



**Wireless Application Protocol
Technical Description
*Qwest Fusion White Paper***

Version 0.2

August 3, 2000

Chip Childers

MC³ - Mobile Computing, Communications, Commerce

Technical Architect / Development Team Lead

Qwest Fusion, Philadelphia

This Document

This is the first draft of this document. Please review/edit/chew on and return comments to Chip Childers (<mailto:chip.childers@qwest.com>).

This document is an overview of the technical environment known as WAP. It will explain the basic concepts of WAP and many of its related protocols as well as provide the locations of more detailed information on each topic.

Table of Contents

THIS DOCUMENT	2
<i>Table of Contents</i>	3
<i>Abbreviations</i>	4
THE PROGRAMMING MODEL	5
<i>Web Programming Model</i>	5
<i>WAP Programming Model</i>	5
<i>WAE</i>	6
THE CONTENT	7
<i>HDML vs. WML</i>	7
<i>WML Explained</i>	7
WML Entities.....	7
WML Data Types.....	8
WML Example	8
<i>WMLScript</i>	9
WMLScript Execution Model.....	9
The WMLScript Language.....	10
WMLScript Example	10
THE WEB SERVER.....	11
<i>Client Configuration</i>	11
<i>Server Configuration Issues</i>	11
MIME Configuration Table	11
Setting MIME Type in Generated Pages	11
Generating both HTML and WML from the same Web Server.....	11
THE WIRELESS TRANSPORT LAYER.....	12
<i>Wireless Transport Layer Reference Model</i>	12
<i>WSP</i>	12
<i>WTP</i>	13
<i>WTLS</i>	13
<i>WDP</i>	14

Abbreviations

DTD – Document Type Definition

HDML – Handheld Device Markup Language

HTML - HyperText Markup Language

HTTP – HyperText Transfer Protocol

HTTPS – HyperText Transfer Protocol over Secure socket layer

RFC – Request for Comments

TCP/IP – Transmission Control Protocol / Internet Protocol

TTML – Tagged Text Markup Language

URL – Uniform Resource Locator

WAE – Wireless Application Environment

WAP – Wireless Application Protocol

WDP – Wireless Datagram Protocol

WML – Wireless Markup Language

WMLScript – Wireless Markup Language Script

WSP – Wireless Session Protocol

WTLS – Wireless Transport Layer Security

WTP – Wireless Transaction Protocol

WWW – World Wide Web

XML – eXtensible Markup Language

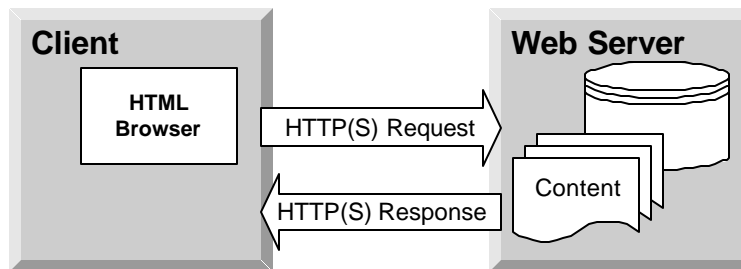
The Programming Model

It is important to understand the high level architecture of WAP before delving into some of the details. The details of the model will be discussed later in this document.

Web Programming Model

Because the WAP Programming model is based on the concepts and standards of the World Wide Web, it is important to briefly review the web programming model.

As shown below, the major architectural components of the web model are a client machine (running the web browser) and a web server (serving either static content or creating content for each request).

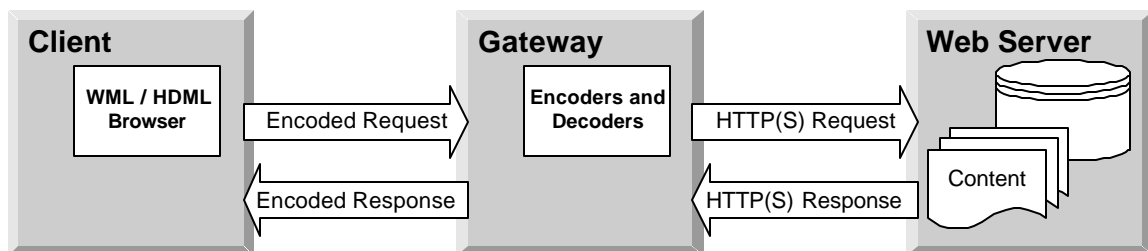


First, the client requests a resource from the web server using an HTTP or HTTPS request message. The web server then reacts to the request by retrieving or generating the content. The requested resource is then returned to the client machine for processing and rendering.

Although appearing simple, the web programming model, shown above, is rather complex at a detail level. Refer to "Hypertext Transfer Protocol: HTTP/1.1 RFC2068" (<http://www.cis.ohio-state.edu/htbin/rfc/rfc2068.html>) if this model requires further explanation.

WAP Programming Model

As stated above, the WAP programming model is based on the web programming model. The following diagram illustrates the model:



In order to request a WAP resource, the browser makes requests using the Wireless Session Protocol and the Wireless Transaction Protocol out over the wireless network utilizing a URL as the resource identifier.

The request is received initially by the gateway system. The gateway system is the translation mechanism from the WAP protocol stack (WSP, WTP, WTLS, and WDP) to the web protocol

stack (HTTP, HTTPS and TCP/IP.) Following the translation, the gateway forwards the request to the web server using either HTTP or HTTPS over TCP/IP (or, theoretically, over any protocol that HTTP/HTTPS can run on.)

The web server acts on the request similarly to any HTTP request. The resulting content has its "content type" header set to either WML or HDML and is formatted using the corresponding DTD.

The result is sent to the gateway, which performs the translation again (in reverse) from the web protocol stack to the WAP protocol stack. The gateway then transmits the encoded response to the client device.

WAE

WAE is the name given to the logical view of the components that may use the WAP and WWW network transport protocols. The full WAE specification can be found in the "Wireless Application Protocol, Wireless Application Environment Specification, Version 1.3" (<http://www1.wapforum.org/tech/documents/WAP-190-WAESpec-20000329-a.pdf>).

WAE contains the following major elements:

User Agents (browsers, phonebooks, message editors, etc...): User agents are WML or WTA devices (applications.) The WAE specification makes no requirement for the type of agent; it only specifies the minimum agent behaviors.

Content Generators (web servers or service based network servers): Content generators are the web servers of WAE and provide the content for the applications.

Standard Content Encoding (WML, WMLScript, image formats, etc...): The current recommended format for content is WML and the current recommended format for agent scripting is WMLScript. The vCard and vCalendar formats are used as content formats through WTA's.

Wireless Telephony Applications (vCard, vCalendar and other format specific application services): WTA's are applications specifically built for wireless telephony devices and extend standard device features through service subscriptions.

WAE does not include WSP, WTP, WTLS, WDP, HTTP(S) or TCP/IP (although it makes use of each as a network transport level.)

Also, currently most WAE interactions are user agent initiated, but WAP allows for the push of content or services to a user agent.

The Content

HDML vs. WML

Although there were other markup standards, proposed by many interested organizations, relating to wireless content formatting (such as TTML), the direct precursor to WML with the largest market share and technical promise was HDML. Therefore we will treat it as the earliest versions of WML.

Unwired Planet developed HDML in 1996 as a markup language specifically for wireless telephony devices. HDML was an excellent start at developing a suitable language for the emerging mobile communications market and WML has inherited many of its traits from HDML.

Currently WML is the recommended markup language for WAE applications and most wireless devices utilize the language, although there are some devices that only support the older HDML standard.

Before developing any wireless application, research should be completed determining the level of markup language sophistication the target audience's devices can process. This is much the same as any web project determining the level of HTML, JavaScript, and browser support required to reach the intended audience.

WML Explained

WML is a language derived directly from XML and inheriting many of the traits of HDML and HTML. The latest and complete WML specification can be found in the "Wireless Application Protocol, Wireless Markup Language Specification, Version 1.3" (<http://www1.wapforum.org/tech/documents/WAP-191-WML-20000219-a.pdf>).

The character set was inherited directly from XML: ISO/IEC10646 (currently identical to Unicode 2.0). Character encoding was inherited from HTML4 and WML fully supports all encoding standards supported by HTML4. Determining the character encoding follows the XML specification, not the HTML method of checking the meta http-equiv statement. Like HTML, transcoding is not recommended due to potential loss of information. Once WML is transformed into a format other than XML, that format's encoding rules apply.

The full XML character entity set is supported by WML.

WML follows the same URL standard for locating resources as the web protocol stack. URL's are utilized to specify navigation and external resources (i.e. images).

The navigation structure of WML follows a deck / card paradigm. The deck represents the downloaded document and the cards represent the individual screens of the document. The HTML concept of fragment anchors (the "#" character) is utilized by WML to navigate and reference the cards in the deck. A reference to the deck without the specification of a card (anchor to the card) represents a reference to the first card in the deck.

WML Entities

Elements are utilized in a similar manner to well formed HTML documents (proper WML formatting), establishing the structure and markup of the document.

```
<tag> content </tag>
```

Additional information about the elements can be placed within attributes.

```
<tag attribute="attribute value"> content </tag>
```

Comments are allowed in accordance with the XML specification.

Variables are used to pass values into the cards or deck. The "\$" symbol is used to signify a variable. Variables may be placed within PCDATA blocks and element attributes defined with the vdata entity type. The escaping of "\$" symbols is an important consideration when a variable is not intended.

WML Data Types

The CDATA and PCDATA types are the containers for literal text blocks. CDATA is used in attributes and PCDATA is used within elements.

Length is a formatting data type representing a number of screen pixels or a percentage of the existing screen pixels.

Flow represents information pertaining to the cards such as actions and events.

Id and class attributes are used to identify elements. The id of an element is a unique identifier. The class of an element is the group (or groups) that the element belongs to.

WML Example

Here is a short example WML deck taken from Steve Mann's [Programming Applications with the Wireless Application Protocol](#). In this example, the similarities between HTML and WML should be readily apparent as well as the specific XML structure that makes WML unique.

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

<!-- the first card -->

  <card id="card1" >
    <onevent type="onenterforward">
      <refresh>
        <setvar name="origin" value="Card # one" />
      </refresh>
    </onevent>
    <onevent type="onoenterbackward">
      <refresh>
        <setvar name="origin" value="Card # one" />
      </refresh>
    </onevent>

    <p>
      Select a destination:<br/>
      <a title="Card 2" href="#card2">
        Card # two
      </a>
      <br/>
      <a title="Display" href="#card3">
        Card # three
      </a>
    </p>
  </card>

<!-- the second card -->
```

```

<card id="card2" >
  <onevent type="onenterforward">
    <refresh>
      <setvar name="origin" value="Card # two" />
    </refresh>
  </onevent>
  <onevent type="onenterbackward">
    <refresh>
      <setvar name="origin" value="Card # two" />
    </refresh>
  </onevent>
  <p>
    Select a destination: <br/><br/>
    <a title="Card 1" href="#card1">
      
    </a>
    <br/>
    <a title="Display" href="#card3">
      
    </a>
  </p>
</card>

<!-- the third card -->

<card id="card3" >
  <p>
    You've just come from $origin <br/>
    Select a destination: <br/>
    <a title="Card 1" href="#card1">
      Card # one
    </a>
    <br/>
    <a title="Card 2" href="#card2">
    </a>
  </p>
</card>
</wml>

```

WMLScript

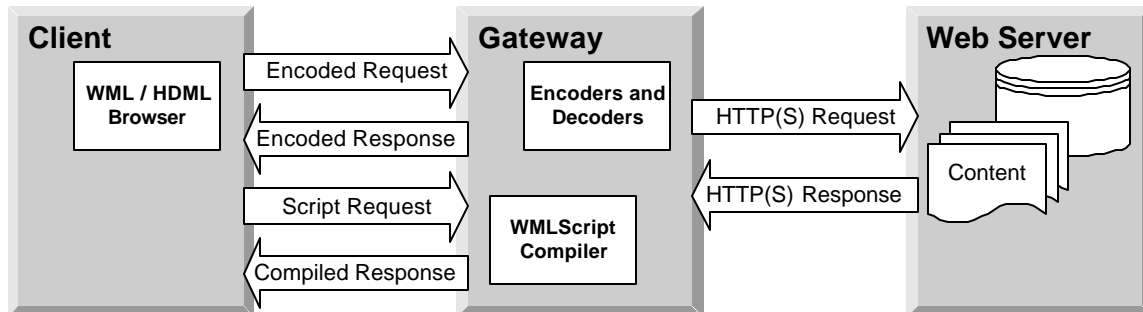
WMLScript is the scripting language developed to compliment WML. It is based on ECMAScript with many modifications for the wireless environment's architectural and functional limitations. The latest and complete WMLScript specification can be found in the "Wireless Application Protocol, WMLScript Language Specification, Version 1.2" (<http://www1.wapforum.org/tech/documents/WAP-193-WMLScript-20000324-a.pdf>).

WMLScript is meant as a general scripting language for WAP. In addition to general programmatic functionality, WMLScript extends the WML functionality by providing the following: validation of user input, enabling access to device resources (such as the phonebook, call manipulation and messaging), generation of content by the device, extending the device's functionality or software and adjustment of the device settings.

It is important to note that WMLScript does not execute within a WML document, but acts as a complimentary (and more powerful) content source for the device.

WMLScript Execution Model

The WMLScript execution model adds to the WAP model slightly as seen below:



The client requests a function point from the gateway. The gateway retrieves the compilation unit source code and compiles the requested function point for more efficient data transport over the wireless network. The client then executes the returned (compiled) script.

The WMLScript Language

The core ECMAScript specification has been left intact within the WMLScript specification, so the learning curve from JavaScript to WMLScript is minimal. See the WMLScript specification for a full explanation of the language syntax.

Pragmas are available in WMLScript allowing external scripts, referenced through a URL, to be included within the script block. The retrieval of these external function points is handled in the same way internal execution points are handled.

WMLScript Example

Here is a short example WMLScript compilation unit from Steve Mann's [Programming Applications with the Wireless Application Protocol](#).

```

/*****
Comments
*****/
extern function InitBoard () {
  var shiploc, shiplocs, x, y;
  var width = WMLBrowser.getVar ( "iwidth" );
  var height = WMLBrowser.getVar ( "iheight" );
  width = Lang.parseInt ( width );
  height = Lang.parseInt ( height );

  // Other Comments...

  WMLBrowser.setVar ( "gNumMoves", 0 );
  WMLBrowser.setVar ( "gBoardWidth", width );
  WMLBrowser.setVar ( "gBoardHeight", height );
  for ( y = 0; y < height; y++ ) {
    x = Lang.random ( width - 1 );
    shiploc = MapCoords ( width, x, y );
    shiplocs = shiplocs + shiploc + ",";
  }
  WMLBrowser.setVar ( "gShipLocations", shiplocs );
  WMLBrowser.setVar ( "gLine1", SetLineLength ( 0 < height, width ) );
  WMLBrowser.setVar ( "gLine2", SetLineLength ( 1 < height, width ) );
  WMLBrowser.setVar ( "gLine3", SetLineLength ( 2 < height, width ) );
  WMLBrowser.setVar ( "gLine4", SetLineLength ( 3 < height, width ) );
  WMLBrowser.setVar ( "gLine5", SetLineLength ( 4 < height, width ) );
  WMLBrowser.setVar ( "gLine6", SetLineLength ( 5 < height, width ) );
  WMLBrowser.go ( "bships2.wml#display" );
  WMLBrowser.refresh();
}

```

The Web Server

Because of WAP's reliance on Internet standards, any web server on the Internet has the potential to serve WAP pages to wireless devices. Both the client device and the server need to be configured to request / handle WAP data through URL's.

Client Configuration

Certain digital wireless carriers have wireless Internet subscription plans that do not allow the user to input URL's directly. In order for an application to be available to WAP users, their device must either have your site as part of its provider-configured library or the option to directly access applications through a URL.

Server Configuration Issues

When configuring a web server to handle WAP requests, the server must have the WAP MIME types established and dynamically generated pages must assure that the proper content type is returned to the requesting gateway.

MIME Configuration Table

In order to process WAP requests, your server must have the following entries in the MIME configuration table. If they are not present, they should be added. (The "hdml" and "wml/wmls" entries are not required at the same time if your application is only written in one or the other.)

<i>File Extension</i>	<i>MIME Type</i>
.wml	text/vnd.wap.wml
.wmls	text/vnd.wap.wmlscript
.hdml	text/x-hdml
.bmp	image/bmp

Setting MIME Type in Generated Pages

Be sure to set the HTTP content type whenever generating the page from a script file. This can be done either programmatically in each page or (if the server only serves one MIME type from generated scripts) through the web server's settings.

The content type should equal the MIME type set in the above table definition.

Generating both HTML and WML from the same Web Server

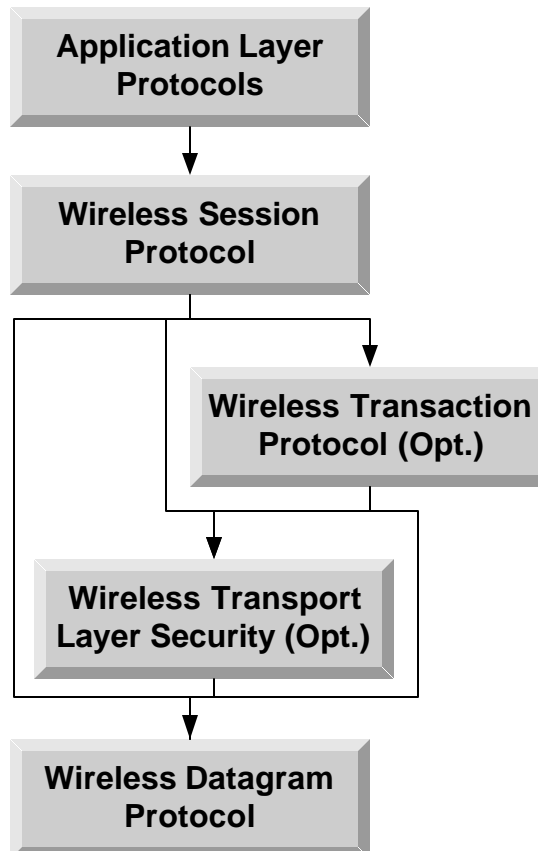
A web site can serve content to both HTML browsers and WML or HDML browsers. The default page for the site must contain script that detects the requesting browser type and returns or redirects to the proper content pages.

There are several available WAP browsers available, so the application needs to be sure to include all possible types.

The Wireless Transport Layer

There are four protocols defined to manage the transport of data over the wireless network and interact with corresponding protocols on the traditional network: WSP, WTP, WTLS and WDP. This section will provide an overview of how the protocols interact with each other as well as some basic concepts specific to each.

Wireless Transport Layer Reference Model



The application layer protocols currently extend to WML, WMLScript, vCalendar, vCard, etc... They function independently from the wireless transport layer protocols and run over TCP/IP quite well.

It is the job of the transport layer to get that same functionality out to the wireless device and maintain the communication channels.

It should be noted that both WTP and WTLS are optional protocols.

WSP

The basic premise of WSP is to function as a binary form of HTTP over the wireless network. WSP functions as a full binary implementation of HTTP 1.1, exposing all the methods described

in the HTTP 1.1 specification.

This binary form allows for smaller and faster transmissions through the reduction of the overall data size and the further reduction of known header values.

WSP session lifecycles are the most apparent break from the HTTP standard and represent a significant improvement over HTTP for the wireless networks. Session lifecycles are not tied to the underlying protocols. This allows for suspended sessions to exist, saving the devices the overhead of constant, lower level communication connections. A lightweight session re-establishment protocol has also been developed to take the suspended session back into active mode much faster than the full session establishment handshaking.

Another interesting function of the protocol is its feature negotiation process. It allows devices on both ends to communicate, regardless of which contains the lowest level of functionality by meeting on the common ground.

This is the protocol that allows for both “push” and “pull” directional communications. A data push can be attempted with or without an established session. When attempted with a session, the client device returns a confirmation or receipt to the calling system. When push is attempted without a session, there is no confirmation of receipt.

The complete WSP specification can be found at <http://www1.wapforum.org/tech/terms.asp?doc=WAP-203-WSP-20000504-a.pdf>.

WTP

The WTP provides the services required by WSP to enable a request/response application. The term transaction refers to the combination of a reliable request/response pair.

The protocol uses a series of question / answer messages to transmit and conform each transaction. These messages are sent and received without a connection being established outside of the individual datagram connections.

The protocol has the ability to transport performance information, or any auxiliary information for that matter, during the last transmission of each transaction without adversely affecting the performance of the client device.

The aborting of a transaction can be user driven, allowing bandwidth to be quickly cleared for another request. This is accomplished through a “flushing” of all unsent and uncompleted data on both the client and server.

Both synchronous and asynchronous transactions are possible with this protocol, allowing the delayed confirmation of receipt.

The complete WTP Specification can be found at <http://www1.wapforum.org/tech/documents/WAP-201-WTP-20000219-a.pdf>.

WTLS

It is the WTLS system's responsibility to provide an optional wrapper around the lower level WDP connections and secure the communications occurring over those connections. The three security functions being accomplished through the WTLS are: data privacy, data integrity and authentication for communication between two machines.

The complete specification for WTLS can be found at

<http://www1.wapforum.org/tech/documents/WAP-163-WTLS-19991105-a.pdf>.

WDP

WDP is the lowest level protocol outside of the specific carrier technologies. It provides the datagram facilities to pass messages around the wireless network. Its strength is that it provides a consistent service to the higher level protocols regardless of the cellular system's technology.

WDP offers port numbering, packet segmentation and repackaging as well as error detection.

Each bearer system technology adoption of the WDP requires a unique implementation of the protocol.

Details on WDP can be found at <http://www1.wapforum.org/tech/documents/WAP-200-WDP-20000219-a.pdf>.