WAP WAE Specification

Version 24-May-1999

Wireless Application Protocol

Wireless Application Environment Specification

Version 1.1

Disclaimer:

This document subject to change without notice.

Contents

1. Scope

Wireless Application Protocol (WAP) is a result of continuous work to define an industry wide specification for developing applications that operate over wireless communication networks. The scope for the WAP Forum is to define a set of specifications to be used by service applications. 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 defines a set of protocols in transport, session and application layers.

Wireless Application Environment (WAE) is part of the WAP Forum's effort to specify an application framework for wireless terminals such as mobile phones, pagers, and PDAs. The framework extends and leverages other WAP technologies, including WTP and WSP, as well as other Internet technologies such as XML, URLs, scripting, and various media types. The effort enables operators, manufacturers, and content developers to meet the challenges in building advanced and differentiating services and implementations in a fast and flexible manner.

This specification represents the root document of the Wireless Application Environment (WAE) specification hierarchy, ie, the normative document hierarchy. The document represents the core of the WAE specifications, from which additional WAE specification documents have evolved. For this reason, chapters or sections in this document reference all other WAE specifications and certain sections may be moved to other specifications in the future. For a general overview of the overall WAE architecture, please refer to [WAEOVER]. For additional information on the WAP architecture, refer to "*Wireless Application Protocol Architecture Specification*" [WAPARCH].

2. Document Status

This document is available online in the following formats:

- PDF format at http://www.wapforum.org/.

2.1 Copyright Notice

2.2 Errata

Known problems associated with this document are published at http://www.wapforum.org/.

2.3 Comments

Comments regarding this document can be submitted to the WAP Forum in the manner published at http://www.wapforum.org/.

3. References

3.1 Normative References

[RFC822] "Standard for the Format of ARPA Internet Text Messages", STD 11, RFC 822, D. Crocker, August 1982. URL: http://info.internet.isi.edu/in-notes/rfc/files/rfc822.txt

[RFC2045] "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", N. Freed, et al., November 1996. URL: http://info.internet.isi.edu/in-notes/rfc/files/rfc2045.txt

[RFC2068] "Hypertext Transfer Protocol - HTTP/1.1", R. Fielding, et al., January 1997. URL: http://info.internet.isi.edu/in-notes/rfc/files/rfc2068.txt

[RFC2119] "Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997. URL: http://info.internet.isi.edu/in-notes/rfc/files/rfc2119.txt

[VCARD] "vCard - The Electronic Business Card", version 2.1,The Internet Mail Consortium (IMC), September 18, 1996, http://www.imc.org/pdi/vcard-21.doc

[VCAL] "vCalendar - the Electronic Calendaring and Scheduling Format", version 1.0, The Internet Mail Consortium (IMC), September 18, 1996, http://www.imc.org/pdi/vcal-10.doc

[WSP] "Wireless Session Protocol", WAP Forum, January 30, 1998. URL: http://www.wapforum.org/

[WAEOVER] "Wireless Application Environment Overview", WAP Forum, 30-April 30, 1998. URL: http://www.wapforum.org/

[WAPARCH] "Wireless Application Protocol Architecture Specification", WAP Forum, April 30, 1998. URL: http://www.wapforum.org/

[WBXML] "WAP Binary XML Content Format", WAP Forum, April 30, 1998. URL: http://www.wapforum.org/

[WDP] "Wireless Datagram Protocol", WAP Forum, April 30, 1998. URL: http://www.wapforum.org/

[WML] "Wireless Markup Language Specification", WAP Forum, April 30, 1998. URL: http://www.wapforum.org/

[WMLScript] "WMLScript Specification", WAP Forum, April 30, 1998. URL: http://www.wapforum.org/

[WTA] "Wireless Telephony Application Specification", WAP Forum, February 5, 1998. URL: http://www.wapforum.org/

[WTAI] "Wireless Telephony Application Interface Specification", WAP Forum, April 30, 1998. URL: http://www.wapforum.org/

3.2 Informative References

[RFC1808] "Relative Uniform Resource Locators", R. Fielding, June 1995.

URL: ftp://ds.internic.net/rfc/rfc1808.txt

[RFC2396] "Uniform Resource Identifiers (URI): Generic Syntax", T. Berners-Lee, et al., August 1998.

URL: ftp://ftp.isi.edu/in-notes/rfc2396.txt.

[UNICODE] "The Unicode Standard: Version 2.0", The Unicode Consortium, Addison-Wesley Developers Press, 1996. URL: http://www.unicode.org/

[XML] "Extensible Markup Language (XML), W3C Proposed Recommendation 10-February-1998, REC-xml-19980210", T. Bray, et al, February 10, 1998. URL: http://www.w3.org/TR/REC-xml

4. Definitions and abbreviations

All non-trivial abbreviations and definitions used in this document are listed in the following sections. The definitions section includes description of general concepts and issues that may be fully defined in other documents. The purpose of this section is merely to advise the reader on the terminology used in the document.

4.1 Definitions

The notation used in the specification part of this document uses the common elements defined here.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY" and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

Author - an author is a person or program that writes or generates WML, WMLScript or other content.

Bytecode - content encoding where the content is typically a set of low-level opcodes (ie, instructions) and operands for a targeted hardware (or virtual) machine.

Client - a device (or application) that initiates a request for connection with a server.

Client Server Communication - communication between a client and a server. Typically the server performs a task (such as generating content) on behalf of the client. Results of the task are usually sent back to the client (eg, generated content.)

Content - synonym for data objects.

Content Encoding - when used as a verb, content encoding indicates the act of converting a data object from one format to another. Typically the resulting format requires less physical space than the original, is easier to process or store, and/or is encrypted. When used as a noun, content encoding specifies a particular format or encoding standard or process.

Content Format - actual representation of content.

Content Generator - a service that generates or formats content. Typically content generators are on origin servers.

Device - a network entity that is capable of sending and receiving packets of information and has a unique device address. A device can act as both a client or a server within a given context or across multiple contexts. For example, a device can service a number of clients (as a server) while being a client to another server.

Origin Server - the server on which a given resource resides or is to be created. Often referred to as a web server or an HTTP server.

Resource - A network data object or service that can be identified by a URL. Resources may be available in multiple representations (eg, multiple languages, data formats, size, and resolutions) or vary in other ways.

Server - a device (or application) that passively waits for connection requests from one or more clients. A server may accept or reject a connection request from a client.

Terminal - a device typically used by a user to request and receiving information. Also called a mobile terminal or mobile station.

User - a user is a person who interacts with a user agent to view, hear, or otherwise use a resource.

User Agent - a user agent is any software or device that interprets WML, WMLScript or other content. This may include textual browsers, voice browsers, search engines, etc.

WML - The Wireless Markup Language is a hypertext markup language used to represent information for delivery to a narrowband device, eg, a phone.

WMLScript - A scripting language used to program the mobile device. WMLScript is an extended subset of the JavaScript™ scripting language.

XML - the Extensible Markup Language is a World Wide Web Consortium (W3C) proposed standard for Internet markup languages, of which WML is one such language. XML is a restricted subset of SGML.

vCalendar - Internet Mail Consortium (IMC) electronic calendar record.

vCard - Internet Mail Consortium (IMC) electronic business card.

4.2 Abbreviations

The following abbreviations apply to this document.

API	Application Programming Interface
BNF	Backus-Naur Form
CGI	Common Gateway Interface
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol [RFC2068]
IANA	Internet Assigned Numbers Authority
IMC	Internet Mail Consortium
LSB	Least Significant Bit
MSB	Most Significant Bit
PDA	Personal Digital Assistant
RFC	Request For Comments
URI	Uniform Resource Identifier [RFC2396]
URL	Uniform Resource Locator [RFC2396]
W3C	World Wide Web Consortium
WWW	World Wide Web
WSP	Wireless Session Protocol
WTP	Wireless Transaction Protocol
WDP	Wireless Datagram Protocol
WAP	Wireless Application Protocol
WAE	Wireless Application Environment
WTA	Wireless Telephony Application
WTAI	Wireless Telephony Application Interface
WBMP	Wireless BitMaP
XML	Extensible Markup Language

5. WAE Specification

The following sections provide a specification for the core components of Wireless Application Environment (WAE), including the Wireless Markup Language (WML), the Wireless Markup Scripting language (WMLScript), WAE User Agents and WAE supported media types.

5.1 General WAE Features

5.1.1 Session Layer Interface

The WML and the Wireless Telephony Application (WTA) user agents communicate using the Wireless Session Protocol (WSP) over one or more WSP sessions per user agent. This network communication is in the form of WSP/HTTP 1.1 headers and content. The WSP session is created and controlled by the Session Management Entity. The Session Management Entity is not defined within the WAP specification framework, and is implementation specific.

5.1.2 Basic Authentication Scheme

WAE user agents should implement Basic Authentication as specified in the HTTP 1.1 specification [RFC2068].

5.1.3 URL Schemes

The following standard URL scheme is defined for WAP User Agents:

http: This scheme identifies a particular URL syntax suitable for naming resources stored on HTTP origin servers (see [RFC2396]). The specification of an `http` scheme does not imply the use of a particular communication protocol between a phone and network gateway. The origin server specified by the URL may be accessed via a WSP-to-HTTP gateway (or proxy). Alternatively, the URL may specify an network server, which combines the function of WSP gateway and origin server into one entity. In this case, the resource is accessed directly across the WSP protocol.

Additional, non-standard URL schemes are defined to access client/terminal specific content within the WTAI specification see [WTAI]. Since these schemes are specific to a particular WAE user agent, they are not included in this section.

5.1.4 User Agent Characteristics

NOTE: the information presented in this section is preliminary and still under consideration and development. The contents are likely to change. Furthermore, it is likely that the changes made will not be compatible with the information presented here.

In order to optimise the WAE client-server model, a number of characteristics are sent from the user agent to the WAP origin server. These characteristics allow the origin server to avoid sending inappropriate content to the user agent. They also provide the server and gateway with a means of customising the response for a particular user agent.

The WSP layer provides typed data transfer for the WAE layer (see reference [WSP]). The WSP/HTTP 1.1 content headers are used to perform content negotiation and define character set encoding and language settings.

The origin server or WAP gateway may need or want to modify responses based on characteristics of the user agent. For each WAP-defined media type included in the WSP/HTTP `Accept` header, the user agent should include a parameter, named `uaprof`, specifying the URI for a profile specifying the user agent characteristics. The syntax and semantics of this profile will be specified in a separate document.

An example of an `Accept` header would be:

```
Accept:   application/x-wap.wmlc;uaprof=http://www.vendor.com/phone1,
          application/x-wap.wmlscriptc;uaprof=http://www.vendor.com/phone1,
          text/x-vcard,
```

```
text/x-vcal
```

For WAP-defined media types, the `uaprof` parameter, combined with the WSP/HTTP `Accept`, `Accept-Language` and `Accept-Charset` headers, should completely describe all negotiable characteristics of the content. The combination of these three items henceforth is collectively referred to as the "characteristic headers."

Gateways which receive a request using characteristic headers must preserve the headers in any requests to origin servers on behalf of that user agent, and also insert the `uaprof` parameter on any media types that will be converted into a type specified with the parameter. For example, if the user agent's `Accept` header specifies that it accepts `application/x-wap.wmlc` and a gateway requests media of type `text/x-wap.wml` from an origin server, then the gateway must copy the user agent's `uaprof` parameter to its `Accept` header.

Some gateways may cache content received from origin servers. If a user agent requests content that a gateway has cached and the request contains characteristic headers, the gateway must not provide cached content to a user agent unless at least one of the three following conditions is true:

1) The characteristic headers specified in the request are identical to the ones used when the gateway initially retrieved the content.

2) The characteristic headers specified in the request are identical to the ones used when the gateway initially retrieved the content except for the `uaprof` parameter, and the profile specified by the URI in the `uaprof` header is semantically equivalent to the profile specified by the URI in the `uaprof` header of the request when the gateway initially retrieved the content.

3) The gateway was able to guarantee through other mechanisms (e.g., analysis of HTTP metadata) that a new request to the origin server using the user agent's `Accept` header would result in the same content that the gateway has cached.

The use of WSP/HTTP headers is summarised below in table 5-1.

ID	User Agent Characteristic	Characteristic Data Type	Description/Example	Method for transmission
1	Character set / encoding	IANA character set name	Character set(s)/encoding scheme(s) supported by the client.	Accept-Charset WSP/HTTP header as defined by [RFC2068]
2	Language	IANA country code	Languages supported by the client.	Accept-Language WSP/HTTP header as defined by [RFC2068]
3	Media Type	IANA media type	Content formats/encoding supported by the client	Accept WSP/HTTP header as defined by [RFC2068]
4	WML Version	Version Number	Refers to WML language supported. The current version of WML is 1.0	Defined with user agent characteristics (see `uaprof` description in section 5.1.4)

ID	User Agent Characteristic	Characteristic Data Type	Description/Example	Method for transmission
5	WMLScript Version WMLScript Support Floating Point Support	Version Number	Refers to WMLScript Language Supported. The current version of WMLScript is 1.0. If a version is reported then the client supports scripting. Otherwise, the origin server must assume it doesn't support/want scripts. Augment version number. The current version of WMLScript with floating point support is 1.1F.	Defined with user agent characteristics (see `uaprof` description in section 5.1.4)
6	Standard Libraries Supported	Version Number	Refers to WMLScript Standard Libraries supported	Defined with user agent characteristics (see `uaprof` description in section 5.1.4)
7	WTA Version Event Tables Used	Version Number	Refers to WTA User Agent version supported. The current version of the WTA User Agent is 1.0. Augment version number. The current version of a WTA User Agent supporting event tables (support for both Server and Client centric mode, see [WTA]) is 1.0C.	Defined with user agent characteristics (see `uaprof` description in section 5.1.4)
8	WTAI Basic Version	Version Number	Refers to WTAI Basic library supported. The current WTAI Basic Library version is 1.0.	Defined with user agent characteristics (see `uaprof` description in section 5.1.4)
9	WTAI Public Version	Version Number	Refers to WTAI public lib supported. The current WTAI public lib version is 1.0.	Defined with user agent characteristics (see `uaprof` description in section 5.1.4)
10	List WTAI Net-Spec Versions List WTAI Network Specific Libraries	List of Version Numbers	Refers to WTAI net-spec lib supported. The possible values are: 1.0GSM, 1.0IS-136, 1.0IS-95, 1.0PDC. Any network specific version must build on top of Basic Library.	Defined with user agent characteristics (see `uaprof` description in section 5.1.4)

Table 5-1: User Agent Characteristics

5.1.5 Wireless Markup Language

The specification of the WML language is available in [WML].

5.1.6 WMLScript

The specification of the WMLScript language is available [WMLScript]

5.1.7 WAE User Agents

The WML User Agent is a fundamental component of the WAE. However, WAE is not limited to WML User Agents. WAE allows the integration of domain specific user agents with varying architectures and environments. In particular, a WTA (Wireless Telephony Application) User Agent and a WTAI (Wireless Telephony Application Interface) programming interface has been specified as part of the WAE specification for the mobile telephony environment. The WTAI functions allow authors to access and interact with mobile phone features (eg, call control) as well as other user agents such as phone book and calendar user agents not specified by WAE.

5.1.7.1 WTA User Agent

The WTA User Agent is not fully specified as part of the WAP Standards specification. A number of requirements and guidelines are specified in [WTA]. A specification of the Wireless Telephony Application Interface (WTAI), which is the WAP Telephony Value Added Service API, is available in [WTAI].

5.1.7.2 WML User Agent

The WML User Agent is not fully specified as part of the WAP Standards specification. A number of requirements and guidelines are provided as part of the WML language specification (see section 5.1.5) and the WMLScript language specification (see section 5.1.6).

5.1.8 WAE Media Types

WAE specifies or adopts a number of content formats that facilitate inter-operable exchange of data. The most important formats are the encoded WML and the WMLScript bytecode formats. The encoding of WML and WMLScript makes transmission of WML and WMLScript more efficient and minimises the computational efforts needed to execute them on the client.

WAE adopts an additional class of media types to facilitate the exchange of data objects between client and server or between two clients. These are currently limited to electronic business cards and electronic calendar objects (see below). Such objects may be exchanged using WDP datagrams or through a WSP session. In case of exchange over datagrams, a set of well-defined ports has been reserved for the exchange to allow interoperability between different implementations (see [WTP]).

Other content formats include the WAE image exchange format and application specific formats. In general the method of data exchange depends on the data type and the user agent involved.

5.1.8.1 Encoded WML format

The WML content format is defined in [WML] and [WBXML].

5.1.8.2 Encoded WMLScript format

The WMLScript content format is defined in [WMLScript].

5.1.8.3 The Electronic Business Card Format (vCard 2.1)

The vCard format was defined by the Versit Consortium and is currently administered by the IMC.

This format is described in [VCARD]. WAP support for vCard is specified in section 7.

5.1.8.4 The Electronic Calendar and Scheduling Exchange Format (vCalendar 1.0)

The vCalendar format was specified by the Versit Consortium and is currently administered by the IMC.

This format is described in [VCAL]. WAP support for vCalendar is specified in section 7.

5.1.8.5 Images

WAE provides a visual environment that is designed to address several competing requirements, including support for multiple pixels depths, support for colourspace tables, small encoding, very low CPU and RAM decoding and presentation demands and allowance for commonly available tools and support.

WAE meets these unique requirements by:

- Supporting standard WSP/HTTP media types for commonly used image formats, eg, *image/png.*
- Introducing an optimised bitmap format, the Wireless BitMaP (WBMP) (WSP/HTTP media type image/x-wap.wbmp).

WBMP is an encapsulation format, ie, a WBMP object is a wrapper object which maps the verbose headers of the full image format to an identification (or typing) of the contents. The actual image contents contain all other information, eg, colour table (if any), image bit planes, etc.

The WBMP specification is thus divided into two parts:

1. The generic header contains the following information, which is common to all image formats.
   - Type
   - Width and height
   - WBMP version number

   The `type` identifier denotes the format of the embedded image. Type `0` is currently specified (see Appendix A and section 6.1).
2. The type-specific formats specification, indicating the data format for a particular WBMP `type.`

The WBMP format supports the definition of compact image formats suitable for encoding a wide variety of image formats and provides the means for optimisation steps such as stripping of superfluous headers and special purpose compression schemes. This leads to efficient communication to and from the client and for efficient presentation in the client display.

A WBMP image has the following characteristics:

- Compact binary encoding
- Scaleability, ie, future support for all image qualities and types (colour depths, animations, stream data, etc.)
- Extensibility (unlimited type definition space)
- Optimised for low computational costs in the client.

5.1.8.6 Multipart Messages

WAE includes a multipart encoding specification, suitable for exchanging multiple typed entities over WSP. WSP translates the MIME multipart entity (see [RFC2045]) into a compact binary form, which is optimised for narrowband environments. See [WSP].

5.1.8.7 WTA Events

WAE defines a separate content type and encoding for delivering events from the WTA server/gateway to the WTA User Agent. See [WTA] for details on the event content type.

6. Wireless Bitmap Format

The WBMP format enables graphical information to be sent to a variety of handsets. The WBMP format is terminal independent and describes only graphical information.

6.1 WBMP Type Identifiers

The WBMP format is configured according to a type field value (TypeField below), which maps to all relevant image encoding information, such as:

- Pixel organisation and encoding
- Palette organisation and encoding
- Compression characteristics
- Animation encoding

For each TypeField value, all relevant image characteristics are fully specified as part of the WAP documentation. Currently, a simple compact, monochrome image format is defined within the WBMP type space.

See Table 6-1 for the type identifiers.

Image Type Identifier, multi-byte integer	Image Format description
0	B/W, no compression

Table 6-1: WBMP Image Type identifier assignments

When initialising a session with a WAP server, the user agent reports all supported WBMP types. These are communicated using standard `Accept:` or `Content-Type:` WSP/HTTP headers, eg,

```
Accept: image/vnd.wap.wbmp; level=0
```

Or

```
Content-Type: image/vnd.wap.wbmp; level=0
```

The levelparameter refers to the WBMP types described in Table 6-1. In the current version of the specification, only `level=0` is supported.

The specification of WBMP level 0 is given in Appendix A.

6.2 WBMP syntax

The following is a BNF-like description of the tokenized structure. The description uses the conventions established in [RFC822], except that the "|" character is used to designate alternatives and capitalised words indicate single-byte tokens, which are defined later. Briefly, "(" and ")" are used to group elements, optional elements are enclosed in "[" and "]". Elements may be preceded with <N>* to specify N or more repetitions of the following element (N defaults to zero when unspecified).

```
W-Bitmap  = Header Image-data

Header = TypeField FixHeaderField [ExtFields] Width Height

TypeField  = ‘Type of image which is defined in Section 6.1’

FixHeaderField = ‘Octet which is defined in Table 6-3’

ExtFields = *ExtFieldType00 | ExtFieldType01  | ExtFieldType10 | *ExtFieldType11

ExtFieldType00 = ‘Octet which is defined in Section 6.3’

ExtFieldType01 = ‘Octet which is defined in Section 6.3’

ExtFieldType10 = ‘Octet which is defined in Section 6.3’

ExtFieldType11 = ParameterHeader ParameterIdentifier ParameterValue

ParameterHeader = ‘Octet which is defined in Table 6-4‘

ParameterIdentifier = ‘Parameter identifier (US-ASCII string), length ≤ 8 bytes
defined in ParameterHeader’

ParameterValue = ‘Parameter value (alphanumeric string), length ≤ 16 bytes
defined in ParameterHeader’

Width  = ‘Horizontal width of the bitmap in pixels (Multi-byte integer)’

Height  = ‘Vertical height of the bitmap in pixels (Multi-byte integer)’

Image-data  = Main-image *Animated-image ;’There can be 0 to 15 animated images’

Main-image  = ‘Bitmap formed according to image data structure specified by the
TypeField’

Animated-image  = ‘Bitmap formed according to image data structure description
below’
```

6.3 Header Data Structure

6.3.1 Multi-byte Integer format

The WBMP image encoding uses a multi-byte representation for integer values. A multi-byte integer consists of a series of octets, where the most significant bit is the continuation flag, and the remaining seven bits are a scalar value. The continuation flag is used to indicate that an octet is not the end of the multi-byte sequence. A single integer value is encoded into a sequence of N octets. The first N-1 octets have the continuation flag set to a value of one (1). The final octet in the series has a continuation flag value of zero. The remaining seven bits in each octet are encoded in a big-endian order, ie, most significant bit first. The octets are arranged in a big-endian order, ie, the most significant seven bits are transmitted first. In the situation where the initial octet has less than seven bits of value, all unused bits must be set to zero (0).

For example, the integer value 0xA0 would be encoded with the two-byte sequence 0x81 0x20. The integer value 0x60 would be encoded with the one-byte sequence 0x60.

6.3.2 Header Formats

The header field contains an image type identifier of multi-byte length (`TypeField`), an octet of general header information (`FixHeaderFiled`), zero or more extension header fields (`ExtField`), a multi-byte width field (`Width`) and a multi-byte height field (`Height`).

Data Type	Length in Bits, h, k, m and n are arbitrary integers
TypeField(s)	8..8*h
FixHeaderField	8
ExtHeaderField(s)	0..8*k
Width	8..8*m
Height	8..8*n

Table 6-2: Length of Header Parts

The extension headers may be of type binary *00* through binary *11*, defined as follows.

- Type 00 indicates a multi-byte bitfield used to specify additional header information. The first bit is set if a type 00 extension header is set if more data follows. The other bits are reserved for future use.
- Type 01 - reserved for future use.
- Type 10 - reserved for future use.
- Type 11 indicates a sequence of parameter/value pairs. These can be used for optimisations and special purpose extensions, eg, animation image formats. The “parameter size” tells the length (1-8 bytes) of the following parameter name. The “value size” gives the length (1-16 bytes) of the following parameter value. The concatenation flag indicates whether another parameter/value pair will follow after reading the specified bytes of data.

The actual organisation of the image data depends on the image type.

Description	Bit
Ext Headers flag, 1 = More will follow, 0 = Last octet	7
Extension Header Type, msb (See Section 6.3)	6
Extension Header Type, lsb	5
Reserved	4
Reserved	3
Reserved	2
Reserved	1
Reserved	0

Table 6-3: FixHeaderField description

ExtField Type 11, description	Bit
Concatenation flag, 1 = More parameters follow, 0 = Last parameter assignment	7
Size of Parameter Identifier in bytes, msb	6
Size of Parameter Identifier in bytes	5
Size of Parameter Identifier in bytes, lsb	4
Size of Parameter Value in bytes, msb	3
Size of Parameter Value in bytes	2
Size of Parameter Value in bytes	1
Size of Parameter Value in bytes, lsb	0

Table 6-4: Extension Field Type 11

6.4 Image Data Structure

The data structure of the image data is dependent on the image type. Appendix A either is defines or references all image types supported by WAP.

6.5 Minimal Required Implementation

If a WAP device supports display of graphical images, it must support WBMP type 0 as defined in Appendix A.

7. Calendar and Phone Book

WAE includes support for the exchange of calendar and phone book data objects.

7.1 Data Formats

WAE has adopted the vCard and vCalendar data formats (see [VCARD] and [VCAL] for more information). These data formats are industry-standard means of exchanging phone book, electronic business card and calendar information, and are in use in a wide variety of devices and software. In this specification, the terms *electronic business cards, phone book information,* and *phone book data* are used to represent data encoded in the vCard format.

7.2 Data Transmission

There are currently two available methods for exchanging vCard and vCalendar data:

- Using WDP datagrams -- enables clients to communicate using vCard and vCalendar without the use of a WAP Gateway or other network proxy.
- Issuing WSP-based requests to a network server -- enables clients to communicate with applications and servers using vCard and vCalendar.

7.2.1 WDP Datagram Data Exchange

To accurately identify data exchanged in a WDP datagram packet, a well-known port number is used. This allows the user agent to presume that any data received on the port is in a particular format. All data must be sent to the correct port number, and data received on that port should be assumed to be of the associated type.

For the vCard and vCalendar port number assignments, please refer to the [WDP] specification.

When datagrams are received at the given WDP port, the information can be provided to a standard vCard or vCard reader, or it can be accessed by another application, such as the WML user agent. The method of display is implementation dependent.

7.2.2 WSP Data Exchange

The WSP Content-Type header identifies data exchanged in a WSP request. This header contains a MIME media type, indicating the data type in the associated WSP payload.

Data Type	MIME Media Type
vCard (phone book and business card data)	text/x-vcard
vCalendar (calendar data)	text/x-vcal

Table 7-1: MIME Media Types

In the situation where the Content-Type header is missing or unavailable, the user agent may use other methods of determining the type of the data. The file extension in the data name may provide an indication of the type as defined in Table 7-2.

Data Type	File Extension Name
vCard (phone book and business card data)	.vcf
vCalendar (calendar data)	.vcs

Table 7-2: File Extension Names

7.3 Required Terminal Behaviour

The following behaviour is defined independent of the data transmission method. The exchange of calendar, phone book or electronic business card data over WDP as described in section 7.2.1 must use the vCalendar and vCard data formats, respectively.

The following additional requirements apply to any terminal supporting phone book data exchange:

- Upon receipt of a vCard, the terminal must be able to display the vCard 'Name' and 'Telephone Number' properties to the user.
- Transmitted vCards must include the 'Name' and 'Telephone Number' properties.

The following additional requirements apply to any terminal supporting calendar data exchange:

- Upon receipt of a vCalendar data object, the terminal must at least be able to display the vEvent object to the user.

8. Client Header Handling

Client headers (as described in section 6.3.3.1 of [WSP]) that WAE user agents send as part of the connect service primitive of the WSP layer shall be used as cached request headers. This means that the request headers in each method invocation received during the lifetime of the established WSP session shall be combined with the cached request headers before the header information is used otherwise. If some header is present in the request headers, any headers with the same name in the cached request headers shall not be taken into account during the processing of the method invocation. If a header with a particular name is not present in the request headers but is present in the cached request headers, the processing of the method invocation shall occur exactly as if the method invocation had also included all the values of that header which are present in the cached request headers. This means that if a cached request header needs to be overridden for a request then the entire value of the original request header must be sent by the client. In the case of request headers that are lists (e.g. the Accept header) then the entire modified list must be sent.
Client headers that WAE user agents send as part of the resume service primitive of the WSP layer shall be used to update the cached request headers. The new cached request headers are created by combining the client headers in the resume primitive with the existing cached request headers using the same combination mechanism as is used with request headers in method invocations.

Example 1, Combining empty request headers with cached request headers:

Cached request headers:

Accept: image/x-wap.wbmp

Accept: text/x-wap.wml

Accept-Language: dk

Accept-Charset: iso-8859-1

[No additional request headers are needed[JeL2]]

Request headers:

empty

WAP Gateway generated headers:

Accept: image/x-wap.wbmp

Accept: text/x-wap.wml

Accept-Language: dk

Accept-Charset: iso-8859-1

Example 2, Changing the Accept-Language request header:

Cached request headers:

Accept: image/x-wap.wbmpAccept: text/x-wap.wml

Accept-Language: dk

Accept-Charset: iso-8859-1

[User changes the default language [JeL3]]

New request header:

Accept-Language: en

WAP Gateway generated headers:

Accept: image/x-wap.wbmp

Accept: text/x-wap.wml

Accept-Language: en

Accept-Charset: iso-8859-1

Example 3, Changing the Accept request header:

Cached request headers:

Accept: image/x-wap.wbmp

Accept: text/x-wap.wml

Accept-Language: dk

Accept-Charset: iso-8859-1

[User disables downloading images]

New request header:

Accept: text/x-wap.wml

WAP Gateway generated headers:

Accept: text/x-wap.wml

Accept-Language: dk

Accept-Charset: iso-8859-1

Appendix A: Specification of well-defined WBMP Types

WBMP Type 0: B/W, Uncompressed Bitmap

WBMP type 0 has the following characteristics:

- No compression
- Colour: one bit with white=1, black=0.
- Depth: 1 bit deep (monochrome)
- The high bit of each byte is the left-most pixel of the byte.
- The first row in the data is the upper row of the image.

The WBMP header encoding is shown in the Tables Table 8-1 and Table 8-2.

Data Type	Length in Bits, h and k are arbitrary integers
TypeField	8
FixHeaderField	8
ExtHeaderField(s)	0
Width	8..8*h
Height	8..8*k

Table 8-1: Length of Header Parts

FixHeaderField, description	Bit Value
Ext Headers flag, 1 = More will follow, 0 = Last octet	0
Extension Header Type, msb (See Table 6-4]	any value
Extension Header Type, lsb	any value
Reserved	0
Reserved	0
Reserved	0
Reserved	0
Reserved	0

Table 8-2: FixHeaderField description

No extension headers are required for this format.

The WBMP image data is organised in pixel rows, which are represented by a sequence of octets. One bit represents one pixel intensity with value white=1 and value black=0. In the situation where the row length is not divisible by 8 the encoding of the next row must start at the beginning of the next octet, and all unused bits must be set to zero (0). The data bits are encoded in a big-endian order, ie, most significant bit first. The octets are arranged in a big-endian order, ie, the most significant octet is transmitted first. The most significant bit in a row represents the intensity of the left most pixel. The first row in the image data is the top row of the image.