WAP Push Architectural Overview

Version 03-Jul-2001

Wireless Application Protocol
WAP-250-PushArchOverview-20010703-a

A list of errata and updates to this document is available from the WAP Forum™ Web site, http://www.wapforum.org/, in the form of SIN documents, which are subject to revision or removal without notice.

© 2001, Wireless Application Protocol Forum, Ltd. All Rights Reserved. Terms and conditions of use are available from the WAP Forum™ Web site (http://www.wapforum.org/what/copyright.htm).

SAMSUNG 1033

© 2001, Wireless Application Protocol Forum, Ltd. All rights reserved.

Terms and conditions of use are available from the WAP Forum™ Web site at http://www.wapforum.org/what/copyright.htm.

You may use this document or any part of the document for internal or educational purposes only, provided you do not modify, edit or take out of context the information in this document in any manner. You may not use this document in any other manner without the prior written permission of the WAP Forum™. The WAP Forum authorises you to copy this document, provided that you retain all copyright and other proprietary notices contained in the original materials on any copies of the materials and that you comply strictly with these terms. This copyright permission does not constitute an endorsement of the products or services offered by you.

The WAP Forum™ assumes no responsibility for errors or omissions in this document. In no event shall the WAP Forum be liable for any special, indirect or consequential damages or any damages whatsoever arising out of or in connection with the use of this information.

WAP Forum™ members have agreed to use reasonable endeavors to disclose in a timely manner to the WAP Forum the existence of all intellectual property rights (IPR's) essential to the present document. The members do not have an obligation to conduct IPR searches. This information is publicly available to members and non-members of the WAP Forum and may be found on the "WAP IPR Declarations" list at http://www.wapforum.org/what/ipr.htm. Essential IPR is available for license on the basis set out in the schedule to the WAP Forum Application Form.

No representations or warranties (whether express or implied) are made by the WAP Forum™ or any WAP Forum member or its affiliates regarding any of the IPR's represented on this list, including but not limited to the accuracy, completeness, validity or relevance of the information or whether or not such rights are essential or non-essential.

This document is available online in PDF format at http://www.wapforum.org/.

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

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

Document History	
WAP-165-PushArchOverview-19991108-a	Approved
WAP-250-PushArchOverview-20010703-a	Current

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

Contents

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

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 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"* [WAPArch].

This document outlines the WAP Push specifications, which together specify a service to push content to mobile devices via the WAP architecture.

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

2. References

2.1. Normative References

None.

2.2. Informative References

[PushCO] "Cache Operation". WAP Forum™. WAP-175-CacheOp-20010731-a. URL:http//www.wapforum.org/

[PushMsg] "Push Message Specification". 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/

[PushPPG] "Push Proxy Gateway Service". WAP Forum™. WAP-249-PPGService-20010713-a. URL:http//www.wapforum.org/

[PushSI] "Service Indication". WAP Forum™. WAP-167-ServiceInd-20010731-a. URL:http//www.wapforum.org/

[PushSL] "Service Loading". WAP Forum™. WAP-168-ServiceLoad-20010731-a. URL:http//www.wapforum.org/

[RFC1951] "DEFLATE Compressed Data Format Specification version 1.3". P. Deutsch. May 1996. http://www.ietf.org/rfc/rfc1951.txt

[RFC2387] "The MIME Multipart/related content type". E. Levinson. August 1998. URL:http://www.ietf.org/rfc/rfc2387.txt

[RFC2616] "Hypertext Transfer Protocol – HTTP/1.1". R. Fielding et al. June 1999. URL:http://www.ietf.org/rfc/rfc2616.txt

[RFC2617] "HTTP Authentication: Basic and Digest Access Authentication". J. Franks et al. June 1999. URL:http://www.ietf.org/rfc/rfc2617.txt

[UAProf] "WAP UAProf". WAP Forum™. WAP-248-UAProf-20010530-a. URL:http//www.wapforum.org/

[WAE] "Wireless Application Environment Specification". WAP Forum™. WAP-236-WAESpec-20010629-a. URL:http//www.wapforum.org/

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

[WAPTLS] "WAP TLS Profile and Tunnelling". WAP Forum™. WAP-219-TLS-20010411-a. URL:http//www.wapforum.org/

[WBXML] "Wireless Binary XML". WAP Forum™. WAP-192-WBXML-20010725-a. URL:http//www.wapforum.org/

[WINA] "WAP Interim Naming Authority". WAP Forum™. URL: http://www.wapforum.org/wina

[WSP] "Wireless Session Protocol". WAP Forum™. WAP-230-WSP-20010705-a. URL: http://www.wapforum.org/

[WTLS] "Wireless Transport Layer Security Protocol". WAP Forum™. WAP-261-WTLS-20010406-a. URL: http://www.wapforum.org/

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

[XML] "Extensible Markup Language (XML) 1.0 (Second Edition)". T. Bray, et al, October 2000. URL: http://www.w3.org/TR/REC-xml

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

3. Terminology and Conventions

3.1. Conventions

This is an informative document, which is not intended to provide testable requirements to implementations.

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

Application-Level Addressing - the ability to address push content between a particular user agent on a WAP client and push initiator on a server.

Bearer Network - a network used to carry the messages of a transport-layer protocol between physical devices. Multiple bearer networks may be used over the life of a single push session.

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".

Contact Point - address information that describes how to reach a push proxy gateway, including transport protocol addres and port of the push proxy gateway.

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"

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

Media Type - a class of information distinguished by its presentation format and/or interchange format. Examples include images, plain text, sounds and video.

Multicast Message - a push message containing a single OTA client address which implicitly specifies more than OTA client address.

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 Framework - the entire WAP push system. The push framework encompasses the protocols, service interfaces, and software entities that provide the means to push data to user agents in the WAP client.

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 protocol 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.

Push Session - A WSP session that is capable of conducting push operations.

Server - a device (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).

Terminal - see "client".

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

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.

XML - see *Extensible Markup Language*

3.3. Abbreviations

CPI	Capability and Preference Information
HTTP	Hypertext Transfer Protocol
IANA	Internet Assigned Numbers Authority
IP	Internet Protocol
MIME	Multipurpose Internet Mail Extensions
MMS	Multimedia Messaging Service
MSISDN	Mobile Station International Subscriber Directory Number
OTA	Over The Air
OTA-HTTP	(Push) OTA over HTTP
OTA-WSP	(Push) OTA over WSP
PAP	Push Access Protocol
PI	Push Initiator
PO-TCP	PPG Originated TCP connection establishment method
PPG	Push Proxy Gateway
QoS	Quality of Service
RFC	Request For Comments
SGML	Standard Generalized Markup Language
SI	Service Indication
SIA	Session Initiation Application
SIR	Session Initiation Request
SL	Service Loading
SMS	Short Message Service
SSL	Secure Socket Layer
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TO-TCP	Terminal Originated TCP connection establishment method
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WAP	Wireless Application Protocol
WBXML	WAP Binary XML
WINA	WAP Interim Naming Authority
WSP	Wireless Session Protocol
WTA	Wireless Telephony Applications
WTLS	Wireless Transport Layer Security
XML	Extensible Mark-up Language

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

4. Introduction

In the "normal" client/server model, a *client* requests a service or information from a *server*, which then responds in transmitting information to the client. This is known as “pull” technology: the client “pulls” information from the server. Browsing the World Wide Web is a typical example of pull technology, where a user enters a URL (the request) that is sent to a server, and the server answers by sending a Web page (the response) to the user.

In contrast to this, there is also “push” technology, which is also based on the client/server model, but where there is no explicit request from the client before the server transmits its content. The WAP Push framework introduces a means to transmit information to a device without a user request.

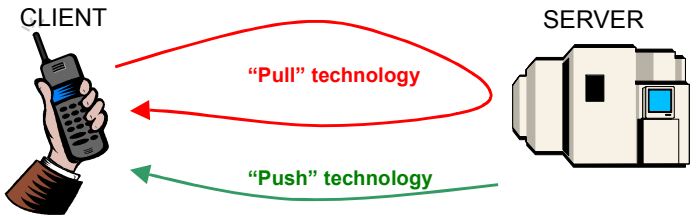

Figure 1 - Comparison of pull vs. push technology

Another way to say this is that whereas “pull” transactions of information are always initiated from the client, “push” transactions are server-initiated.

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

5. The Push Framework

A push operation in WAP is accomplished by allowing a *Push Initiator* (PI) to transmit *push content* and *delivery instructions* to a *Push Proxy Gateway* (PPG), which then delivers the push content to the WAP client (henceforth referred to as "client" or "terminal") according to the delivery instructions.

The PI is typically an application that runs on an ordinary web server. It communicates with the PPG using the *Push Access Protocol* (PAP). The PPG uses the *Push Over-The-Air* (OTA) *Protocol* to deliver the push content to the client. Figure 2 illustrates the push framework:

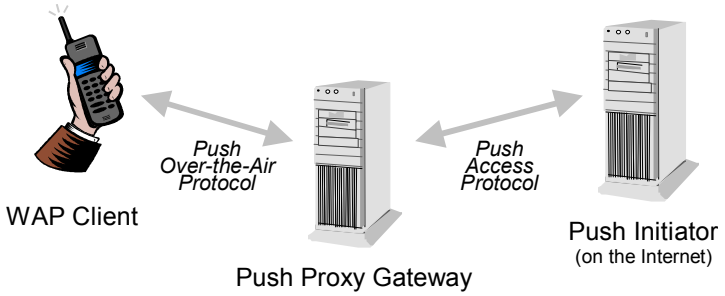

Figure 2: The Push Framework

PAP is based on standard Internet protocols; XML is used to express the delivery instructions, and the push content can be any MIME media type. These standards help make WAP Push flexible and extensible.

As mentioned, the **PPG** is responsible for delivering the push content to the client. In doing so it potentially may need to translate the client address provided by the PI into a format understood by the mobile network, transform the push content to adapt it to the client's capabilities, store the content if the client is currently unavailable, etc. The PPG does more than deliver messages. For example, it may notify the PI about the final outcome of a push submission and optionally handle cancellation, replace, or client capability requests from the PI.

The **OTA** protocol provides both connectionless and connection-oriented services. While the (mandatory) connectionless service relies upon Wireless Session Protocol (WSP), the (optional) connection-oriented service may be provided in conjunction with both WSP (OTA-WSP) and HTTP (OTA-HTTP). An important part of the OTA protocol is the Session Initiation Application (SIA), which is further described in section 8.3.

Figure 2 illustrates the PI and the PPG as separate entities, which likely will be the most common configuration. It shall however be noted that the PI and the PPG may be co-located. The latter could, for example, be feasible for PPG operator services, large service providers, or when transport level end-to-end security is needed.

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

6. The Push Proxy Gateway

The Push Proxy Gateway (*PPG*) is the entity that does most of the work in the Push framework. Its responsibilities include acting as an access point for content pushes from the Internet to the mobile network, and everything associated therewith (authentication, address resolution, etc).

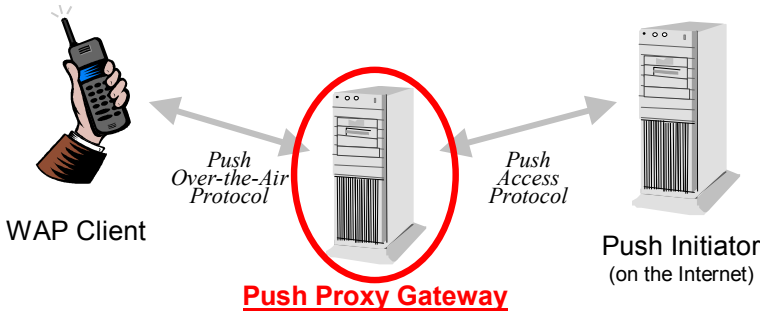

Figure 3 - PPG highlighted

As the PPG is the entry point to a mobile network, it may implement network access-control policies about who is able to gain access to the network, i.e. who is able to push content and who is not, and under which circumstances, etc.

It should be noted that both PPG (push) and WAP proxy [WAP] (pull) functionality may be built into a single proxy.

6.1. Services Overview

The PPG provides the Push framework with several services. It is the entry point for content pushed from outside the wireless network (e.g. the Internet) destined for the WAP client. The PPG may perform the following:

- PI identification and authentication; access control
- Parsing of and error detection in push content and control information
- Client discovery services (including client capabilities)
- Address resolution of push recipient
- Binary encoding and compilation of certain content types, or general compression, to improve efficiency OTA
- Protocol conversion

6.2. Message Acceptance and Rejection

The PPG accepts content from the PI using the Push Access Protocol (see section 7). This content is divided into several parts using a multipart/related content type where the first part contains control information for the PPG itself. Such information includes recipient address(es), delivery time constraints, Quality of Service (QoS) information, notification requests, etc.

The PPG will acknowledge successful (or report unsuccessful) parsing of the control information, and may in addition report debug information about the push content itself.

6.3. Message Handling Service

Once the content has been accepted for delivery, the PPG attempts to find the correct client and deliver the content to that client using the Push Over-The-Air protocol (see section 8). The PPG may attempt to deliver the content until a timeout expires. This timeout may be set by the PI and/or policies of the PPG operator.

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

The PPG may also send a notification to the PI when the final status of the push submission (delivered, cancelled, expired, etc.) has been reached, if the PI so requests. Hence, the service is asynchronous from the PI's point of view (the PI need not wait on-line for the PPG to complete its delivery).

6.4. Encoding, Compilation, and Compression

The PPG may encode WAP content types (e.g. WML 1.2 and SI) into their binary counterparts [WBXML]. This textual-to-binary translation would take place before delivery over-the-air. Other content types, unknown to the PPG, may be forwarded as received unless it is known that the client does not support them.

When OTA-HTTP (see section 8.2) is used, the PPG may use deflate encoding [RFC1951] (i.e. general compression) to obtain better over-the-air efficiency also for content types that do not have a binary counterpart (e.g. WML 2.0).

6.5. Multicast, Broadcast and Aliasing Considerations

The PPG may implement address aliasing schemes to enable multicast and broadcast services, where special addresses may translate into such operations. These are implementation dependent.

6.6. Client Capabilities Query Service

A PI may query the PPG for a specific client's capabilities and preferences to aid in creating better-formatted content. The capability and preference information returned by the PPG is formatted as a UAProf document [UAProf]. This feature is optional in the PPG.

6.7. Reference

For more information, see [PushPPG].

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

7. The Push Access Protocol

The Push Access Protocol (*PAP*) is the means by which a PI pushes content to a mobile network, addressing its PPG.

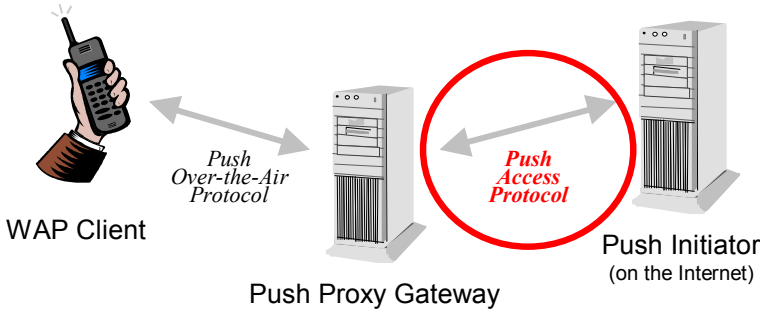

Figure 4 - PAP highlighted

PAP was designed to be independent of the underlying transport; it can be transported over virtually any protocol that allows MIME types to be transported over the Internet. HTTP is the first protocol to be specified as a transport protocol for PAP, other protocols (e.g. SMTP) may be added in the future.

7.1. General Structure

As mentioned in previous sections, PAP is used to carry push related control information that is used by the PPG. This information is expressed using XML [XML]. When, for example, a new message is submitted to the PPG the control information and the push content are carried in a MIME multipart/related [RFC2387] entity. This implies that a single (MIME) entity is conveyed independent of the type of operation.

7.2. PAP Operations

The PAP supports the following operations:

- Push Submission (PI to PPG)
- Result Notification (PPG to PI)
- Push Cancellation (PI to PPG)
- Push Replacement (PI to PPG)
- Status Query (PI to PPG)
- Client Capabilities Query (PI to PPG)

7.2.1. Push Submission

The Push message contains three entities: a control entity, a content entity, and optionally a capability entity. These are bundled together in a multipart/related message, which is sent from the PI to the PPG.

The control entity is an XML document that contains delivery instructions destined for the PPG, whereas the content entity is destined for the client. The PPG may or may not convert the content entity into a more bandwidth-optimised form before forwarding it over-the-air (see section 6.4).

The optional capability entity contains the client capabilities that the message was formatted according to, in UAPROF [UAProf] format. The PI may include this entity to indicate what it *assumes* the client capabilities are. The PPG may reject the message if the assumed capabilities do not match those known by the PPG.

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

7.2.2. Result Notification

If the PI has requested information about the final outcome of the delivery, the Result Notification message is transmitted from the PPG to the URI specified by the PI. It is an XML entity indicating successful or unsuccessful (client not reachable, timeout, etc.) delivery.

One key feature of the Push Framework is the possibility for a PI to rely on the response from the PPG; a confirmed push is confirmed by the WAP device when (and only when) the target application has taken responsibility for the pushed content. If it cannot take that responsibility, it must abort the operation, and the PI will know that the content never reached its destination.

7.2.3. Push Cancellation

This is an XML entity transmitted from the PI to the PPG, requesting cancellation of a previously submitted message. The PPG responds with an XML entity indicating if the cancellation operation was successful.

7.2.4. Push Replacement

The push submission operation described in section 7.2.1 may – if the PI so requests – cause a previously submitted message to be replaced. It is possible to specify if the new message only should be sent to those recipients whom have not received the original message, or if the new message should be sent to all recipients. In either case, the original message is cancelled for those recipients to whom the original message has not been delivered.

7.2.5. Status Query

This is an XML entity transmitted from the PI to the PPG, requesting the status of a previously submitted message. The PPG responds with an XML entity containing the current status.

7.2.6. Client Capabilities Query

This is an XML entity transmitted from the PI to the PPG, requesting the capabilities of a particular device on the network (see also *Client Capabilities Query*, section 6.6). The PPG responds with a multipart/related entity containing two parts, where the first part contains the result of the request, and the second part contains the capabilities of the device formatted according to the WAP User Agent Profile vocabulary [UAProf].

7.3. HTTP Transport

When HTTP is used a transport protocol for PAP, the HTTP POST request method and its response are used to transport the information. The HTTP response always contains result code 202 (“accepted for processing”) when the HTTP transaction itself succeeds; the response PAP document may contain a PAP error though. See [RFC2616] for more information about HTTP/1.1.

7.4. Reference

For more information, see [PushPAP].

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

8. The Push Over-The-Air Protocol

The Push Over-The-Air (OTA) protocol is the part of the Push Framework that is responsible for transporting content from the PPG to the client and its user agents. It is designed to run on top of HTTP (OTA-HTTP) or WSP (OTA-WSP). The parts of OTA-WSP pertaining to connectionless push must always be implemented in both PPGs and clients. Connection-oriented push, using either OTA-HTTP or OTA-WSP, is optional.

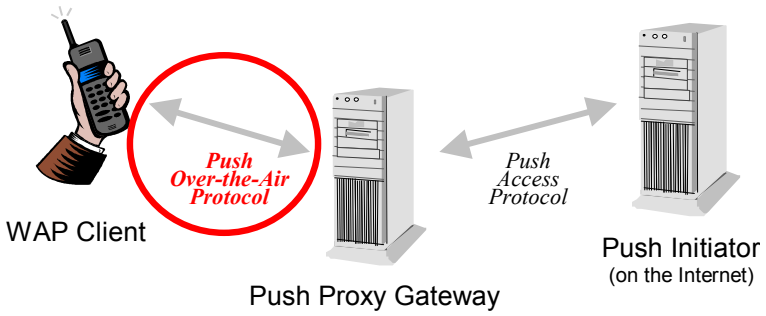

Figure 5 - OTA highlighted

8.1. OTA-WSP

The OTA-WSP protocol variant is architecturally a thin protocol layer on top of WSP, and may hence be used with any bearer addressed by WAP. OTA-WSP utilises the features provided by WSP (see [WSP] for details), and extends those to address push specific needs; basically by introducing new service primitives and extending existing ones with new header fields. For example, OTA-WSP relies upon WSP's capability negotiation feature (possibly using UAProf), and it provides both connectionless (unconfirmed push) and connection-oriented (unconfirmed and confirmed push) services.

8.2. OTA-HTTP

This protocol variant utilises HTTP for over-the-air communication between the PPG and the client and is hence primarily to be used in conjunction with IP bearers. The HTTP variant provides only connection-oriented push. Push content is delivered using the HTTP POST method, implying that the PPG acts as an HTTP client and the client (i.e. the mobile device) as an HTTP server. To avoid confusion the client is therefore referred to as "terminal" in [PushOTA] when OTA-HTTP is discussed.

The [PushOTA] specification defines two methods for establishing an active TCP connection (i.e. a TCP connection to be used for push delivery). The methods are *PPG Originated TCP* (PO-TCP) and *Terminal Originated TCP* (TO-TCP). PO-TCP allows an active TCP connection to be established when the bearer is active (or can be activated by the PPG) and the terminal's IP address is known by the PPG. The TO-TCP method addresses other cases, and is usually used in combination with the Session Initiation Request (SIR) mechanism (see section 8.3).

By using the concept of (long-lived) sessions, OTA-WSP provides a means for the client to convey its capabilities to the PPG. In OTA-HTTP the terminal registers with a PPG to provide similar functionality. This is accomplished by the PPG by sending an HTTP OPTIONS request to the terminal, whereby the terminal includes its Capability and Preference Information (CPI) in the response (optionally using UAProf). A mechanism to avoid the information being sent when it is known by the PPG is provided to improve over-the-air efficiency.

OTA-HTTP also provides a means for identifying and optionally authenticating both the terminal and the PPG. The authentication schemes "basic" and "digest", as specified in [RFC2617], are used to authenticate the PPG to the terminal, while a slightly modified variant is used to authenticate the terminal to the PPG (this since RFC2617 only specifies how an HTTP client, in this case the PPG, is authenticated).

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

8.3. Session Initiation Application

No matter if OTA-HTTP or OTA-WSP is used, it is often necessary for the client to initiate the communication for reasons explained below. The Session Initiation Application (SIA), which is a client-side application, has been specified for this purpose. The SIA listens to Session Initiation Requests (SIR) from the PPG, and responds by activating the appropriate bearer and contacting the desired PPG.

When OTA-WSP is used, it is always the client that takes the initiative to establish the underlying WSP session. An SIR is sent from the PPG to the client when it wishes to create a WSP session for push purposes. Upon reception of the SIR, the client activates the bearer indicated in the SIR and establishes a WSP session towards the indicated PPG over that bearer.

The SIR mechanism is also used in conjunction with OTA-HTTP, in particular when the client's IP address is not known by the PPG, and/or when the PPG cannot activate the desired bearer. In that case the SIR instructs the client to activate a specific bearer and establish an active TCP connection towards the PPG specified in the SIR (using the TO-TCP method).

The SIR is typically sent to the client using connectionless push (provided by OTA-WSP) independent of whether the client will use OTA-WSP or OTA-HTTP when it subsequently contacts the PPG. Attention has been paid to ensure that the SIR is compact enough to fit into a single SMS in the normal case. SMS is available in most current mobile networks, provides a means to use a well-known client address (MSISDN) and provides transport level reliability (i.e. provides good reliability also when connectionless push is used).

8.4. Reference

For more information, see [PushOTA].

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

9. Push Specific Media Types

The WAP push framework allows any MIME media type to be delivered between the PI and the client. The media types described in this section have been created to add capabilities not already provided by existing MIME types. Other media types with push specific semantics have been defined by the WAP Forum to meet the needs for specific applications (e.g. MMS, Provisioning, and WTA).

Note: If push specific semantics are neither defined for the media type itself, nor for the user agent receiving a certain media type, such media types are placed in the cache memory or discarded when received via push (this applies to e.g. WML). For more information, see [WAE].

9.1. Service Indication

The *Service Indication* (SI) media type provides the ability to send notifications to end-users in an asynchronous manner. Such notifications may, for example, be about new e-mails, changes in stock price, news headlines, advertising, reminders of e.g. low prepaid balance, etc.

In its most basic form, an SI contains a short message and a URI indicating a service. The message is presented to the end-user upon reception, and the user is given the choice to either start the service indicated by the URI immediately, or postpone the SI for later handling. If the SI is postponed, the client stores it and the end-user is given the possibility to act upon it at a later point of time.

In addition to the basic functionality described above, SI allows the PI to control the following:

- The level of user-intrusiveness (assign an SI a certain priority)
- Replacement (replacement of an older SI with a new one upon reception)
- Deletion (delete an already received SI by sending a "delete" SI)
- Expiration (assign an expiration time to an SI after which it will be expired)

SI is the only media type among those described in this section that is mandatory in clients implementing push.

9.1.1. Reference

For more information, see [PushSI].

9.2. Service Loading

In contrast to SI, *Service Loading* (SL) does not imply any user involvement. An SL conveys an URI that points to some content that is loaded by the client without end-user confirmation, and an instruction whether the content should be executed/rendered or placed in the cache memory. If the content should be executed/rendered, the PI can control the level of user-intrusiveness.

9.2.1. Reference

For more information, see [PushSL].

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

9.3. Cache Operation

The *Cache Operation* (CO) media type provides a means for invalidating specific objects, or all objects sharing the same URI prefix, stored in the client's cache memory. This feature is useful in situations when the cached content's expiration time cannot be determined beforehand (e.g. a view of a mailbox) and the content changes (e.g. new emails arrive) more often than the user accesses it.

9.3.1. Reference

For more information, see [PushCO].

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

10. Addressing

WAP Push addressing occurs on client and application levels. In addition, two registered ports (secure and non-secure) are used on the client for connectionless push. When connection-oriented OTA-WSP is used, any WSP session with the push capability set can be used. OTA-HTTP, which is connection-oriented only, uses the concept of active TCP connections, which are dedicated for push specifically. More details about ports, sessions, and active TCP connections can be found in [PushOTA].

10.1. Client addressing

The PI uses the client address to instruct the PPG which client the pushed message is intended for. The [PushPPG] specification introduces an addressing scheme that allows:

- **User-defined identifiers**
  An arbitrary text string (e.g. an email address) is used to identify the client. The PPG is responsible for translating the string into an address format understood by the mobile network.

  Examples from [PushPPG]:

  ```
  WAPPUSH=john.doe%40wapforum.org/TYPE=USER@ppg.carrier.com
  ; user-defined identifier for john.doe@wapforum.org
  WAPPUSH=+155519990730/TYPE=USER@ppg.carrier.com
  ; user-defined identifier that looks like a phone number
  ```

- **Device addresses**
  An address understood by the mobile network, e.g. MSISDN (SMS etc.) or IP address (GPRS etc.).

  Examples from [PushPPG]:

  ```
  WAPPUSH=+155519990730/TYPE=PLMN@ppg.carrier.com
  ; device address for a phone number of some wireless network
  WAPPUSH=195.153.199.30/TYPE=IPv4@ppg.carrier.com
  ; device address for an IP v4 address
  ```

The `TYPE` switch indicates the type of address (user-defined or device address including type of address), and the `ppg.carrier.com` part is the Internet host name of the PPG. For more information, see [PushPPG].

10.2. Application-Level Addressing

Pushed content always targets a specific user agent (or more general, a specific application) on the device. An application identifier, which is a URI or a numeric value registered with [WINA], identifies a user agent. The PI includes the application identifier in a push message by including the `X-Wap-Application-Id` header defined in [PushMsg]. This header is also conveyed to the client when the PPG delivers the message, allowing the client to dispatch the message to the intended user agent.

10.2.1. OTA Efficiency and Numeric Identifiers

To improve over-the-air efficiency, a numeric identifier may be used instead of a URI. WINA [WINA] has assigned numbers to well-known user agents such as WAE and WTA, to avoid the overhead of sending a URI.

If a PPG is requested to push content with an application identifier URI that it recognises as a URI that has a numeric identifier assigned by WINA, the URI is replaced with the numeric identifier.

The PI may itself use a numeric identifier when the push message is submitted to the PPG, possibly an identifier that is not registered. The latter is discouraged with deployed applications because of the possibility of collisions. It is mainly intended for experimental user agents that have not yet been publicly deployed.

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

10.3. Example

Let's assume a PI has submitted a message intended for client Foo, for an application called Bar, to a PPG serving client Foo. In addition, the PI has requested that the message should be delivered in a confirmed manner (implying connection-oriented delivery). The PPG (supporting both OTA-HTTP and OTA-WSP) has not communicated with the client before, so it does not know if it supports OTA-HTTP or OTA- WSP (or both). There's currently no push session or active TCP connection between the PPG and the client called Foo, so either needs to be established.

The PPG sends an SIR to Foo in a connectionless manner using e.g. SMS, indicating that it wants to push some content to application Bar. Since the PPG does not know if the client supports OTA-HTTP or OTA-WSP it includes PPG contact points for both variants in the SIR. This request is sent to the SIA at Foo just like any other push content (i.e. by targeting one of the ports dedicated for connectionless push and including the SIA application identifier). The client receives the content, sees it's for the SIA, and sends it onward. The SIA, on receipt of this request, checks if the target application is installed in Foo and possibly that the user preferences indicate that the target application accepts pushed content. It notes that application Bar is, in fact, installed on this client, so the client acts upon the SIR. Lets assume that the client supports only OTA-WSP, implying that a session should be established towards the PPG.

Now, the owner of this particular device does not want to expose what applications he has installed in the device to anybody (privacy issue). The SIA notes this and sets up a push session with the PPG, indicating that the session accepts content for any application; if the user had been less paranoid, applications for which this session could be used would have been explicitly listed instead.

Once the session has been established, the PPG performs the confirmed push over that session, and the client gets the push content originally submitted by the PI. The client gets the content, sees that it is for application Bar, and passes it to this application. When (and only when) the Bar application takes responsibility for the push content, the push is confirmed all the way back to the PI (if so requested).

10.4. Reference

For more information, see [PushPPG] and [PushOTA].

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

11. Security Considerations

When implementing WAP Push, security and trust models come into consideration in several areas. These are examples of questions that may arise:

- How can a PI be authenticated?
- What role could the PPG play in the security and trust model?
- What are the access control policies for a PI and pushed content?
- How can a client authenticate something if it has no certificate?

Regardless of these issues, it should be kept in mind that the Push Framework is capable of providing the client with enough information to have a trust model and security policy of its own.

11.1. Authenticating a Push Initiator

It is important that a PI is authenticated in one form or another, depending on the security environments in which the PI and PPG are operating. This is an attempt to list some of the possible solutions.

- **Use of Session-level Certificates (TLS, SSL)**
  If the network between the PI and PPG is not trusted (e.g., the Internet, a very large intranet, etc.), TLS/SSL can be used between the PI and the PPG.

- **HTTP Authentication**
  Even though the most common form of HTTP authentication is the basic authentication (i.e., a user-id/password pair), other forms of HTTP authentication (e.g., digest) might be preferable. The major difference between this approach and the use of TLS/SSL is that the latter is stronger in scalability and confidence, while the former is weaker in these aspects.

- **A Combination of Technologies**
  Technologies could be combined. For example, the PI can establish an anonymous TLS/SSL session with a PPG, whereupon HTTP authentication could be used to authenticate the PI.

- **Trusted Network**
  In some real world installations, the network between the PI and the PPG is a private network. Therefore, the PI is implicitly trusted in such installations.

11.2. Client Delegation of PI Authentication

"Delegation of Authentication" refers to the principle that authentication can be transitive. If a client and a PPG can establish trust, the PPG can authenticate a PI on behalf of the client.

For example, after a client has used the means provided by [WTLS] or [WAPTLS] to authenticate a PPG, the client could look in its list of trusted PPG's. If the PPG is listed as trusted, the client can trust the PPG, and hence also that the PI is correctly identified. Using the methods described in the previous section, a PPG can authenticate a PI with various levels of confidence. If it does, the OTA protocol makes it possible for the PPG to indicate that the PI is authenticated in the message that is delivered to the client.

11.3. Possible PPG Filtering and Access Control

The PPG can perform filtering and access control to discard pushed content that originates from a non-trusted or unauthorised PI. Such a feature is left to the discretion of the PPG implementer and the business relationship between the WAP service subscriber and the PPG operator.

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

12. Scope of the Push Specifications

- **WAP Push Architectural Overview**
  The purpose of this document is to serve as a starting point for anybody wanting to know more about the WAP Push technology, before taking on the other specifications.

- **Push Access Protocol Specification**
  This document specifies the protocol with which a Push Initiator communicates with the PPG. See section 7 for a brief description.

- **Push Proxy Gateway Service Specification**
  This document specifies the Push Proxy Gateway functionality, and how it interacts with the Push Access Protocol and the Push Over-The-Air protocol. See section 6 for a brief description.

- **Push OTA Protocol Specification**
  This document specifies the protocol with which a PPG communicates with a push-capable client. See section 8 for a brief description.

- **Push Message Specification**
  This document specifies end-to-end properties of a push message.

- **Service Indication Specification**
  This document specifies a content type used for notifying users they have new information waiting on a server. See section 9 for a brief description.

- **Service Loading Specification**
  This document specifies a content type that instructs the client to automatically load a URI. See section 9 for a brief description.

- **Cache Operation Specification**
  This document specifies a content type that instructs the client to invalidate cached resources. See section 9 for a brief description.

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved

Appendix A. Change History

Type of Change	Date	Section	Description
WAP-250-PushArchOverview-20010703-a	03-Jul-2001		The initial version of this document.

© 2001, Wireless Application Protocol Forum, Ltd.
All rights reserved