E-Mail Notification

Approved Version 1.0 – 19 Oct 2007

Open Mobile Alliance

OMA-TS-Push_EMN-V1_0-20071019-A

Contents

Figures

1. Scope

Open Mobile Alliance (OMA) 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 Open Mobile Alliance 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 providing new 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. For additional information on the WAP architecture, refer to "*Wireless Application Protocol Architecture Specification" [WAP]*.

This specification defines a content type that can be used to notify e-mail clients that new e-mail is available for retrieval on an e-mail server. The e-mail notification (EMN) content can be sent using the WAP Push [PUSH] framework to a device with an EMN User Agent. This specification defines the EMN User Agent to be the client-side application that handles the EMN and triggers the e-mail client for further action. This specification does not define any normative behaviour for e-mail clients nor e-mail servers, as both fall out of the scope of the WAP Forum. Furthermore, this specification is independent of e-mail protocols such as POP3 or IMAP4. Nevertheless, use cases are presented to identify possible usage of the e-mail notification.

The EMN content type is an application of the Extensible Markup Language (XML) 1.0 [XML]. WBXML [WBXML] tokens are defined to allow for efficient over-the-air transmission.

2. References

2.1 Normative References

[IOPProc] "OMA Interoperability Policy and Process". Open Mobile Alliance™. OMA-IOP-Process-v1_0. URL: http://www.openmobilealliance.org

[ISO10646] "Information Technology - Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane", ISO/IEC 10646-1:2000.

[OMNA] "OMA Naming Authority", OMA™,
URL: http://www.openmobilealliance.org/tech/omna/index.htm

[RFC2119] "Key words for use in RFCs to Indicate Requirement Levels". S. Bradner. March 1997.
URL: http://www.ietf.org/rfc/rfc2119.txt

[RFC2234] "Augmented BNF for Syntax Specifications: ABNF". D. Crocker, Ed., P. Overell.
November 1997. URL: http://www.ietf.org/rfc/rfc2234.txt

[RFC2396] "Uniform Resource Identifiers (URI): Generic Syntax". T. Berners-Lee et al. August 1998
URL: http://www.ietf.org/rfc/rfc2396.txt

[RFC3023] "XML Media Types", M. Murata et al., January 2001. URL: http://www.ietf.org/rfc/rfc3023.txt

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

[WBXML] "Binary XML Content Format Format", WAP Forum, WAP-192-WBXML-20010725-a, 25-Jul-2001,
URL: http://www.wapforum.org/

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

2.2 Informative References

[ISO8601] "Data elements and interchange formats - Information interchange - Representation of dates and times", International Organization For Standardization (ISO), 15-June-1988

"Data elements and interchange formats - Information interchange - Representation of dates and times, Technical Corrigendum 1", International Organization For Standardization (ISO) - Technical Committee ISO/TC 154, 01-May-1991

[RFC2192] "IMAP URL Scheme". C. Newman. September 1997. URL: http://www.ietf.org/rfc/rfc2192.txt

[RFC2384] "POP URL Scheme". R. Gellens. August 1998. URL: http://www.ietf.org/rfc/rfc2384.txt

[RFC2822] "Internet Message Format". P. Pesnick, Ed. April 2001.
URL: http://www.ietf.org/rfc/rfc2822.txt

[PUSH] "WAP Push Architectural Overview". WAP Forum™.
WAP-250-PushArchOverview-20010703-a. URL: http//www.wapforum.org/

[PUSHMSG] "Push Message". WAP Forum™.
WAP-251-PushMessage-20010322-a. URL: http://www.wapforum.org/

[PushOTA] "Push OTA Protocol". WAP Forum™.
WAP-235-PushOTA-20010425-a. URL: http//www.wapforum.org/

[PushPAP] "Push Access Protocol". WAP Forum™.
WAP-247-PAP-20010429-a. URL: http://www.wapforum.org/

[WAP] "WAP Architecture". WAP Forum™. WAP-210-WAPArch-20010712, URL: http://www.wapforum.org/

3. Terminology and Conventions

3.1 Conventions

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].

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.

3.2 Definitions

Application	A value-added data service provided to a WAP Client. The application may utilise both push and pull data transfer to deliver content
Client	in the context of push, a client is a device (or service) that expects to receive push content from a server. In the context of pull a client, it is a device initiates a request to a server for content or data. See also “device”.
Content	subject matter (data) stored or generated at an origin server. Content is typically displayed or interpreted by a user agent on a client. Content can both be returned in response to a user request, or being pushed directly to a client.
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.
Device	is a network entity that is capable of sending and/or receiving packets of information and has a unique device address. A device can act as either 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
End-user	see “user”
E-Mail Client	is a client-side application capable of accessing e-mail. This may include both applications specifically designed to access e-mail (e.g. by using POP3 for mail retrieval) or more generic applications such as a XHTML Mobile Profile capable User Agent (e.g. for “web mail”).
E-Mail Server	is a server capable of handling e-mail.
EMN User Agent	is a client-side application capable of receiving E-Mail Notifications (EMN) and triggering the E-Mail Client for appropriate action. The EMN UA may be an integral part of the E-Mail Client, but may also be a separate application.
Push Access Protocol	a protocol used for conveying content that should be pushed to a client, and push related control information, between a Push Initiator and a Push Proxy/Gateway.
Push Initiator	the entity that originates push content and submits it to the push framework for delivery to a user agent on a client.
Push OTA Protocol	A otocol used for conveying content between a Push Proxy/Gateway and a certain user agent on a client.
Push Proxy Gateway	a proxy gateway that provides push proxy services
Server	A vice (or service) that passively waits for connection requests from one or more clients. A server may accept or reject a connection request from a client. A server may initiate a connection to a client as part of a service (push).
User	a user is a person who interacts with a user agent to view, hear, or otherwise use a rendered content. Also referred to as end-user
User agent	a user agent (or content interpreter) is any software or device that interprets resources. This may include textual browsers, voice browsers, search engines, etc.

WAP Proxy	an intermediary program which acts as both a server and a client for the purpose of making requests on behalf of other clients. Requests are serviced internally or by passing them on, with possible translation, to other servers. It may provide functions of protocol enhancement, transcoding or any number of other optimisation or transformation functions and may be associated with any gateways, proxies or servers being used in the deployment architecture. WAP gateway is one of the optional functionalities of WAP proxy.

3.3 Abbreviations

DTD	Document Type Definition
EMN	E-Mail Notification
EMN UA	EMN User Agent
HTTP	Hypertext Transfer Protocol
IANA	Internet Assigned Numbers Authority
IMAP	Internet Message Access Protocol
IOProc	Interoperability Procedure
MIME	Multipurpose Internet Mail Extensions
OMNA	Open Mobile Alliance Naming Authority
OTA	Over The Air
PAP	Push Access Protocol
PI	Push Initiator
POP	Post Office Protocol
PPG	Push Proxy Gateway
RFC	Request For Comments
SGML	Standard Generalized Markup Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTC	Universal Time Co-ordinated
WAP	Wireless Application Protocol
WBXML	WAP Binary XML
XML	Extensible Mark-up Language

4. Introduction

E-mail has become a vital part of everyday life. More than simple messaging, e-mail, like the telephone, has become an indispensable communication tool. E-mail has been adopted as a messaging standard because of its low cost, simplicity, ease-of-use, and global compatibility. In that light, there is a clear need for this communication tool to be used in a mobile environment, allowing users to have access to e-mail anywhere. Although there are many mobile e-mail implementations already in use today, there is no standard method to notify an e-mail client that new e-mail has been received.

The value of providing e-mail notification is the same as providing voicemail notification. E-mail notification will allow mobile users to be notified of incoming e-mail upon reception. Linking a notification feature with existing Internet e-mail creates a value added service for the end-user by offering an "always-connected" feature to e-mail.

The purpose of this specification is to define a single format for the e-mail notification. The primary objective of this e-mail notification is to invoke the device to launch the e-mail client, which may then (depending on implementation and user settings) retrieve the e-mail. This will allow e-mail servers to send notifications in a standard way, without having to worry about various e-mail client implementations. The added value of specifying a notification mechanism is that it can be handled by the e-mail client transparently for the user, for example by retrieving the message before notifying the user, so that the e-mail is directly available for viewing.

This specification defines the EMN User Agent (EMN UA), which is the client-side application on the terminal that handles EMN. The specification further affects e-mail clients on the terminal as well as e-mail servers. However, it does not affect the e-mail protocol that is being used for transport of the e-mails themselves (e.g. POP3, IMAP4). More specifically, all functionalities of the e-mail client and the e-mail server beside the e-mail notification are unaffected by this specification; i.e. a user will still be able to send and receive e-mails in the same way she is used to from the Internet.

Figure 1 below shows a typical flow of events as they may take place after an e-mail is received by the e-mail server. In this example, the mobile e-mail client supports the POP3 protocol.

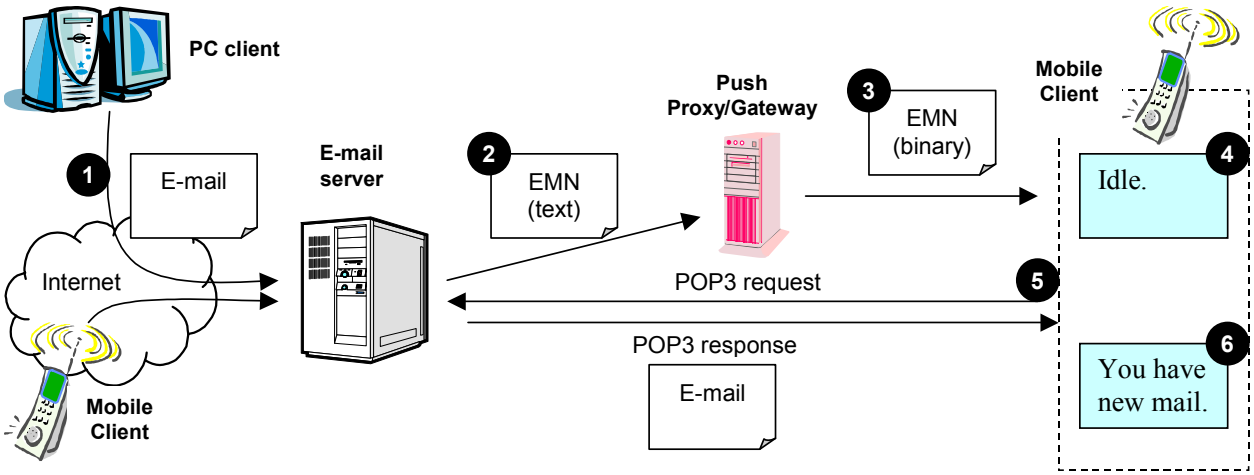


Figure 1: Example of automatic e-mail retrieval using EMN

The example illustrates how the e-mail client on the mobile phone is notified of the arrival of a new e-mail, retrieves the e-mail and then alerts the user that new e-mail has been received. Since the e-mail is already present on the handset, the user can immediately view it. The following steps are involved:

1. An Internet user sends an e-mail from his PC or his mobile e-mail client.
2. The e-mail server instructs the Push Proxy/Gateway to push an EMN to the mobile client using the Push Access Protocol [PushPAP].
3. The Push Proxy/Gateway sends the EMN to the EMN UA on the mobile device using the Push Over-The-Air protocol [PushOTA].

4. The EMN UAreceives the EMN and passes an appropriate trigger to the e-mail client for further processing. For the user, the mobile client is still idle.

5. In this example, the e-mail client is set up by the user to automatically retrieve all new e-mails. The e-mail client retrieves the e-mail from the server by sending a POP3 request for all new mails.

6. The new e-mail is stored locally on the mobile client and the e-mail client displays an alert indicating that a new e-mail has been received. This is the first indication to the user that new e-mail is available, i.e. all previous steps occur transparently for the user.

5. The E-mail Notification content format

This section defines the content format used to represent the E-Mail Notification (EMN), which is an application of XML version 1.0 [XML]. The complete E-Mail Notification DTD, which an implementation conforming to this specification MUST support, is defined in chapter 9. E-Mail Notifications that do not validate against this DTD MUST be discarded by the EMN UA.

5.1 E-Mail Notification Character Set

The document character set for EMN is the Universal Character Set, defined jointly by the Unicode Standard [UNICODE] and ISO/IEC 10646 [ISO10646]. A conformant EMN UA that interprets the EMN MUST support character encodings UTF-8 and UTF-16 as required by XML [XML]. Additional support MAY be available in an EMN UA using compatible character sets and encodings from which the characters can be transcoded into the Universal Character Set (e.g., US-ASCII, ISO-8859-1).

When processing XML documents, it is REQUIRED to treat character encoding of the document as specified in [RFC3023]. The EMN UA that interprets the EMN MUST silently discard the EMN if the XML document includes unknown characters.

Although listed as an optional parameter, the use of the charset parameter in the Content-Type field is strongly recommended, unless otherwise this information is reasonably embedded in the content (e.g., by a charset field in WBXML). An EMN UA or a WAP proxy uses this information to determine the character encoding of the content.

5.2 The EMN element

```
<!ELEMENT emn EMPTY>
<!ATTLIST emn
  mailbox     %URI;           #REQUIRED
  timestamp   %DateTime;      #IMPLIED
>
```

Attributes

`mailbox=%URI`

The syntax of the `%URI` value must follow the rules of [RFC2396]. This attribute indicates the e-mail account where the e-mail is to be fetched. This attribute is necessary because e-mail clients may support multiple e-mail accounts.

Usage of URI schemes is further specified in section 5.3.

`timestamp=%DateTime`

This attribute may be used to specify the date and time associated with the notification. It is used by the EMN UA to determine the age of the EMN in order to resolve race conditions between EMNs that arrive in an order different from the one in which the push initiator sent them. If the value of the `timestamp` attribute in a received EMN is older than this attribute value in any other EMN with an identical `mailbox` attribute value, the received EMN SHOULD be silently discarded.

If used, the attribute value MUST be expressed in a date/time representation based on [ISO8601] as specified below. EMN does not allow use of time zones; the time MUST always be expressed in Co-ordinated Universal Time (UTC), a 24-hour timekeeping system (indicated by the "`Z`"). The format is:

`YYYY-MM-DDThh:mm:ssZ`

Where: `YYYY` = 4 digit year ("0000" ... "9999")
`MM` = 2 digit month ("01"=January, "02"=February ... "12"=December)
`DD` = 2 digit day ("01", "02" ... "31")
`hh` = 2 digit hour, 24-hour timekeeping system ("00" ... "23")
`mm` = 2 digit minute ("00" ... "59")
`ss` = 2 digit second ("00" ... "59")
Note: "`T`" and "`Z`" appear literally in the string.

Example: "`2002-04-08T06:40:00Z`" means 6.40 in the morning UTC on the 8th of April 2002.

5.3 URI schemes

Two types of URI schemes can be used to identify the location of the e-mail:

1. *"mailat" URI scheme*:
   As default, this URI scheme should be used to specify the account to which new e-mail has arrived. The ABNF [RFC2234] format of this URI is:

   ```
   mailat-uri = "mailat:" addr-spec
   ; addr-spec as defined in [RFC2822]
   ```

   This URI scheme can be used if the server does not need to request a specific protocol (e.g. because it supports various protocols) or if it does not know which protocols the e-mail client supports. It allows the EMN UA to identify which account to indicate when triggering the e-mail client to act upon the notification. An example `mailbox` attribute (see section 5.2) is given below:

   ```
   mailbox="mailat:user@wapforum.org"
   ```

   The EMN UA MUST support the "mailat" URI scheme.

2. *Any other valid URI scheme*:
   Protocol specific URI schemes may be used instead of the generic URI defined above. For example, the POP [RFC2384] or IMAP [RFC2192] URI schemes may be used. An example `mailbox` attribute (see section 5.2) for POP3 is given below:

   ```
   mailbox="pop://userxyz;auth=3598302@pop.wapforum.org"
   ```

   If the protocol indicated by the URI is supported by the e-mail client, the EMN UA MUST trigger the e-mail client to use that particular protocol to access the e-mail. If not supported, the EMN UA SHOULD discard the EMN.

   Support for these URI schemes is OPTIONAL.

6. Handling of EMN

The e-mail server can utilise the EMN content format when it needs to inform the e-mail client that a message is available for delivery. The EMN message can be sent by the e-mail server to the EMN UA using the WAP PUSH framework [PUSH]. The EMN will then be sent as the message body of a Push Message [PUSHMSG].

The EMN UA is the client-side application that handles the EMN and is subject to the conformance requirements of this specification. It may form an integral part of the E-Mail Client, but may also be a separate application that triggers the E-Mail Client appropriately. The interface between EMN UA and E-Mail Client is an implementation issue, outside the scope of this specification.

In order to designate the EMN UA as the handling application for EMN, a specific `app-id` is conveyed in the `X-Wap-Application-ID` header [PUSHMSG]:

- "`x-wap-application:emn.ua`" if the `absoluteURI` form of the `app-id` syntax is used;
- The hexadecimal value assigned by [OMNA] if the `app-assigned-code` form of the `app-id` syntax is used.

7. Security Considerations

This section is informative.

A mobile device that supports EMN is subject to certain attacks. For example, the POP URI scheme provides a means to include server address, user name and password, allowing the originator of the EMN to potentially direct the user to a fraudulent mailbox. The implementation should provide a means to protect the e-mail client against such security risks. An EMN, or the resource referred by EMN, may be discarded if a chosen security policy is not satisfied. The security policy is implementation specific.

Some possible security measures are:

- The EMN UA providing a means to disable acceptance of the EMN
- The EMN UA discarding any EMN which is not authorized, for example by:
  - Comparing securely provisioned parameters with those supplied in the EMN `mailbox` attribute
  - Relying on authentication of the EMN
- A PPG providing a means to control which Push Initiators are allowed to push EMNs .

8. EMN Reference Information

E-Mail Notification (EMN) is an application of [XML] version 1.0.

8.1 Document Identifiers

8.1.1 SGML Public Identifier

```
-//WAPFORUM//DTD EMN 1.0//EN
```

8.1.2 EMN Media Type

Editor's note: These types are not yet registered with the IANA, and are consequently experimental media types.

Textual form:

```
text/vnd.wap.emn+xml
```

Tokenized form:

```
application/vnd.wap.emn+wbxml
```

8.2 Document Type Definition (DTD)

```
<!--
E-Mail Notification (EMN) Document Type Definition.
EMN is an XML language. Typical usage:
     <?xml version="1.0"?>
     <!DOCTYPE emn PUBLIC "-//OMA//DTD EMN 1.0//EN"
          "http://www.openmobilealliance.com/tech/DTD/emn.dtd">
     <emn
     ...
     />
-->

<!ENTITY % Datetime "CDATA">           <!-- ISO date and time -->
<!ENTITY % URI "CDATA">                <!-- URI designating an
                                       e-mail account -->

<!--==================== The EMN Element ====================-->
<!ELEMENT emn EMPTY>
<!ATTLIST emn
     mailbox              %URI;        #REQUIRED
     timestamp            %Datetime;   #IMPLIED
>
```

9. A compact binary representation of E-Mail Notification

The EMN content format MAY be encoded using a compact binary representation. This content format is based upon the WAP Binary XML Content Format [WBXML].

9.1 Extension Tokens

9.1.1 Tag Tokens

EMN defines a set of single-byte tokens corresponding to the tags defined in the DTD. All of these tokens are defined within code page zero.

9.1.2 Attribute Tokens

EMN defines a set of single-byte tokens corresponding to the attribute names and values defined in the DTD. All of these tokens are defined within code page zero.

9.2 Encoding Semantics

9.2.1 Document Validation

XML document validation (see [XML]) SHOULD occur during the process of tokenising an EMN and, if done, it MUST be based on the DOCTYPE declared in the EMN. When validating the source text, the tokenisation process MUST accept any DOCTYPE or public identifier, if the document is identified as an EMN content format (see section 9.1.2).

The tokenisation process MUST check that the source EMN is XML well-formed, and it SHOULD notify the end-user (in the case of pull) or the push initiator (in the case of push) of any well-formedness or validity errors detected in the source EMN.

9.2.1.1 Validate %Datetime;

The EMN tokenisation process SHOULD validate that attribute values defined as `%Datetime;` follow the syntax defined in section 6.2.

9.2.2 Encoding of %Datetime;

`%Datetime;` data MUST be encoded as OPAQUE data with each number in the string represented by its 4-bit binary value. Any non-numerical characters ("`T`", "`Z`", "`-`", and "`:`") are discarded. Trailing zeros (from right to left) MUST be pair-wise omitted.

For example, "2002-04-16T06:40:00*Z*" is encoded into six octets as follows:

Number	"2"	"0"	"0"	"2"	"0"	"4"	"1"	"6"	"0"	"6"	"4"	"0"	"0"	"0"
Binary value	0010	0000	0000	0010	0000	0100	0001	0110	0000	0110	0100	0000	0000	0000
Octet (hex)	00010000 (20)		00000010 (02)		00000100 (04)		00010110 (16)		00000110 (06)		01000000 (40)		ommitted	

9.3 Numeric Constants

9.3.1 Tag Tokens

The following token codes represent tags in code page zero (0). All numbers are in hexadecimal.

Tag Name	***Token***

Emn	5

9.3.2 Attribute Start Tokens

The following token codes represent the start of an attribute in code page zero (0). All numbers are in hexadecimal. Tokens B0-FF are reserved to be used as attribute value prefixes for the `mailbox` attribute for vendor specific schemes, which must be registered with the OMNA registrar [OMNA].

Attribute Name	***Attribute Value Prefix***	***Token***
timestamp		5
mailbox		6
mailbox	mailat:	7
mailbox	pop://	8
mailbox	imap://	9
mailbox	http://	A
mailbox	http://www.	B
mailbox	https://	C
mailbox	https://www.	D

9.3.3 Attribute Value Tokens

The following token codes represent attribute values in code page zero (0). All numbers are in hexadecimal.

Attribute Value	***Token***
.com	85
.edu	86
.net	87
.org	88

Appendix A. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [IOPProc].

A.1 Client Features

A.1.1 Character Sets and Encoding

Item	Function	Reference	Status	Requirement
EMN-CSE-C-001	Support for UTF-8 encoding	6.1	M	
EMN-CSE-C-002	Support for UTF-16 encoding	6.1	M	
EMN-CSE-C-003	Treat the character encoding of an XML document as defined in [RFC3023]	6.1	M	
EMN-CSE-C-004	Other character encoding	6.1	O	

A.1.2 Content Format and Tokenisation

Item	Function	Reference	Status	Requirement
EMN-CF-C-001	Support for the EMN DTD	6	M	
EMN-CF-C-002	Discard EMNs that do not validate against the EMN DTD	6	M	
EMN-CF-C-003	Support for EMN in textual form (`text/vnd.wap.emn`)	9	M	
EMN-CF-C-004	Support for EMN in tokenised form (`text/vnd.wap.emnc`)	9	M	
EMN-CF-C-005	Support for `%DateTime;` encoded as OPAQUE data	10.2	M	

A.1.3 EMN Semantics

Item	Function	Reference	Status	Requirement
EMN-SEM-C-001	Support for mailat URI scheme	5.3	M	
EMN-SEM-C-002	Support for other URI schemes	5.3	O	EMN-SEM-C-003
EMN-SEM-C-003	Handling of EMNs with a specific URI scheme when the corresponding protocol is supported	5.3	O	
EMN-SEM-C-004	Discarding of EMN with an unsupported URI scheme	5.3	O	
EMN-SEM-C-005	Handling of EMNs that arrive out of order	5.2	O	

A.2 Push Proxy Gateway Features

A.2.1 General

Item	Function	Reference	Status	Requirement
EMN-PPG-S-001	Support for EMN in textual form (`text/vnd.wap.emn`)	9	M	
EMN-PPG-S-002	Support for EMN in tokenised form (`text/vnd.wap.emnc`)	9	M	
EMN-PPG-S-003	Support for the EMN token table	10.3	M	

A.2.2 Validation

Item	Function	Reference	Status	Requirement
EMN-VAL-S-001	XML well-formed	10.2	M	
EMN-VAL-S-002	XML validation	10.2	O	

Appendix B. Examples

This section is informative.

B.1 Use cases

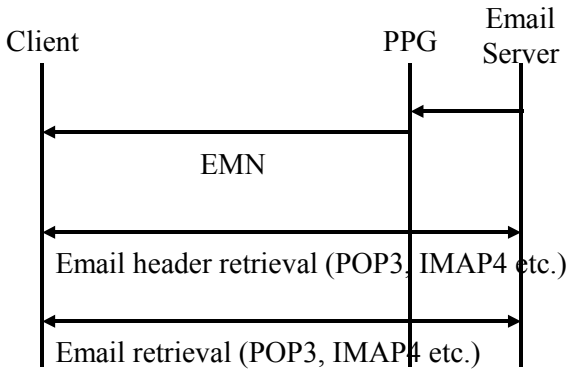
This section describes the use cases for EMN and how a client might behave upon reception of an EMN. It is assumed that client behaviour will depend on its capability (what protocols it supports etc.) and data provisioned on the terminal.

Three-step retrieval process

A typical implementation involving three main steps is depicted below:

- EMN is pushed to the client with minimum information
- When the client receives an EMN, the client determines what information to pull from the email server (e.g., subject, sender).
- The user can select which messages to download based on the header information.

The 3-step process provides security and flexibility by having all client behaviour dictated by the client rather than the server.



Other use cases

Other possible implementations of client behaviour upon reception of the EMN are listed below:

- Retrieve messages, including attachments, without user interaction
- Retrieve messages without attachments, without user interaction. Attachments are kept on the e-mail server for deferred retrieval.
- Retrieve message-header information (sender, subject, date, message size, type of attachment, etc), without user interaction. Message is kept on the server for deferred retrieval.
- Client does not poll for messages, but indicates that email is waiting to be retrieved by displaying an icon (similar to voicemail). The user can manually poll for messages at a later time.
- The user may choose to deactivate the notification feature, but keep the email service active. In such a case, there will be no “mail-is-waiting” icon displayed. The user can manually poll for messages at any time.

B.2 Example

The example below illustrates how the EMN used in section 5 can be designed and tokenised.

```
<?xml version="1.0"?>
     <!DOCTYPE emn PUBLIC "-//OMA/DTD EMN 1.0//EN"
     "http://www.openmobilealliance.com/tech/DTD/emn.dtd">
         <emn
                     mailbox="mailat:user@wapforum.org"
                     timestamp="2002-04-16T06:40:00Z"
         />
```

The tokenised form of the example above (numbers in hexadecimal), using the WBXML encoding defined in section 10, is found below. This example assumes UTF-8 character encoding and NULL terminated strings.

In this example, the textual EMN consists of 190 octets, while the encoded form consists of 32 octets.

```
03   0D   6A   00   85  07  03  'u'  's'  'e'  'r'  '@'  'w'  'a'  'p'  'f'
'o'  'r'  'u'  'm'  00  88  05  C3   06   20   02   04   16   06   40   01
```

In an expanded and annotated form:

Token Stream	***Description***
03	Version number – WBXML version 1.3
0D	EMN 1.0 Public Identifier
6A	Charset = UTF-8 (MIBEnum 106)
00	String table length
85	`emn` with attributes
07	`mailbox="mailat:"`
03	Inline string follows
'u', 's', 'e', 'r', '@', 'w', 'a', 'p', 'f', 'o', 'r', 'u', 'm', 00	string
88	`".org"`
05	`timestamp=`
C3	OPAQUE data follows
06	Length field (6 bytes)
20, 02, 04, 16, 06, 40	Data
01	END (of emn and attribute list)

Appendix C. Change History (Informative)

C.1 Approved Version 1.0 History

Reference	Date	Description
OMA-TS-Push_EMN-V1_0	19 Oct 2007	Status changed to Approved by TP TP Ref# OMA-TP-2007-0416-INP_EMN_V1_0_ERP_for_Final_Approval