Push Proxy Gateway Service

Approved Version 2.2 – 09 Aug 2011

Open Mobile Alliance

OMA-TS- PPGService –V2_2-20110809-A

Contents

Figures

1. Scope

A part of the WAP effort is the specification of a push architecture, illustrated in figure 1, which allows content to be pushed from wired networks to push compliant mobile devices. The scope of this document is the specification of the Push Proxy Gateway, a gateway intended to provide push connectivity between wired and wireless networks.

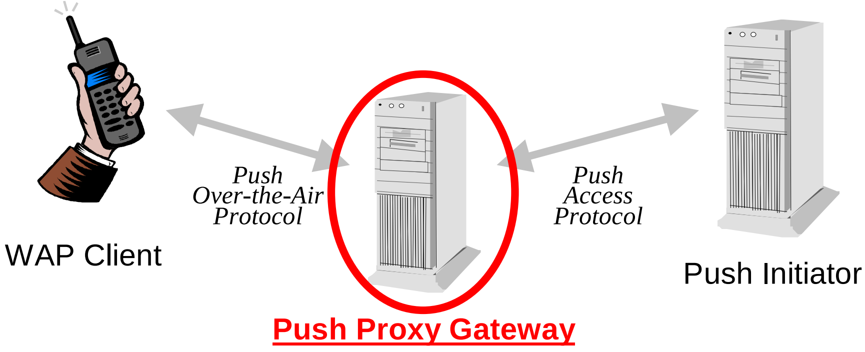

Figure 1 - Push Architecture

2. References

2.1 Normative References

[Mobitex] “Mobitex Interface Specification (MIS) ”. Rev R4A. Document number LZY 232 105. Ericsson.
URL://www.ericsson.com/

[PushMsg] "Push Message Specification". Open Mobile Alliance™. OMA-TS-Push_Message-V2_2.
URL:http://www.openmobilealliance.org/

[PushOTA] "Push OTA Protocol Specification". Open Mobile Alliance™. OMA-TS-PushOTA-V2_2.
URL:http://www.openmobilealliance.org/

[PushPAP] "Push Access Protocol Specification". Open Mobile Alliance™. OMA-TS-PAP-V2_2
URL:http://www.openmobilealliance.org/

[RFC791] “Internet Protocol”. J. Postel. September 1981. URL: http://www.ietf.org/rfc/rfc791.txt

[RFC822] “Standard for the Format of ARPA Internet Text Messages”. David H. Crocker. August 1982.
URL: http://www.ietf.org/rfc/rfc822.txt

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

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

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

[RFC3513] “IP Version 6 Addressing Architecture”. R. Hinden, et al. July 1998.
URL: http://www.ietf.org/rfc/rfc3513.txt

[RFC4234] “Augmented BNF for Syntax Specifications: ABNF”. D. Crocker, Ed., P. Overell. October 2005,
URL:http://www.ietf.org/rfc/rfc4234.txt

[SCRRULES] “SCR Rules and Procedures”, Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures,
URL:http://www.openmobilealliance.org/

[WBXML] “Binary XML Content Format Specification”. WAP Forum™. WAP-192-WBXML
URL:http://www.openmobilealliance.org/

[WDP] “Wireless Datagram Protocol”. WAP Forum™. WAP-259-WDP. URL:http://www.openmobilealliance.org/

[WSP] “Wireless Session Protocol”. WAP Forum™. WAP-230-WSP705 URL:http://www.openmobilealliance.org/

2.2 Informative References

[OMNA] “OMA Naming Authority”. Open Mobile Alliance™.
URL:http://www.openmobilealliance.org/tech/OMNA.aspx

[PushArch] "Push Architectural Overview". Open Mobile Alliance™. OMA-AD-Push-V2_2
URL:http://www.openmobilealliance.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.

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

Within this document, `courier font` is used to identify literal names of elements, attributes, parameters, and values in referenced specifications. For example, the following table indicates that the [PushPAP] `message-state` attribute contains the `"pending"` value.

PAP Attribute	Value
`message-state`	`"pending"`

3.2 Definitions

Application A value-added data service provided to a 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 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, 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 address 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 be 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"

Multicast Message a push message containing a single address which implicitly specifies more than one 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 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 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.

Registration Context a state where the PPG is aware of at least the last capabilities and preferences conveyed from the terminal.

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).

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.

3.3 Abbreviations

ABNF	Augmented Backus-Naur Form
DTD	Document Type Definition
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
MAN	Mobitex Subscription Number
OMNA	Open Mobile Alliance Naming Authority
OTA	Over The Air
OTA-HTTP	(Push) OTA over HTTP
OTA-WSP	(Push) OTA over WSP
OTA-SIP	(Push) OTA over SIP
PAP	Push Access Protocol
PI	Push Initiator
PPG	Push Proxy Gateway
QoS	Quality of Service
RFC	Request For Comments
SIP	Session Initiation Protocol
SIR	Session Initiation Request
URI	Uniform Resource Identifier
URI	Uniform Resource Locator
WAP	Wireless Application Protocol
WDP	Wireless Datagram Protocol
WSP	Wireless Session Protocol
WBXML	WAP Binary XML
XML	Extensible Mark-up Language

4. Introduction

This document is part of the Push specification suite. These specifications address the needs of a content provider seeking to "push" (i.e., send without a synchronous request) content to a client (i.e., a push-compliant mobile device). This is in contrast to "pull" technology, which requires a synchronous request from the client.

Push to a client is facilitated by a gateway between the wired and wireless networks. This gateway is called the Push Proxy Gateway (PPG). The purpose of this document is to specify the function of PPG.

In addition to the PPG, the push architecture provides protocols to push content to the gateway and on to the client, additional functionality within clients, new addressing schemes, and several standard message and content types. These are outside the scope of this document. For a complete overview, see [PushArch].

4.1 Version 1.0

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. 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.

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 relies upon Wireless Session Protocol (WSP).

Figure 1 illustrates the push framework:

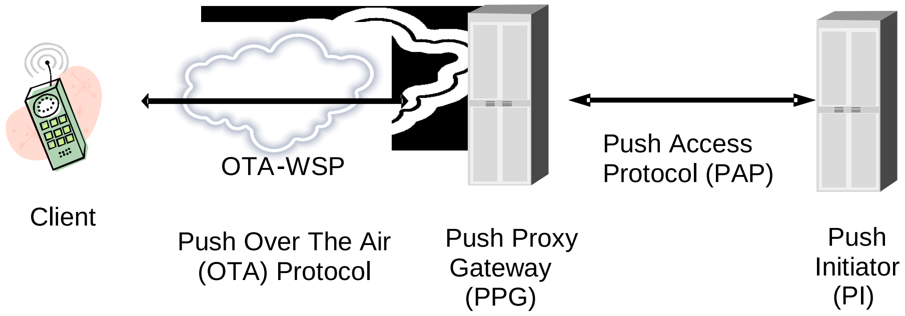

Figure 2 - The Version 1.0 Push Framework

4.2 Version 2.0

WSP is suitable for use with low-bandwidth bearers that do not support TCP/IP, e.g. SMS, but can be used on top of TCP/IP as well. A new OTA (OTA-HTTP) variant is designed to run on top of HTTP 1.1 and is intended to be used in conjunction with bearers that support TCP/IP, e.g. GPRS, WIMAX or WLAN.

Figure 3 illustrates the enhanced push framework:

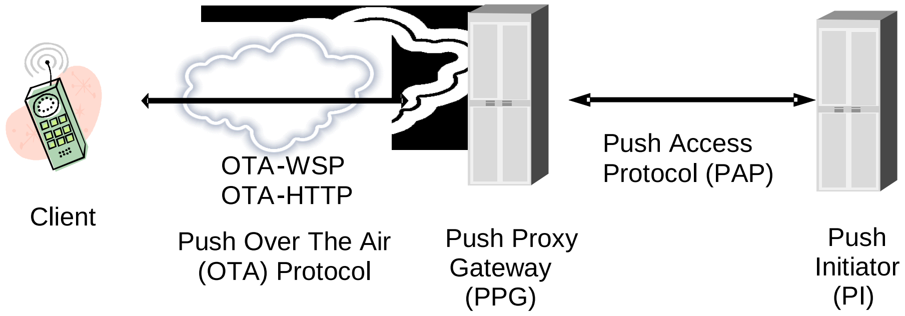

Figure 3 - The Version 2.0 Push Framework

4.3 Version 2.1

The quality-of-service Element is extended to allow one-shot delivery and confirmed push with response. In the one-shot delivery service the PPG must attempt to deliver the message only once, and ensure that a one-shot delivery attempt can be made on the underlying bearer (e.g. not use the store-and-forward and retry capabilities provided by an SMSC). In the confirmed push with response service, the PI informs the PPG if it is ready to receive content from the client in response of the push request. The content is transferred to the PI in the corresponding resultnotification message.

4.4 Version 2.2

In order to protect against denial of service attacks and push from unauthorized sources a Push Whitelist mechanism is defined. The Push Whitelist consists of a list of trusted PPGs and a list of trusted SMSCs. In addition to authenticating the source of the push message from a network perspective (either a PPG or intermediate SME / SMSC) a secondary mechanism may be used to verify the originating source of the content (either PI or PPG). The content is trusted due to it being ‘signed’ using a shared secret. This shared secret may be user defined or it might be some specific information that is related to the bearer or network.

A third Over The Air protocol is added to allow push messages be send over SIP in a convergent or IP only network.

Figure 4 illustrates the enhanced push framework:

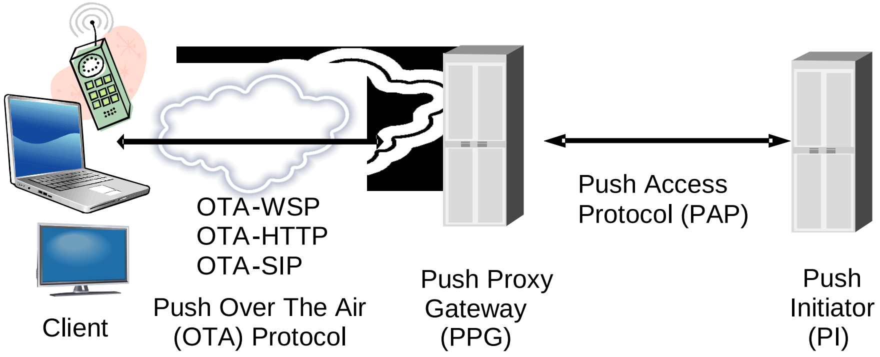

Figure 4 - The Version 2.2 Push Framework

5. PPG Operations

This section defines the operations performed by a PPG. These operations include push submission processing, result notification, delivery cancellation, and Push Access Protocol (PAP) status query.

PPG operations are defined as handling each push submission (and subsequent operations related to its push message) independently from other push submissions. However, there may be limited interaction between push submissions. For example, a PPG implementation MAY support multiple delivery priorities. This could cause one message to affect the time at which another (e.g., lower priority) message is processed, and consequently the ultimate success or failure of its delivery. Note that a PPG is not required to deliver push messages in any specific order.

5.1 Push Submission Processing

A Push Initiator (PI) triggers push message processing by sending the PPG a push message. Push submission processing includes four operations. The following three operations must be performed in order:

- push submission acceptance or rejection,
- over-the-air message delivery, if the message is accepted and can be delivered in accordance with PPG policies and PI requirements; and
- message delivery result notification, if the message is accepted and the push initiator has requested message delivery notification.

The fourth operation may, as determined by the PPG implementation, be performed at any time after push message acceptance:

- PAP push message response.

These four functions are described in this section.

5.1.1 Push Submission Acceptance or Rejection

Each PAP push submission received by the PPG is either accepted or rejected.

The PPG SHOULD accept a PAP push submission if it might ultimately be delivered to the OTA client. The PPG MUST reject any push submission containing a PAP `push-message` element that is not valid with respect to its document type definition (DTD). Additional criteria used to determine whether to accept or reject a `push-message` are implementation dependent.

An accepted, undelivered PAP push submission for which message handling (described in the next section) for over-the-air delivery have not been completed MUST have the following message status reportable:

PAP Attribute	**Value**
`message-state`	`"pending"`

5.1.1.1 Replacement of a Previously Submitted Push Message

This OPTIONAL function allows replacement of a previously submitted, still pending push message.

If the PPG supports replacement, and the message is in a state from which message replacement may be assured, the PPG MUST replace the message as requested by a PAP `push-message` message [PushPAP].

A PPG that does not support the replace operation MUST reject the push submission if the PI requests replacement.

5.1.1.2 Request for Content from the Client

The PPG must support OTA-HTTP or OTA-SIP to adequately serve a `push-message` [PushPAP] indicating that the PI accepts content from the client in response to a confirmed push (by setting the `delivery-method` attribute to "`confirmed-with-response`" in the `quality-of-service` element). A PPG that does not support OTA-HTTP or OTA-SIP MUST reject the push submission if the PI uses this feature.

5.1.2 Over-the-Air Message Delivery

Over-the-Air message delivery consists of two functions:

- Message handling
- Over-the-air message transmission.

These functions are described in this section.

5.1.2.1 Message Handling

The PPG may transform the push message (as defined in [PushMsg]) entity contained within the push submission in preparation for over-the-air transmission. Typical reasons for transformation include compilations/optimisations for over-the-air efficiency, and translation of entities to a content type acceptable to the client. This section describes the transformations.

5.1.2.1.1 Entity and Header Transformation

A PPG MUST NOT transform the body of any entity, which falls under the scope of a No-Transform cache control directive as defined in [RFC2616]; otherwise, a PPG MAY translate entities in an implementation-dependent manner. The headers of all transformed entities MUST be revised as needed to correctly represent the transformed entity. All transformations MUST be in conformance with the requirements of [PushMsg].

5.1.2.1.1.1. WSP specific transformations

A PPG MUST support binary header encoding as specified in [WSP]. It MUST also encode content entities into their compact binary format [WBXML] (if such is specified) for transmission over OTA-WSP [PushOTA], unless it is positively known that the addressed terminal supports the non-encoded format. For example, Service Indication must usually be encoded into WBXML [WBXML] when delivered in connectionless mode.

5.1.2.1.1.2. HTTP specific transformations

A PPG SHOULD support content encoding for OTA transmission over OTA-HTTP [PushOTA] in order to minimize the volume of data sent over the air. When supported, the PPG MUST support deflate coding as specified in [RFC1951].

5.1.2.1.1.3. SIP specific transformations

A PPG SHOULD support content encoding for OTA transmission over OTA-SIP [PushOTA] in order to minimize the volume of data sent over the air. When supported, the PPG MUST support deflate coding as specified in [RFC1951]. Content entities MAY be encoded into their compact binary format [WBXML] when it is positively known that the addressed terminal supports the encoded format.

5.1.2.1.2 `X-Wap-Application-Id` (Application-ID) header processing

A PPG MUST process a [PushMsg] `X-Wap-Application-Id` (Application-ID) header as follows:

> If the header contains a [PushMsg] `absoluteURI` format Application-ID for which an `app-assigned-code` has been registered with [OMNA], the PPG MUST remove any [PushMsg] `app-assigned-code` format Application-ID (if present) from the header and then substitute the registered `app-assigned-code` format Application-ID for the `absoluteURI` format Application-ID.

If the header contains a [PushMsg] absoluteURI format Application-ID for which no app-assigned-code has been registered with [OMNA], the PPG MUST use this value unless a [PushMsg] app-assigned-code format Application-ID is present. In this case (if the app-assigned-code format Application-ID is present), the absoluteURI format Application-ID must be removed.

A header containing only a [PushMsg] app-assigned-code format Application-ID requires no substitutions or deletions.

If the resulting header identifies a default application known to the client, the PPG MAY delete this header.

If no [PushMsg] X-Wap-Application-Id header is present in the push message, the PPG MUST, unless the client's default Application-ID is the WML user agent, add this header. If added, the Application-ID MUST be that of the WML user agent.

If OTA-SIP is used then a PPG MUST add the X-WAP-Application-Id header value in the URN form into the g.oma.pusheventapp feature tag as Push Resource Identifier. If it is a well known value registered with OMNA then only the Namespace Specific String SHOULD be used. Otherwise the URN Namespace Identifier MUST be included.

Otherwise a PPG MAY remove any header, which specifies a default value known to the client. This default may be specified in the over-the-air protocol, provisioned, or established using an implementation-dependent mechanism. For example, an X-Wap-Application-Id header might be removed if a client has only one push application, optimising over-the-air communications. X-Wap-Application-Id headers containing a registered value MUST NOT be sent over the air without being encoded in numeric format.

5.1.2.1.3 Message State

For each push submission for which errors are encountered in the steps above, or for which it is apparent that successful message delivery is not possible, message delivery MUST NOT be attempted. Note that this may cause a PAP resultnotification-message to be sent. Messages that fail the entity and header transformation process MUST have the following status reportable:

PAP Attribute	Value
message-state	"undeliverable"
Code	"transformation-failure"

If message handling is successfully completed, an undelivered message MUST have the following status reportable:

PAP Attribute	Value
message-state	"pending"

5.1.2.2 Over-the-Air Transmission

The purpose of this function is to deliver messages to the OTA client. Key elements of this function are selection of Push OTA [PushOTA] protocol, selection of confirmed or unconfirmed push, and message delivery. A PPG implementation may include tests for message expiration and cancellation, message retransmission and delivery timeout, bearer management and WSP session (if OTA-WSP is used), registration context (if OTA-HTTP is used) management or registration status (if OTA-SIP is used).

5.1.2.2.1 Selection of Push OTA Protocol

A mobile terminal may support OTA-WSP, OTA-HTTP and OTA-SIP [PushOTA]. OTA-WSP or OTA-SIP may be selected for the connectionless push. The PPG selects the OTA protocol variant for connection-oriented push (from among those supported by the terminal and/or as required by the Push Initiator) in an implementation dependent manner.

The PPG MAY be able to select the OTA protocol variants through getting the terminals information or network information, e.g.

- for example base on checking if there has SIP Registration between mobile terminal and PPG
- if the bearers is available,
- if the mobile terminal supports the protocol variants based on terminal’s capability,
- or the PI’s indication, etc.

The PPG MAY also be able to hand over the decision to the terminal by sending a *Session Initiation Request* (SIR) that contains lists of contact points for OTA-WSP, OTA-HTTP and OTA-SIP. This approach and the SIR are defined in [PushOTA].

However, OTA-HTTP or OTA-SIP MUST be selected if the PI indicates that it accepts content from the client in response to a confirmed push (also see section 5.1.1.2). If the PPG fails to select OTA-HTTP or OTA-SIP, the PAP `resultnotification-message` MUST indicate failure of selecting the specified delivery method.

5.1.2.2.2 Bearer Network Selection

If the QoS section of the PAP `push-message` element requires a specific bearer and/or network to be used, the PPG MUST use the specified bearer and/or network, or fail to deliver the message with the following messages status reportable:

PAP Attribute	**Value**
`message-state`	`"undeliverable"`
`Desc`	An appropriate, implementation-dependent value
`event-time`	Time or estimated time of failure

5.1.2.2.3 Session or Registration Context Selection/Creation

The PPG may use an existing WSP session (if OTA-WSP is used) or registration context (if OTA-HTTP is used), or take implementation-dependent action(s) to create a suitable WSP session or registration context (e.g. send an OTA Session Initiation Request). For OTA-SIP, the PPG SHOULD use the procedures described in [PushOTA] to determine when delivery actions can be taken. If the PPG elects to attempt no further delivery action(s) due to the lack of and/or failure to create a suitable WSP session or registration context, the following messages status MUST be reportable:

PAP Attribute	**Value**
`message-state`	`"undeliverable"`
`Desc`	An appropriate, implementation-dependent value
`event-time`	Time or estimated time of failure

5.1.2.2.4 Delivery Time Constraints

If the PPG supports delivery time constraints, the PPG MUST NOT deliver the push message prior to the PAP `deliver-after-timestamp` time and MUST, if unable to deliver by the PAP `deliver-before-timestamp` time, fail with the following message status reportable:

PAP Attribute	**Value**
`message-state`	`"expired"`
`Desc`	An appropriate, implementation-dependent value
`event-time`	Time or estimated time of failure

5.1.2.2.5 Delivery

Assuming no errors, if OTA-WSP is used for OTA delivery, the PPG MUST deliver either a confirmed (`Po-ConfirmedPush`) or unconfirmed (`Po-Push` or `Po-Unit-Push`) [PushOTA] push primitive; if OTA-HTTP is used for OTA delivery, the PPG MUST deliver messages by using the HTTP POST method. If OTA-SIP is used, the PPG MUST deliver messages using SIP INVITE/MSRP. If OTA-HTTP or OTA-SIP is used and the PI indicates that it accepts content from the client in response to a confirmed push (also see section 5.1.1.2), the `X-Wap-Push-Info` header [PushOTA] MUST contain the "`response`" attribute token when the message is pushed to the client.

The use of confirmed or unconfirmed push depends on the PAP `delivery-method` attribute and implementation-dependent PPG policies.

5.1.2.2.5.1. Unconfirmed Push

A PPG MUST deliver `"unconfirmed"` messages using OTA-WSP (`Po-Push.req` or `Po-Unit-Push.req` primitive), OTA-HTTP or OTA-SIP. If OTA-HTTP or OTA-SIP is used, the PPG MUST report the same PAP `result-notification` message as if the message were pushed in an unconfirmed manner using OTA-WSP.

If the PPG sends a `Po-Push.req` or `Po-Unit-Push.req` primitive, or the PPG sends messages by using OTA-HTTP or OTA-SIP instead of these primitives, the following message status MUST be reportable:

PAP Attribute	**Value**
`message-state`	`"delivered"`
`Delivery-method`	`"unconfirmed"`
`event-time`	Time or estimated time of delivery

5.1.2.2.5.2. Confirmed Push

A PPG MUST deliver `"confirmed"` messages using OTA-WSP (`Po-ConfirmedPush.req` primitive), OTA-HTTP or OTA-SIP. If OTA-SIP is used then the INVITE/MSRP method MUST be used and a delivery confirmation MUST be requested by adding a Success-Report header set to yes. The remaining process depends on the type of push as follows:

If the PPG sends a `Po-ConfirmedPush.req` primitive or uses OTA-HTTP or uses the INVITE/MSRP method with delivery confirmation, the outcome depends as follows on whether or not the push message is acknowledged:

Success: If the PPG receives a `Po-ConfirmedPush.cnf` primitive indicating successful delivery to the OTA client, or a HTTP response including a `X-Wap-Push-Status` header indicating successful delivery or receives a MSRP Success-Report indicating successful delivery, possibly after a PPG's implementation-dependent retries, the following message status MUST be reportable:

PAP Attribute	**Value**
`message-state`	`"delivered"`
`Delivery-method`	`"confirmed"`
`event-time`	Time or estimated time of delivery

Failure due to abort: If the PPG receives a `Po-PushAbort.ind` primitive indicating an aborted push attempt (OTA-WSP), a `X-Wap-Push-Status` header indicating that the push message was rejected (OTA-HTTP) or MSRP Failure-Report indicating delivery failure (OTA-SIP), the following message status MUST be reportable:

PAP Attribute	**Value**
`message-state`	`"aborted"`
`Code`	PAP-specified representation of the abort parameter specified in [PushOTA]
`Desc`	An appropriate, implementation-dependent value
`event-time`	Time or estimated time of aborted delivery attempt

Failure due to timeout: If OTA-WSP is used, a timeout occurs when the PPG does not receive an OTA `Po-ConfirmedPush.cnf` primitive within an implementation-dependent period of time. If OTA-HTTP is used, a timeout occurs when the PPG does not receive a response to a HTTP POST request within an implementation-dependent period of time. If OTA-SIP is used a timeout occurs when the PPG does not receive a MSRP Success-Report indicating delivery success within an implementation-dependent period of time. If the PPG elects to attempt no further delivery action(s) when a timeout occurs, the following messages status MUST be reportable:

PAP Attribute	**Value**
`message-state`	`"timeout"`
`Desc`	An appropriate, implementation-dependent value
`event-time`	Time or estimated time of last delivery attempt

5.1.2.2.5.3. Oneshot delivery

A PPG MUST deliver "`oneshot`" messages as described in section 5.1.2.2.5.1. In addition the PPG MUST attempt to deliver the message only once, and ensure that a one-shot delivery attempt can be made on the underlying bearer. The following message status MUST be reportable for a message delivered using this method:

PAP Attribute	**Value**
`message-state`	`"delivered"`

`Delivery-method`	`"oneshot"`
`event-time`	Time or estimated time of delivery

5.2 Result Notification

The PPG MUST, if requested by the push initiator during push message submission, send a PAP `resultnotification-message` to the push initiator or its designee.

5.2.1 Time of Result Notification

A result notification, if requested, should be sent as soon as practical after the completion (successful or unsuccessful) of the Over-the-Air message delivery process.

5.2.2 Result Notification Contents

The PAP `resultnotification-message` indicates the reportable message status, which includes the message state and other information as specified earlier in this document. The status should reflect the message just before, within the limits of practicality, sending the result notification.

Assuming the PI requested a result notification and indicated that it accepts content from the client in response to a confirmed push (see section 5.1.1.2), content returned from the client in the response to a push via OTA-HTTP, if any, MUST be sent along with the `resultnotification-message`. If the PI did not indicate that it accepts content from the client in response to a confirmed push, the content entity MUST not be present when the `resultnotification-message` is returned to the PI. See [PushPAP] for further details.

5.3 PAP Status Query

This OPTIONAL function provides message status on receipt of a PAP `statusquery-message`.

The status query reply indicates the reportable message status, which includes the message state, and other information as specified earlier in this document. The status should reflect the message just before, within the limits of practicality, sending the result notification.

5.4 Delivery Cancellation

This OPTIONAL function allows delivery cancellation of a pending push message.

If the PPG supports cancellation of a push message, and the message is in a state from which delivery cancellation may be assured, the PPG MUST cancel delivery of the message as requested by a PAP `cancel-message`, and the following message status MUST be reportable:

PAP Attribute	**Value**
`message-state`	`"cancelled"`
`Desc`	An appropriate, implementation-dependent value
`event-time`	Time or estimated time of cancellation

If the PPG cannot assure cancellation of the message delivery, it MUST reject the delivery cancellation.

Successful cancellation of a push message will trigger a delivery result notification, if requested during the push message submission.

6. Client Addressing

Push Initiators are able to identify clients to the PPG using a special textual address format. The PPG MUST transform these addresses into a form that can be used to deliver over the wireless network. Conversely, the PPG MUST transform network-specific addresses into the textual address format for communication to a Push Initiator. If a Push Initiator has used a particular address value to identify a client in a request sent to the PPG, this address value MUST be used when referring to this client in the corresponding response and any subsequent result notification.

A client address is composed of a client specifier and a PPG specifier. Inclusion of the PPG specifier provides a mechanism to ensure that the address is unambiguous, permitting requests to be routed through proxies. The PPG specifier does not necessarily identify a physical PPG, and is not required to be the hostname of the PPG receiving the address from a PI.

There are multiple types of client specifiers. A PPG MUST support at least one of these client specifier types:

a) User-defined identifiers

b) Device addresses

User-defined identifiers are arbitrary values that are mapped to wireless network addresses in an unspecified manner. The PPG has complete control over which bearer-level address will be used in delivering the push message to the client. The user-defined identifier MAY be expanded to several bearer-level addresses for one or more clients. In this case the PPG MUST interact with the Push Initiator in the same way as when the user-defined identifier maps to a single bearer-level address. The interpretation of user-defined identifiers is based on a mutual understanding between the Push Initiator and the PPG. This permits them to be assigned values that are useful for the application using push services. For instance, they could be e-mail addresses.

Device addresses use static values from well-known network address spaces. One example is telephone numbers in the public land mobile network (PLMN). The PPG MAY use any of the client's bearer-level addresses in delivering the push message to the client. How the PPG determines this is not specified, but may be based, for instance, on the characteristics of the bearers used by the client.

The bearer-level address may invoke a point-to-multipoint delivery in the wireless network, for example, using cell broadcast. In this case there still MUST be a single result notification, if one has been requested.

6.1 Client Address Format

The external representation of addresses processed by the PPG is defined using ABNF [RFC4234]. The format is compatible with Internet e-mail addresses [RFC822]. The PPG MUST be able to parse this address format, and it MUST be able to determine whether it supports the specified address type or not.

```
wappush-address = ["/"] wappush-client-address ["/"] "@" ppg-specifier

wappush-client-address = "WAPPUSH" "=" client-specifier
ppg-specifier = dom-fragment *( "." dom-fragment )
dom-fragment = ( ALPHA / DIGIT ) *( ALPHA / DIGIT / "-" )

client-specifier = ( user-defined-identifier / device-address )
user-defined-identifier = ( escaped-value ext-qualifiers "/TYPE=USER" )
device-address = ( global-phone-number ext-qualifiers "/TYPE=PLMN" )
             / ( ipv4 ext-qualifiers "/TYPE=IPv4" )
             / ( ipv6 ext-qualifiers "/TYPE=IPv6" )
             / ( man ext-qualifiers "/TYPE=MAN" )
             / ( URI ext-qualifiers "/TYPE=URI" )
             / ( escaped-value ext-qualifiers "/TYPE=" address-type )
address-type = 1*address-char
; A network bearer address type [WDP]
```

```
address-char = ( ALPHA / DIGIT / "_" )

ext-qualifiers = *( "/" keyword "=" value )
; for future extensions, e.g. special well-known user-defined identifier types
keyword = 1*( DIGIT / ALPHA / "-" )
value = 1*( %x20-2E / %x30-3C / %x3E-7E )
escaped-value = 1*( safe-char )
; the actual value escaped to use only safe characters by replacing
; any unsafe-octet with its hex-escape
safe-char = ALPHA / DIGIT / "+" / "-" / "." / "%" / "_"
unsafe-octet = %x00-2A / %x2C / %x2F / %x3A-40 / %x5B-60 / %x7B-FF
hex-escape = "%" 2HEXDIG   ; value of octet as hexadecimal value

global-phone-number = "+" 1*( DIGIT / written-sep )
written-sep = ( "-" / "." )
ipv4 = 1*3DIGIT 3( "." 1*3DIGIT )       ; IPv4 address value [RFC791]
ipv6 = 4HEXDIG 7( ":" 4HEXDIG )         ; IPv6 address value [RFC3513]
man = 8DIGIT                            ; Mobitex MAN address format [Mobitex]
```

Each value of a `user-defined-identifier` is a sequence of arbitrary octets. They can be safely embedded in this address syntax only by escaping potentially offending values. The conversion to `escaped-value` is done by replacing each instance of `unsafe-octet` by a `hex-escape` which encodes the numeric value of the octet.
URI is as defined in [RFC3986] with the additional requirement that any character not in the safe-char set needs to be hex escaped, e.g. it needs to match escaped-value.

6.2 Client Address Examples

Addresses using user-defined identifiers:

```
WAPPUSH=john.doe%40wapforum.org/TYPE=USER@ppg.carrier.com
; user-defined identifier for john.doe@wapforum.org

wappush=47397547589/type=user@carrier.com
; user-defined identifier for 47397547589

WAPPUSH=47397547589/TYPE=USER@Carrier.com
; equivalent to previous one

WAPPUSH=+155519990730/TYPE=USER@ppg.carrier.com
; user-defined identifier that looks like a phone number
```

Addresses using device addresses:

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

WAPPUSH=FEDC:BA98:7654:3210:FEDC:BA98:7654:3210/TYPE=IPv6@carrier.com
; device address for an IP v6 address

WAPPUSH=195.153.199.30/TYPE=IPv4@ppg.carrier.com
; device address for an IP v4 address
```

WAPPUSH=12345678/TYPE=MAN@ppg.carrier.com
; device address for a MAN address

WAPPUSH=sip%3Aalice%40atlanta.com/TYPE=URI@ppg.carrier.com
; device address for an URI address

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
Approved Version: OMA-TS-PPGService-V2_2-20110809-A	09 Aug 2011	Status changed to Candidate by TP: OMA-TP-2011-0282-INP_Push_V2_2_ERP_for_Final_Approval

Appendix B. Static Conformance Requirements (Normative)

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

B.1 SCR for Push Proxy Gateway

B.1.1 Predicates

These items are only used as predicates and do not state any requirements on the implementation.

Item	Function	Reference	Status	Requirement
PPG-CO-S-001	Confirmed push is supported		O	(OTA-CO-S-002 OR OTA-CO-S-003) AND PPG-GEN-S-013

B.1.2 Operations

Item	Function	Reference	Status	Requirement
PPG-GEN-S-001	Push Submission Rejection	5.1.1	M	
PPG-GEN-S-002	Incomplete message handling reportable	5.1.1	M	
PPG-GEN-S-003	Entity transformation under the scope of a No-Transform cache control directive	5.1.2.1.1	M	
PPG-GEN-S-004	Revising headers of transformed entities	5.1.2.1.1	M	
PPG-GEN-S-005	`X-Wap-Application-Id` header processing	5.1.2.1.2	M	
PPG-GEN-S-006	Registered `X-Wap-Application-Id` value sent over-the-air in numeric encoded format	5.1.2.1.2	M	
PPG-GEN-S-007	Reportable message states	5.1.2.1.3	M	
PPG-GEN-S-008	Bearer Network Selection (QoS)	5.1.2.2.2	M	
PPG-GEN-S-009	Reporting of failed Session or Registration Context Selection/Creation	5.1.2.2.3	M	
PPG-GEN-S-010	Delivery Time Constraints	0	M	
PPG-GEN-S-011	Delivery	0	M	
PPG-GEN-S-012	Reportable status associated with unconfirmed push	5.1.2.2.5.1	M	
PPG-GEN-S-013	Reportable statuses associated with confirmed push	5.1.2.2.5.2	O	
PPG-GEN-S-014	Sending of `resultnotification-message`	5.2	M	

PPG-GEN-S-015	PAP Status Query	5.3	O	PAP-OPS-S-004
PPG-GEN-S-016	Delivery Cancellation	5.4	O	
PPG-GEN-S-017	Handling message cancellation request	5.4	M	
PPG-GEN-S-018	Support for WSP specific transformations	5.1.2.1.1.1	M	
PPG-GEN-S-019	Support for HTTP specific transformations	5.1.2.1.1.2	O	OTA-CO-S-003
PPG-GEN-S-020	Support for push message replacement	5.1.1.1	O	
PPG-GEN-S-021	Support for binary header encoding	5.1.2.1.1.1	M	
PPG-GEN-S-022	Support for content encoding using WBXML	5.1.2.1.1.1	M	
PPG-GEN-S-023	Support for content encoding using 'deflate'	5.1.2.1.1.2	O	
PPG-GEN-S-024	Handling of push a `push-message` with the `delivery-method` attribute set to "`confirmed-with-response`" in the `quality-of-service` element	5.1.1.2	M	
PPG-GEN-S-025	Selection of Push OTA Protocol	5.1.2.2.1	M	
PPG-GEN-S-026	Inclusion of content returned from the client in a `resultnotification-message`	5.2.2	M	
PPG-GEN-S-027	Reportable statuses associated with oneshot delivery	5.1.2.2.5.3	O	

B.1.3 Client Addressing

Item	Function	Reference	Status	Requirement
PPG-ADD-S-001	Client Addressing	6	M	PPG-ADD-S-002 OR PPG-ADD-S-003
PPG-ADD-S-002	Support for user-defined identifiers	6	O	
PPG-ADD-S-003	Support for device addresses	6	O	
PPG-ADD-S-004	Support for client address format	6.1	M	