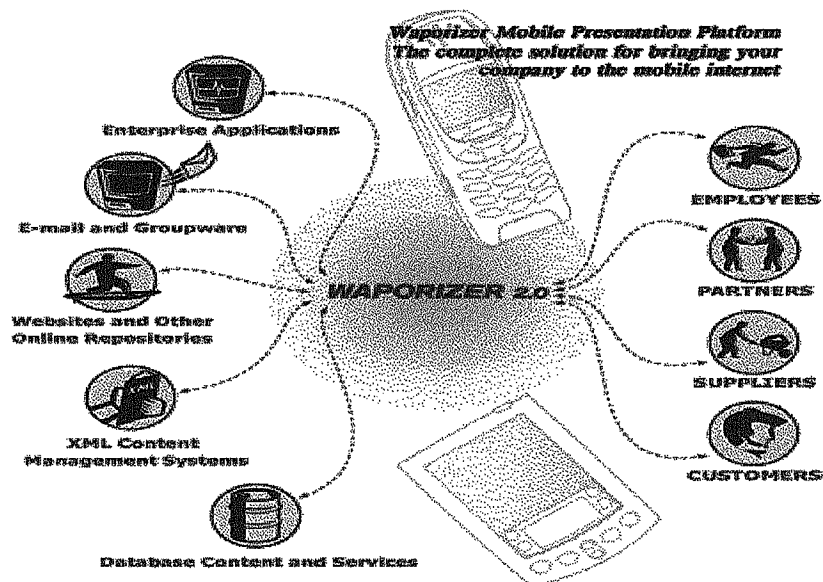
In an attempt to standardize data transfer and synchronization between disparate mobile devices *in the short-distance range*, Intel and Microsoft established in 1998 a major industry consortium that included IBM, Toshiba, Ericsson, Nokia, and Puma Technology.

Code-named Blue Tooth for the 10th century Danish king who unified Denmark, the companies have created a single synchronization protocol to address end-user problems arising from the proliferation of various mobile devices -- including smart phones, smart pagers, PDAs, handheld PCs, copiers, printers, notebooks, and many future digital appliances at home -- that need to keep data consistent from one device to another. The proposed Bluetooth solutions (hardware and software-based) would automatically synchronize mobile devices when end-users enter their offices or home. Intel and others are designing the sending and receiving radio frequency chip sets. Price point for hardware is in \$5--20 range eventually. Since the start of this

initiative in 1998, interest in Bluetooth has grown tremendously - signified by 1800 members of Bluetooth consortium by mid 2000.

While Bluetooth consortium demonstrated prototype products in the 1999-2000, there are no production-quality enduser products using blue tooth technology as of now, as far as we know. Component products (radios and chips) that can be integrated into finished products have started becoming available from Ericsson and others. However, here is an opportunity for more start-up companies. irDA is a competing technology and has been implemented in many products for over 6-7 years now but BlueTooth has a few distinct advantages - with Ericsson/Microsoft/Intel team behind it.

In our opinion, there are relative benefits with several competing technologies - there is some overlap too. Let competitive products thrive so that we the users get the best solutions.



Figure(3.4)

CHAPTER FOUR

4.WAP FIT IN THE WIRELESS COMPUTING APPLICATION

4.1 WAP Fit in the Wireless Computing Application

There are three essential product components that we need to extend our host applications and data to WAP-enabled devices. These three components are:

- WAP Microbrowser – residing in the client handheld device
- WAP Gateway – typically on wireless ISP's network infrastructure
- WAP Server - residing either on ISP's infrastructure or on enduser organization's infrastructure

4.2 WAP Micro-browser

A WAP micro-browser is a client software designed to overcome challenges of mobile handheld devices that enables wireless access to services such as Internet information in combination with a suitable network server. Lots of WAP browsers and emulators are available free of cost which can be used to test our WAP pages. Many of these browsers and emulators are specific to mobile devices. For example the R380s WAP emulator is intended to be used for testing WML applications developed for the WAP Browser in the Ericsson Smartphone R380s.

4.2.1 Warnings about GPRS and WAP Hype - Study

European wireless operators are deploying faster wireless General Packet Radio Service (GPRS), but like Wireless Application Protocol (WAP), the service is misunderstood, overhyped and is unlikely to achieve widespread acceptance, a new study predicts.

"WAP and GPRS represent major steps forward in cellular technology," a new study by Aberdeen Group says. "Their failure lies in the gross mismatch between reality and customer expectations. This situation cannot continue as the (wireless) industry, once saluted for its visionary capacity and slick delivery, now lurches from one problem to the

next."

The study, *Cutting EDGE: The 3G Alternative*, notes that "customer expectations of streaming video and exotic applications (via GPRS) are wildly off the mark." Among the reasons for that is, despite fast GPRS speeds in theory, actual speeds of live networks are estimated to be only 14Kbps to 28Kbps. "For subscribers accustomed to high-speed, fixed-wire Internet access, these transmission speeds are intolerable, even if GPRS is 'always on,'" the study notes. "Combined with the scarcity of compatible handsets and inflated performance expectations, these low transmission speeds put GPRS at an all-but-insurmountable disadvantage from the outset." The study suggests that GPRS will disappoint the industry in the same way as WAP did, which was launched with soaring expectations that have yet to be met.

"WAP (has) less than two percent penetration and lower than 30 percent repeat users -- hardly a commercial success," the study notes. The study concludes that GPRS is unlikely to reach 10 percent user penetration and it won't have much impact on existing low usage of the wireless Internet. Instead, technology using Enhanced Data Rates for GSM Evolution (EDGE), will quickly supplant GPRS, the study predicts. EDGE is expected to begin deployment in the third quarter of 2001, is another interim technology on the way to 3G and supports speeds as high as 380Kbps. Making matters worse for GPRS, major phone vendors such as Ericsson and Motorola have not announced many GPRS phones, the study says. It notes, however, that handset availability also is an issue for EDGE. As for WAP, the study notes its "underwhelming" acceptance in Europe, where the greatest hope was held for it. "The subscriber numbers (in Europe) are underwhelming, service offerings are spotty, and the lack of handset availability (initially, at least) has worked to undermine the launch," the report notes. "One explicit lesson (to operators) was the importance of ensuring the simultaneous launch of WAP infrastructure, handsets and applications. The report also notes that operators could have done a better job of explaining to users the nature of WAP access. Specifically, users were led to believe that the wireless Net was "the Internet made mobile" when, in fact, it was a far more truncated experience. "WAP's failure has created a high degree of subscriber skepticism regarding wireless data," the report notes. Wireless operators would do well to learn from the success of NTT DoCoMo's i-mode, the study says. For one thing, DoCoMo supported, but didn't control, implementation, which led to greater development. This reflects DoCoMo's understanding of the difference between operating a voice service and a data service, the study notes.

4.3 Nokia & WAP

Wireless Application Protocol (WAP) provides a universal open standard for bringing Internet content and advanced Value Added Services to mobile phones and other wireless devices. As such, WAP is at the heart of Total Connectivity usage.

StreetWise.ie is a small Irish company dedicated to the design and development of affordable mobile internet (and indeed conventional internet) systems in Ireland. WAP (Wireless Application Protocol) is the current standard for facilitating wireless Internet access on mobile phones and personal digital assistants and has been wholeheartedly embraced by the two major mobile phone operators in Ireland (and many others throughout the world). Arising from recent research into WAP systems, StreetWise.ie *Navigator* has been developed. StreetWise.ie *Navigator* is a free WAP based interactive street map information service. Geographical data is based on 1:50,000 vector detail provided by Ordnance Survey Ireland (permit number NE0001600). The current pilot system incorporates a 2 km square tile centred on the GPO, Dublin, Ireland. Wireless applications suggest somewhat of a paradigm shift from typical fixed-line internet systems. Principally, a mobile internet system is best suited as a provider of *location-relevant* or *timely* information. Implicit in this definition is the requirement for a real-time data facility between the mobile internet device and information server. We believe that this necessity for dynamic information coupled with the processing simplicity of client mobile devices requires a different set of skills than that needed for more typical internet applications: it is precise WAP architecture closely follows that of the conventional internet model. Applications are written in the Wireless Mark-up Language (WML) and WML Script and typically stored on a normal web server. The WAP Gateway acts as a gateway between the wireless cellular network and the internet. Data sent between the origin server and mobile device is binary encoded to optimise transmission over the narrow bandwidth of cellular network.

Since conventional web servers can be employed, a number of different technologies can be used to facilitate dynamic information flow between client and server such as ASP, Java Servlets, and Perl / CGI. However, due to restrictions on mobile devices, e.g. lack of processing power, limited screen size, certain precautions must be taken into consideration when designing effective wireless internet applications.

WAP applications can be made to run concurrently with existing internet sites or can be standalone. A number of Internet Service Providers (ISPs) now support the WAP protocol, a good example of which is IE Internet, where StreetWise.ie is hosted.

WAP Application Development SDKs & Higher Level Tools

Vendor	Tool
<u>Asialab</u>	HTML to WML Conversion Aid from <u>Asialab</u>
Ericsson	<u>Ericsson's WAP Development Environment</u>
Nokia's WAP Server	Nokia provides both client and server side application development tools including the communications interface. <u>Nokia's WAP server and application development tools</u> are being used by major systems integrators - definitely worth evaluating for enterprise applications.
Openwave	UP Software Development Kit - SDK - <u>OpenWave.Com</u> supplies an SDK for application development.
SpeedWare	MobileDev is the first Wireless Development Environment (WDE) specifically for WAP Internet applications. Its innovative open-ended development model integrates a graphical application mapper with a wizard interface and a rich tool set. Versatile enough to satisfy the most demanding professional, MobileDev supports WAP technologies like WML, HDML, Microsoft Active Server Pages (ASP), Perl and Java Server Pages (JSP).
SourceGear	<u>lloop</u> - an open source software toolkit for creating community portals, codenamed "lloop" has extensive support for WML, the markup language used by WAP phones and other wireless devices.

"loop" consists of a server platform and a collection of ready-for-use ASP-like scripts that produce dynamic content for display on a variety of devices, such as desktop PC's, WAP Phones, and the Palm VII handheld. The entire project is based upon Open Source technology and is licensed under the GNU Public License

WAPMinc.Com

WAPPage 2.0 is a WYSIWYG development tool for WAP (Wireless Application Protocol) Sites. WAPPage 2.0 allows a developer to edit, compile and integrate WML (Wireless Markup Language) Pages. WAPPage 2.0 supports WML 1.1.

Yospace

Smart Phone Emulator simulate a Nokia 7110 (and others shortly) smart phone while developing and demonstrating WAP capabilities.

4.4 WAP Applications and Examples

Examples suitable for WAP applications are limited only by one's imagination. Possible systems include:

- Traffic information
- Shopping guides
- Entertainment guides
- Tourist information
- Customer / sales information databases
- Financial information
- Ordering services (so called m-commerce)
- Geographical information*
- E-mail services*
- Games
- Translators

* denotes applications developed by StreetWise.ie

“At first, the most popular mobile Internet service is likely to be e-mail. SMS (short message service) messages have proved a big success in the Nordic nations and volumes are growing rapidly throughout western Europe”

One of the most significant advantages of Internet access from mobile rather than our PC is the ability to instantly identify users geographic location. This opens up a huge opportunity for highly customized services.

As Ericsson puts it,

“The content providers will know where their users are geographically and will be able to direct them to specific destinations - restaurants or theaters, for example handheld devices are mobile, but their position is instantly identifiable. So think of content that knows where the user is, and offers content tailored to that geography. Weather forecasts, restaurant locations (with table availability and instant reservations fast food delivery, finding and booking a plumber, dating services (with pre-recorded video profiles and e-mail or voicemail exchanges) any service where physical proximity is important can migrate a vital part of its value-added to the new devices.”

4.5 Interesting applications of WAP

Some of the interesting applications of WAP (already existing or being worked on) are:

- ✓ Computer Sciences Corporation (CSC) and Nokia are working with a Finnish fashion retailer who plans to send clothing offers direct to mobile telephones using a combination of cursors, touch screen technology and WAP to allow would-be shoppers to hot-link to order-entry pages on the web.
- ✓ In Finland, children already play new versions of competitive games such as "Battleships", via the cellular networks. In the music world, Virgin Mobile in the UK offers to download the latest pop hits to customers in a daily offering.
- ✓ Scala has developed several WAP products for small to medium-sized companies which would allow, for example, a field sales force to access customer order information and stock availability details via a WAP handset.
- ✓ A key growth area for the technology will be business-to-workforce, with companies using WAP applications to reach employees at any time. Scala is


currently working on time-sheet applications and techniques for entering and filing expense claims via the mobile phone.

- ✓ Nokia says applications that will benefit from WAP include customer care and provisioning, message notification and call management, e-mail, mapping and location services, weather and traffic alerts, sports and financial services, address book and directory services and corporate intranet applications.
- ✓ As brought out by [1] and the examples above WAP services are currently limited to simple information services, but as higher speeds become available and some of the technical issues specific to WAP are resolved, several new service types will emerge, including:
 - ✓ Infotainment : They could include weather forecasts, stock quotes, horoscopes and news
 - ✓ Messaging : services such as e-mail, voicemail and unified messaging
 - ✓ Personal information management : services such as call management and personal directories, which enable the modification of personal information
 - ✓ Financial services : mobile banking and mobile e-commerce services
 - ✓ Location-based services : services that are dependent on location include mapping and vehicle location information

The (table4.1) below should give an idea about the kind of services being offered by operators around the world.

(Table4.1) Source: Ovum (WAP Market Strategies/B

Examples of the current services offered
Cinema information, share prices and some banking services
Movie information, hotel information and news
News, flight information, lottery information and movie listings
Share prices, news, weather forecasts and e-mail
Cinema information, last minute offers (via lastminute.com), news and business directories
News, e-mail, SmarTone WAP portal, language translation



and directories
Information services, e-mail, directories and financial data
News, share prices, mapping, weather forecasts, directories and e-mail
News, traffic updates, some banking transactions, directories and e-mail

4.6 The Problem areas

One of the problem, basically to do with infrastructure (and not WAP) is that as the mobile Internet access, thanks to WAP, increases it is likely to put ever greater demands on existing technology infrastructures as it encourages higher m-commerce volumes. A live example is I-mode services in Japan, where the mobile data access has seen a unprecedented rate of growth. So, unless the infrastructure is geared up to expect unexpected volumes, this can have significant impact on these data services since most of these systems are simply inadequate for big volumes. So there is a possibility of unsatisfactory performances observed by mobile data users.

Another problem area is that the delay in the delivery of long-promised terminals and service launches are narrowing the window of opportunity for WAP, while the proposed developments in faster mobile networks and more sophisticated terminals come closer. Further developments in WAP are still required and in the meantime, other solutions will emerge.

Also as with many other technologies what matters most and what guides the development of a technology is the emergence of "killer applications". So, unless some killer applications hit the market, which influence the mood of the enduser, WAP just like other technologies has a difficult path ahead. Already due to lots of hype WAP proponents find them selves in a little tight position. So, this presents a big opportunity for the developer community to develop new and innovative applications that can realize the advantage of WAP. There is going to be big appetite for WAP applications in the very near future.

CHAPTER FIVE

5. WAP COMMUNICATION PROTOCOL & ITS COMPONENTS

The WAP Protocols cover both the application (WAE), and the underlying transport layers (WSP and WTP, WTLS, and WDP). WML and WMLScript are collectively known as WAE, the Wireless Application Environment. As described earlier the 'bearer' level of WAP depends on the type of mobile network. It could be CSD, SMS, CDMA, or any of a large number of possible data carriers. Whichever bearer our target client is using, the development above remains the same. Although it's not absolutely essential for a developer to know the details of the WAP communication protocols, a brief understanding of the various protocols involved, their significance and the capabilities can help a lot while looking for specific solutions. In this section I'll briefly describe the protocol components we say in the figures above.

5.1 WAE & WML

WML and WMLScript are collectively known as WAE, the Wireless Application Environment. WML is the WAP equivalent to HTML (HyperText Markup Language). It is a markup language based on XML (eXtensible Markup Language). The WAE specification defines the syntax, variables, and elements used in a valid WML file. WML 1.1 Document Type Definition (DTD) Note that XML is a meta-language defined by the W3C. This means that it is a series of rules for how to create other languages for specific applications. Content is not directly encoded in XML, but in a specific markup language defined, using XML. WML is an example of a specific language for wireless applications that is fully compliant with XML's rules. WML is thus an XML application.

For those who know HTML, WML should be quite easy to learn since many tags and attributes in WML are almost the same as in HTML, but there are fewer tags. Unlike HTML, we can play around with variables in WML, making it more dynamic. In WML it is also possible to have many sub-pages (cards) in one WML page (deck). Each WML card works like a web page and its content is displayed to the user.

WML introduces the concept of decks and cards. A WML deck consists of one or several cards, and each card functions like an independent page, with its own title and preferences. This approach helps conserve valuable bandwidth because so the HTTP header information is not occupying bandwidth for each single little page.

WML is an application of XML. So WML is more XML than HTML, but the syntax of WML is a lot like HTML. Where as in HTML one can take some liberties of skipping some end tags and omit quotation marks, the rules in WML are much stricter. In WML documents, the DOCTYPE definition must always be written in the beginning of each deck, after the XML declaration, while this is optional in HTML documents. The head tag in WML is not used very often. Instead, http header information is sent using some server-side script language. In WML the `<card>` tag replaces the `<head>` tag in HTML. There is no body tag in WML.

In WML, it is possible to use variables, which opens for slight dynamic functionality.

My aim is not to teach we how to code in WML and WML script, I'll briefly talk about the WML syntax to give a better feel of WML.

All decks must start with the following XML header:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
```

The first line gives the version number of the XML version used. The second line shows the URL to the Document Type Definition provided by W3C

The body of a WML file starts with `<wml>`.

Between this and the end tag `</wml>` comes the cards.

A card is defined like this:

```
<card id="card1" title="WML example">
```

The 'id' must be unique inside each deck. Whereas the id is used to navigate between different cards in a deck the title is not that significant. It actually specifies the text that is presented on top of the window in WAP browsers but not all browsers display it.

In order to display text inside a card, we must write a paragraph start and end tag and apply the text inside them:

```
<p>
```

This is card1 text!

```
</p>
```

align= "center" or "left" or "right" and line break
 tags do the normal formatting. (tags that do not delimit content must have a '/' at the end)

The image tag is similar as in HTML.

```

```

Note that there is no closing tag for the tag, but as we can see, the tag ends with a / (slash) unlike the tags that need a closing tag

Links can either be written like this:

```
<a href="http://domain.com/wmlfile.wml">Linktext</a>
```

or like this:

```
<anchor><go href="http://domain.com/wmlfile.wml"/> Linktext</anchor>
```

```
<a href="#cardID">Linktext</a> provides a link to another card in a deck
```

We can use a timer to automatically load a card (same or other)

In the example below, the second card automatically loads after 4 seconds.

```
<card id="card1" title="" ontimer="#card2">
```

```
<timer value="40"/> → (Note the / at the end of timer tag)
```

<p>

This is card one!

</p>

</card>

<card id="card2" title="Card two">

<p>

This is card two!

</p>

</card>

5.2 Some editors we can use for creating WAP pages

✚ WAP Page

✚ WAPtor is a WYSIWYG (What We See Is what We Get) editor for writing WML

✚ For those developing WML with ASP, ASPEdit, Frontpage or Visual InterDev can be used.

✚ Other possible editors are: Notepad, TextPad, Emacs, SynEdit (for Java, Perl & PHP).

Note that there is a limitation on the maximum compiled deck code size. The size limitation varies from phone to phone. Maximum size for a compiled deck or image on Nokia 7110 is 1397 bytes. Maximum deck size on Ericsson R320 is 3000 bytes (excluding images). Therefore it makes sense to limit the size of our deck to less than 1300 bytes.

5.3 WML SCRIPT

Just as javascript allows client side processing for use with HTML, WMLScript (Wireless Markup Language Script) is a client side script language for use with WML. JavaScript can be used to make the pages more dynamic, and to perform advanced

mathematical calculations in HTML pages. WMLScript is very similar to JavaScript and therefore easy to learn for those who have used that before. WMLScript is actually based on ECMAScript (which is based on Netscape's JavaScript language), however it has been modified in places to support low bandwidth communications and thin clients. WMLScript makes minimal demands on memory and CPU usage, while omitting a number of functions that are not required from other scripting languages. WMLScript is integrated with WML in a particularly flexible way for developers.

Server based computation means that several round trips to and from mobile to server have to be made in case of any interaction or computation. This is not very desirable in case of low bandwidth systems. WMLScript allows code to be built into files transferred to *mobile client* so that many of these round-trips can be eliminated. WML script also allows developer to provide interactivity in WAP pages without taxing the very valuable air interface. A big difference with javascript though, is that WMLScript must be kept in a separate file, and can not be used with inline tags like JavaScript. WMLScript can perform mathematical calculations, manipulate strings of text, make redirections etc.

5.4 WBMP

WBMP stands for Wireless BitMaP. It is the default picture format for WAP. The current version of WBMP is called type 0. WBMPs are uncompressed, monochrome black/white bitmaps intended for use in devices with small screens and narrow bandwidth connection. The constraints when using WBMP are the small screen size, limited graphics capabilities and the limited bandwidths available. As a thumb rule, a WBMP should not be wider than 96 pixels and higher than 48 pixels (at 72 dots per inch). Nokia's toolkit have a nice WBMP editor where we can draw and edit. Other WBMP tools create the WBMPs from BMPs, GIFs or JPGs. There are also plug-ins available for Paintshop, Photoshop and Gimp, which let us save WBMP files with these programs.

- i. [Teraflops online converter](#)
- ii. [pic2wbmp](#)
- iii. [WAP Pictus](#)
- iv. [WAP Draw](#)

- v. WBMPconv & UnWired plug-in for Photoshop/PaintShop
- vi. Plug-in for Gimp
- vii. Teraflops online converter
- viii. Applepie Solutions

To convert our existing pictures to WBMP A look at the following pictures will give we a clear idea about graphics and graphics support in WAP. Remember that WAP is designed to enable Internet access in low bandwidth systems. Therefore the idea is providing content fit for low bandwidth and low-resolution devices rather than providing high-resolution content. If we are confused as to which picture is in WBMP format (figure 5.1).

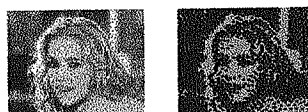


Figure 5.1

5.5 WTLS

The Security layer protocol in the WAP architecture is called the Wireless Transport Layer Security, WTLS. The WTLS layer operates above the transport protocol layer. The WTLS layer is modular and it depends on the required security level of the given application whether it is used or not. WTLS provides the upper-level layer of WAP with a secure transport service interface that preserves the transport service interface below it. In addition, WTLS provides an interface for managing (e.g., creating and terminating) secure connections.

WTLS is fast becoming the de facto standard for providing privacy, data integrity, and authentication for applications in cellular phones and other small wireless terminals. WTLS bears a close resemblance to the SSL and TLS protocols. Already WTLS has found place in a large number of mobile devices. WTLS, though similar to SSL and TLS has some differences basically because of the nature of mobile data

communications. The differences arise due to specific requirements of the WTLS protocol because of the constraints presented by the mobile data systems.

- ❑ Long round-trip times.
- ❑ Memory limitations of mobile devices
- ❑ The low bandwidth (most of the bearers)
- ❑ The limited processing power of mobile devices
- ❑ The restrictions on exporting and using cryptography

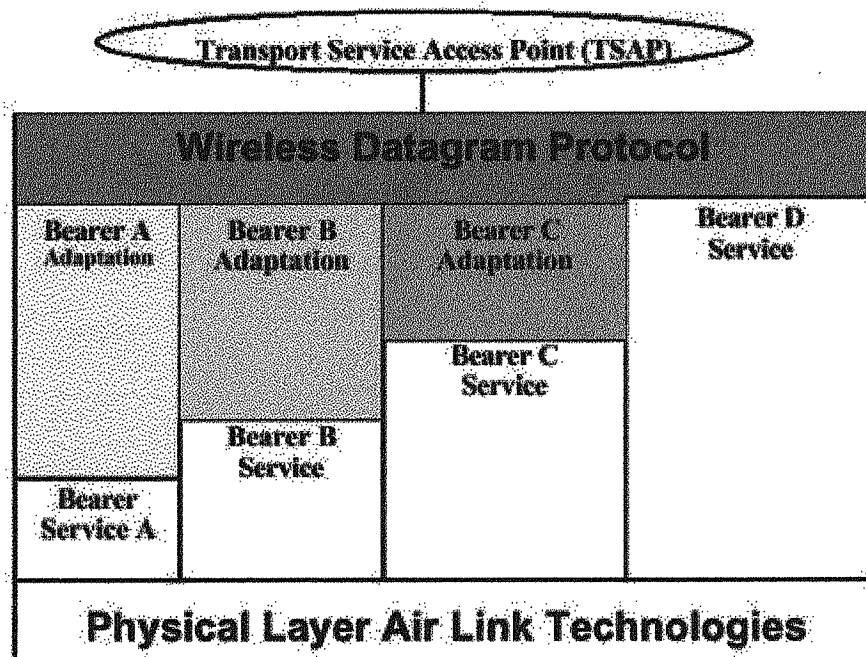
In other words, design of WTLS is based on the requirements to provide datagram support, optimize the packet size, have an optimized handshake mechanism, provide dynamic key refreshing and select fast algorithms into the algorithm suite. The primary goal of the WTLS layer is to provide privacy, data integrity and authentication between two communicating applications.

WTLS is designed to function on connection-oriented and/or datagram transport protocols. Security is assumed to be an optional layer above the transport layer. The security layer preserves the transport service interfaces. The session or application management entities are assumed to provide additional support required to manage (e.g., initiate and terminate) secure connections. Using WTLS the client and server agree on protocol options to use. The negotiation may include the security parameters (e.g., cryptographic algorithms and key lengths), key exchange and authentication. Since this exchange of parameters involve several steps it is possible for either client or server to terminate the negotiation process at will (if e.g. some parameters are not supported).

5.6 WTP

The Transport layer protocol in the WAP architecture consists of the Wireless Transaction Protocol (WTP) and the Wireless Datagram Protocol (WDP). The WDP protocol operates above the data capable bearer services supported by multiple network types. As a general datagram service, WDP offers a consistent service to the upper layer protocol (Security, Transaction and Session) of WAP and communicate transparently over one of the available bearer services.

The protocols in the WAP family are designed for use over narrow band bearers in wireless telecommunications networks. Since the WDP protocols provide a common interface to the upper layer protocols (Security, Transaction and Session layers), they are able to function independently of the underlying wireless network. This is accomplished by adapting the transport layer to specific features of the underlying bearer. WDP services include application addressing by port numbers, segmentation and reassembly of messages and error detection (optional). As we mentioned above, WDP can be mapped onto different bearers, with different characteristics. In order to optimize the protocol with respect to memory usage and radio transmission efficiency, the protocol performance over each bearer may vary. However, the WDP service and service primitives will remain the same, providing a consistent interface to the higher layers. This is visible in the figure above. WDP supports several simultaneous communication instances from a higher layer over a single underlying 'WDP' bearer service. The port number identifies the higher layer entity above WDP



(figure 5.3) Source: WAP- WDP Specifications

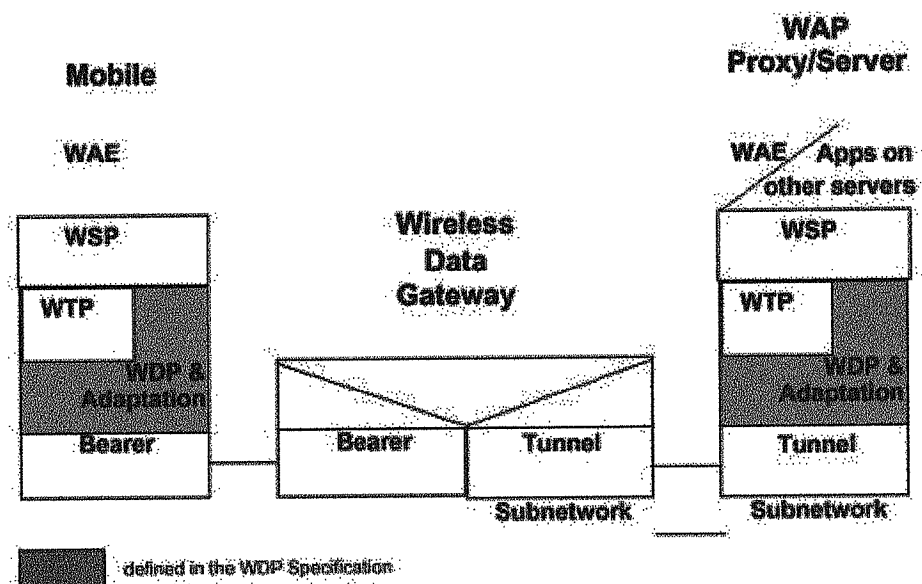
This may be another protocol layer such as the Wireless Transaction Protocol (WTP) or the Wireless Session Protocol (WSP) or an application such as electronic mail. By reusing the elements of the underlying bearers, WDP can be implemented to support

multiple bearers and yet be optimized for efficient operation within the limited resources of a mobile device. The figure 5.3,5.4 below describes the WDP architecture.

5.7 WAP address limitations of wireless Internet

WAP is based on existing Internet standards. The WAP architecture was designed to enable standard off-the-shelf Internet servers to provide services to wireless devices. While communicating with wireless devices, WAP uses many Internet standards such as XML, UDP, and IP. The WAP wireless protocols are based on Internet standards such as HTTP and TLS, but have been optimized for the unique constraints of the wireless environment.

The trouble with Internet standards such as HTML, HTTP, TLS, and TCP is that they require large amounts of mainly text based data to be sent which is a big problem in bandwidth constrained systems like mobile wireless systems. As an example HTTP sends its headers and commands in an inefficient text format instead of compressed binary. Another example is the TLS security standard that requires many messages to be exchanged between client and server. Also, standard HTML web content generally cannot be displayed in an effective way on the small size screens of



(figure 5.3) Source: WAP- WDP Specifications

CHAPTER SIX

6. WHAT ARE THE WAP PRODUCTS - MICRO BROWSER, WAP GATEWAY, WAP SERVERS

6.1 WAP Gateway

The idea behind WAP specifications is to connect the mobile networks to the Internet. To connect these two mega-networks, the WAP Specification assumes there will be a WAP Gateway. At its simplest level, this is a stack converter, which will convert the WAP request into a Web request and the Web response into a WAP response.

WAP Gateway is a piece of software that sits between the mobile device and the external network like the Internet. The gateway does the job of converting Internet content i.e. the WML pages into byte code (WMLC) which can be understood by a WAP device. Usually located on a server of a mobile operator it handles incoming requests from our WAP phone, takes care of the conversion required during WTLS/SSL sessions and handles incoming requests from our WAP phone. Although in theory, the gateway could also be made to convert the HTML page content itself on-the-fly as well, there are some problems. HTML pages can be full of graphics and with inline scripting. Converting these to WML may return something that is not of any relevance to anybody.

Some of the WAP Gateway products that are now coming on to the market (such as Nokia's WAP Server) also provide hosting capabilities themselves. In future it could be possible to integrate our WAP Server into the mobile network to gain information about the subscriber's location.

If we host our own gateways, then it may be required to maintain some sort of connection with the mobile network. For example, in case of GSM networks we may need to have say a dial up connection with the network's SMS engine or we may need to provide dial in modems for CSD access (Circuit Switched Data, around 9.6 kbps data rate)

pocket-sized mobile phones and pagers.. HTTP and TCP are not optimized for the usual problems associated with wireless networks like intermittent coverage, long latencies and limited bandwidth, which, with wireless transmission latencies, results in a very slow response for the user.

WAP uses binary transmission for greater compression of data and is hence optimized for long latency and low to medium bandwidth. WAP sessions cope with intermittent coverage and can operate over a wide variety of wireless transports using IP where possible and other optimized protocols where IP is impossible. The WML language used for WAP content makes optimum use of small screens and allows easy navigation with one hand. It also includes a built-in scalability from two-line text displays to the full graphic screens on smart phones and communicators.

There is a large list of available gateways at wapdrive.net. We can get our WAP gateway form

- ❑ Kannel
- ❑ Materna Carrier WAP gateway \
- ❑ Materna Corporate WAP gateway

A WAP server is simply a combined web server and WAP gateway. WAP devices do not use SSL. Instead they use WTLS. Most existing web servers should be able to support WAP content as well. Some new MIME types need to be added to our web server to enable it support WAP content. MIME stands for Multipurpose Internet Mail Extension, and in the web context, MIME can be thought of as a piece of header information that comes down with every file sent from a web server to a browser.

The following types basically need to be added:

- ❖ text/vnd.wap.wml for .wml files (WML source files)
- ❖ application/vnd.wap.wmlc for .wmlc files (WML compiled files)
- ❖ text/vnd.wap.wmlscript for .wmls files (WMLScript source files)
- ❖ application/vnd.wap.wmlscriptc for .wmlsc files (WMLScript compiled files)
- ❖ image/vnd.wap.wbmp for .wbmp files (wireless bitmaps)

On Microsoft Servers, there's a HTTP header tab on the Management Console. On Apache servers, we use an .htaccess file.

When we are accessing a secure service (using WTLS) the data is sent encrypted to the WAP gateway. The gateway decrypts it and encrypts it using SSL before passing it on to the web server. When the data returns, it will be decrypted and encrypted using WTLS before being sent to our WAP device. By combining the web server and the WAP gateway it is possible to enhance the overall security in these operations involve. see in figure 6.1.

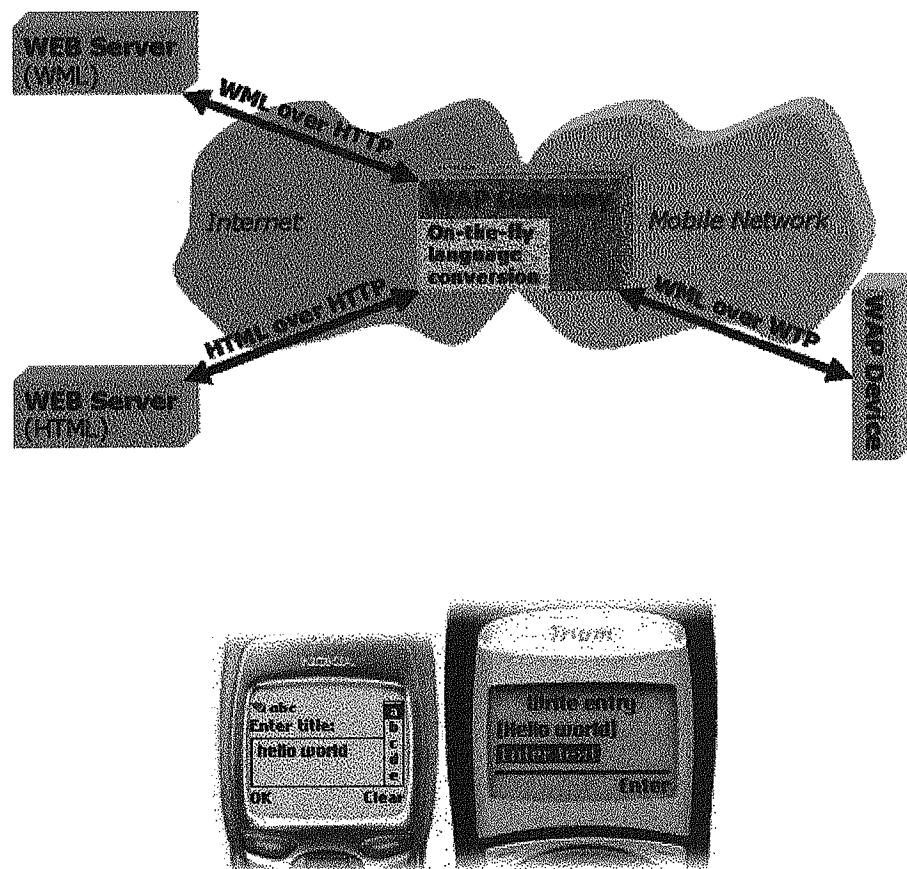


Figure 6.1

6.2 WAP server

A WAP server is simply a combined webserver and WAP gateway. One might wonder what the purpose of combining these would be, and the obvious answer is security. When we are accessing a secure service (using WTLS) the data is sent encrypted to the WAP gateway. The gateway decrypts it and encrypts it using SSL before passing it on to the webserver. When the data returns, it will be decrypted and encrypted using WTLS before being sent to our WAP device. It's easy to see where the weak link is: the WAP gateway. However, by using a WAP server, we've eliminated this weak link because we're not only in control of the content, but also the gateway where the encryption/decryption process takes place.

There are several WAP servers available:

- A. Nokia WAP server 1.1
- B. ANTAU Wireless Internet Platform

6.3 Gateways

- i. Columbitech
- ii. Ericsson WAP Gateway
- iii. Nokia WAP Server 2.0 - Professional and Enterprise Edition
- iv. Phone.Com's
- v. Lava NT WAP Gateway
- vi. ANTAU Wireless Internet Platform

6.4 Sending A Wireless Text-Message With Java

By Joe Lauer (joelauer@simplewire.com), President and Founder of Simplewire. Simplewire is a wireless text-messaging platform capable of messaging across all wireless networks regardless of carrier. After years of hype, the wireless market is finally upon us. Built upon competitive prices and a large growing user base around the globe, the wireless market is expected to grow at astronomical rates. The GSM Association anticipates monthly global SMS volumes to reach an amazing 25 billion mark by December 2001.

(http://www.gsmworld.com/news/press_2001/press_releases_4.html) Fortunately, for application developers, data and information services are experiencing the largest growth levels within the wireless market, thereby bringing new opportunities for revenue. However, even in the midst of "wireless" hype, many developers are still having difficulties navigating around the challenges of the esoteric SMS world. Obstacles such as fragmentation of carriers, inconsistent transport mechanisms and differing message lengths pose numerous complications. Developers who successfully surpass these obstacles are then faced with a new set of difficulties for building a solution in Java.

6.5 Challenges of sending an SMS Message from Java

Although SMS (Short Message Service) has proven itself as a robust medium for mobile information, there are many hidden problems preventing developers from using the technology in their applications. SMS is merely a description of services, which wireless carriers provide, rather than a description for a method of delivery. There are many different wireless networks that deliver SMS such as PDC, CDMA, TDMA, GSM or iDEN. These networks communicate over numerous protocols such as SMPP (Short Message Peer-to-Peer), UCP (Universal Computer Protocol), HTTP (Hypertext Transfer Protocol), SMTP (Simple Mail Transport Protocol), etc. Unfortunately, this lack of standards for SMS has increased development learning curves and slowed down adoption rates.

Inconsistent transports are only a small part of the larger dilemma. Typically, carriers do not allow public access to their networks and some will not even allow private access. This exclusivity leaves developers with very few options. Many developers opt to simply bypass the carrier and implement their own somewhat functional solutions. The problem is additionally magnified with the proliferation of various mobile devices, each built with differing technologies and capabilities. Some mobile phones support one-way text-messaging, while others support two-way text-messaging. Some devices are capable of receiving only 140 character messages, while others can receive 600 character messages. Other mobile phones can display images and download binary data, while others cannot. Because of these frustrating differences, the process of sending and receiving wireless messages with mobile devices requires the development and maintenance of extensive technological resources and information about each carrier and every device.

Aside from the process of sending and receiving text-messages, numerous extended services, which surround wireless devices, are also required in order to realize the full potential offered by today's wireless technologies. Wireless device metadata is essential in developing and deploying many new applications such as instant messaging, advertisements, location-based services, etc. Businesses and consumers are left with ad hoc solutions that fail to unleash the true power of today's mobile devices. It often becomes cost prohibitive for businesses to integrate emerging wireless technologies into their existing infrastructure. If true communication to any wireless

device, regardless of carrier, were possible, many past successes would seem miniscule in light of future opportunities.

6.6 Using SMTP to Send a Text-Message

Many wireless carriers now expose an SMTP (Simple Mail Transport Protocol) or an email interface to send SMS. In many cases, the email address will typically be the device's phone number or pager identification number along with the carrier's special domain. For example, a phone with Sprint PCS service will have an email address of 3135551212@messaging.sprintpcs.com. SMTP is an attractive method of sending SMS. The use of SMTP is free for the sender and software development kits for implementing the protocol are easily accessible. Unfortunately, most cost free solutions are littered with drawbacks, particularly in the use of SMTP. Disadvantages of SMTP include the lack of speed, assurance, error checking, features and consistency. By its original design, email was not developed to function as a quick system. As a result, most text-messages sent through email experience latency problems ranging from 1 minute to 2 hours. The speed of an email also relies on a dependable SMTP server-one that does not crash somewhere along the path of an email. Email cannot guarantee that the recipient of the message has subscribed to text-messaging in their carrier's normal service plan.

Typical SMS lengths range from 80 to 256 characters. Messages delivered via SMTP will normally eliminate portions of the message that go over the maximum character length. In many cases, the most important part of the message will be deleted unknowingly. The code will never discover the mishap. An additional conflict arises within the diversity of email formats used between services. In some instances, only the subject field or body field of a text-message will be used. SMTP code needs to account for these intricacies to assure that certain elements of the message will transmit to the mobile device. The From field showcases another imperfection when using SMTP. This field utilizes up to 20 to 30 characters within an already limited space of 80 to 256 characters. Beyond the problems associated with speed and assurance, SMTP fails to expose many of the best features specifically available with SMS such as one-touch callback and presence information (e.g. phone on/off or signal strength). Review the following sample Java code in order to learn how SMS might be delivered via SMTP.

The following example uses Sun Microsystem's JavaMail package to implement SMTP.

```
// Package Imports
import java.io.*;
import java.net.InetAddress;
import java.util.Properties;
import java.util.Date;
import javax.mail.*;
import javax.mail.internet.*;
import javax.activation.*;

// Public Class
public class EmailSMS {

    // Global Variables
    String      TO;
    String      FROM;
    String      SUBJECT;
    String      TEXT;
    String      MAILHOST;
    String      LASTERROR;

    // Main function executed
    public static void main(String[] args) throws Exception {
        EmailSMS SMS = new EmailSMS();

        SMS.setMailHost("mail.domain.com");
        SMS.setTo("1234567890@messaging.nextel.com");
        SMS.setFrom("name@domain.com");
        SMS.setSubject("");
        SMS.setText("Hello World!");
        boolean ret = SMS.send();
    }
}
```

```

        if (ret) {
            System.out.println("SMS was sent!");
        } else {
            System.out.println("SMS was not sent - " + SMS.getLastError());
        }
    }

// Public Constructor
public EmailSMS() {
    TO = null;
    FROM = null;
    SUBJECT = null;
    TEXT = null;
    MAILHOST = null;
    LASTERROR = "No method called.";
}

public void setTo(String to) {
    TO = to;
}

public String getTo() {
    return TO;
}

public void setFrom(String from) {
    FROM = from;
}

public String getFrom() {
    return FROM;
}

```

```

public void setSubject(String subject) {
    SUBJECT = subject;
}

public String getSubject() {
    return SUBJECT;
}

public void setText(String text) {
    TEXT = text;
}

public String getText() {
    return TEXT;
}

public void setMailHost(String host) {
    MAILHOST = host;
}

public String getMailHost() {
    return MAILHOST;
}

public String getLastError() {
    return LASTERROR;
}

// Will attempt to send the Email SMS and return a boolean meaning it
// either failed or succeeded.
public boolean send() {

    // Variables to check message length.
    int maxLength;

```

```

int msgLength;

// Check to make sure that the parameters are correct
if (TO.indexOf("mobile.att.net") > 0) {
    maxLength = 140;
} else if (TO.indexOf("messaging.nextel.com") > 0) {
    maxLength = 280;
} else if (TO.indexOf("messaging.sprintpcs.com") > 0) {
    maxLength = 100;
} else {
    maxLength = 160;
}

// Calculate message length
msgLength = FROM.length() + 1 + SUBJECT.length() + 1 + TEXT.length();

// Typically, there are at least two characters of delimiter
// between the from, subject, and text. This is here to make
// sure the message isn't longer than the device supports.
if (msgLength > maxLength) {
    LASTERROR = "SMS length too long.";
    return false;
}

// Set Email Properties
Properties props = System.getProperties();

if (MAILHOST != null) {
    props.put("mail.smtp.host", MAILHOST);
}

// Get a Session object
Session session = Session.getDefaultInstance(props, null);

```

```

try {

    // Construct the email
    Message msg = new MimeMessage(session);

    // Set From
    if (FROM != null) {
        msg.setFrom(new InternetAddress(FROM));
    } else {
        msg.setFrom();
    }

    // Set Subject
    msg.setSubject(SUBJECT);

    // Set Text
    msg.setText(TEXT);

    // Add Recipient
    msg.setRecipients(Message.RecipientType.TO,
        InternetAddress.parse(TO, false));

    // Sent Date
    msg.setSentDate(new Date());

    // Send Email SMS
    Transport.send(msg);

    LASTERROR = "Success.";
    return true;
} catch (MessagingException mex) {
    LASTERROR = mex.getMessage();
    return false;
}

```


6.7 Using a Third-Party Wireless Platform to Send a Text-Message

If SMTP falls short of easing our wireless pain, then the use of a third-party platform will solve our problem, while hiding the intricacies involved with SMS. Third party solutions often range from shrink-wrapped software to application service provider gateways. One such third-party platform is provided by Simplewire, Inc. Simplewire acts as an operating system between the Internet and any wireless device. Using Simplewire's Java Software Development Kit allows for the easy integration of powerful text-messaging services. Natively implementing text- messaging services into individual applications forgoes the need to worry about wireless technologies or varying protocols.

The advantages associated with the use of a third-party platform outweigh several of the short-term yet half-baked benefits of SMTP. Third-party platforms such as Simplewire's not only can consistently cover every major carrier, but also confirm message delivery while having faster performance than SMTP. The relatively simple solution of using a third-party platform solves a complex wireless problem for numerous developers. The following example demonstrates how to send a text-message using Simplewire's Java Software Development Kit:

```
// Imports
import com.simplewire.sms.*;

// Public Class
public class SimplewireSMS extends java.lang.Object {

    public static void main(String[] args) throws Exception
    {
        // Create SMS Object
        SMS sms = new SMS();

        // Set Message Properties
        sms.setMsgPin("1005101234");
```

i-MODE Links

Steps 1-5

Select 'i-MODE Services Links' and push the i-mode key.

Select one of the 2 items (Mobile Banking, Travel Service) and push the i-mode key.

```

sms.setMsgFrom("Joe");
sms.setMsgCallback("6165551212");
sms.setMsgText("Hello World From Java SMS!");

// Display The Status
System.out.println("Submitting SMS To Simplewire...");

// Send the request to simplewire
sms.msgSend();

// Check if the request was sent
if (sms.isSuccess())
{
    // Display the status
    System.out.println("The message was sent!");
}
// Display the error info
else
{
    // Check If Carrier Recognition Error
    if(sms.getErrorCode().equals("345"))
    {
        System.out.println("Sample request was success.");
        System.out.println("However, could not recognize carrier.\n");
    }
    else
    {
        // Display Error Info
        System.out.println("The message was not sent!");
        System.out.println("Error Code: " + sms.getErrorCode());
        System.out.println("Error Description: " + sms.getErrorDesc() + "\n");
    }
}
}
}

```

6.8 The players in this WAP game

"By 2004 mobile subscribers are expected to exceed 1 billion. This number represents about one sixth of world population. No wonder every one wants to cash in this huge opportunity". The mobile data opportunity is huge and needless to say, every one has interests. On the one side are hardware and technology vendors, the companies which supply the physical infrastructure for mobile networks and have developed enabling technologies such as WAP. The names include digital mobile phone manufacturers such as Nokia, Motorola and Ericsson. Then there are software suppliers developing wireless communicator operating systems to run on the next generation of hand-held devices. These include the Symbian partnership comprising Psion, Nokia, Ericsson, Motorola, Matsushita and Sony with the EPOC operating system, and Microsoft Windows CE. Application software vendors are also a very important part of the software community which have begun to offer applications software such as e-mail and web browsers. At the other end are the mobile phone network operators - Vodafone and others - which are expected to deliver content and value-added services such as news, share quotes, timetable and weather data to mobile devices with WAP.

6.9 I-mode

I-mode is a mobile Internet service from the Japanese tele provider NTT DoCoMo, started in February 1999. After one year, there were 5 million subscribers, now there are more than 10 million out of nearly 60 million mobile phone users. From being a pure Japanese service, i-mode is now expanding to other countries. The i-mode phone network is packet switched, so the users are charged for the amount of data they transfer, not the time they consume. The phones are always online (if they are within reach of the signals), and email can be pushed out to the recipients immediately. This makes email the most popular service. Other services are weather forecasts, ticket booking, online banking, stock trading, chatting etc.

i-mode terminal will let us play certain demonstration.
the instructions are below.

We will need to use 3 types of keys:

- The Menu key (situated to the left)

- The i-mode key (situated in the center)
- The UP/DOWN key (surrounding the i-mode key)

Going / Returning to the i mode menu

Push the i-mode (blue key) key to access the i mode menu from the Wallpaper screen. After each demo, the i-mode terminal will return to the initial wallpaper screen automatically.

i mode menu initialization steps

From the initial window, push the i-mode key.

1. Go to the i mode menu.
2. Using the Up or Down key to select the items you want, then push the i-mode key to confirm.
3. Select 'Internet' and push the i-mode key.
4. Select 'Bookmark' and push the i-mode key.
5. Select 'DoCoMo Europe HomePage' and push the i-mode key.

After connecting, select one of the 3 following items.

Setting of Wallpaper

Steps 1-5

Select 'Set Wallpaper' and push the i-mode key. Push the Menu key (left) then select 'Choose Picture'.

When 2 pictures appear simultaneously, use the UP/DOWN keys to toggle a black frame around one picture or the other.

Push the i-mode key to confirm your picture selection.

After the 'Picture Saved' message, push the i-mode key again to activate the new Wallpaper.

Melody Selector

Steps 1-5

Select 'Melody Selector' and push the i-mode key.

Select one music and push the i-mode key.

After the 'Download completed' message, select 'Play the melody' and push the i-mode key.

CONCLUSION

The Wireless Application Protocol (WAP) is a result of the WAP forum efforts to promote industry-wide specifications for technology useful in developing applications and services that operate over wireless communication networks. The wireless market is growing very quickly, and reaching new customers and services. To enable operators and manufacturers to meet the challenges in advanced services, differentiation and fast/flexible service creation WAP forum defines a set of protocols in transport, security, transaction, session and application layers.

WAP is a global standard and is not controlled by any single company. Ericsson, Nokia, Motorola, and Unwired Planet founded the WAP Forum in the summer of 1997 with the initial purpose of defining an industry-wide specification for developing applications over wireless communications networks. The WAP specifications define a set of protocols in application, session, transaction, security, and transport layers, which enable operators, manufacturers, and applications providers to meet the challenges in advanced wireless service differentiation and fast/flexible service creation. There are now over one hundred members representing terminal and infrastructure manufacturers, operators, carriers, service providers, software houses, content providers, and companies developing services and applications for mobile devices.

References

1. <http://www.dimonsoftware.com>
2. http://www.streetwise_ie.htm
3. <http://www.wapdrive.com/DOCS/wap/gateway.html>
4. <http://www.kannel.org>
5. http://www.annyyway.com/DOCS/wap_gateway.html
6. <http://lava2140.com/>
7. <http://www.phone.com/products/index.html>
8. <http://www.ericsson.com/cdmasystems/data/wap.stml>
9. http://www.gsmworld.com/news/press_2001/press_release_4.html
10. <http://www.tantau.com/product/products.asp>
11. <http://domain.com/wmlfile.vml>
12. http://www.wapforum.org/DTD/wml_1.1.xml
13. <http://www.wap.com>