Interface Specification

Universal Gateway Push Request Protocol

Document number: 90-290

Revision: 1.3 2010-03-23

Contents

Document number: 90-290
Revision: 1.3 2010-03-23

Document number: 90-290
Revision: 1.3 2010-03-23

1 Introduction

This document describes the communication protocol between the Universal Gateway (UG) and the Push Client that the UG supports from version 3.3. This corresponds to Delivery Platform version 6.0 and later DP version. The Push Client is defined as the program that generates a Push Request to be sent to a Mobile Station.

The intended audience is persons that design and implement applications that perform push operations as part of the application.

For backward compatibility, the UG supports earlier version of this document as well, but the use of the syntax specified herein is encouraged.

The protocol used for Push Requests is a subset of the *WAP Push Access Protocol* [5] (PAP) over HTTP [1].

The UG has to be correctly configured in order to support this specification.

1.1 Backward compatibility

Special attention has been paid to ensuring smooth evolution of the specifications. Wherever backward compatibility with DP 5 has not been possible to achieve, it has been clearly stated in the *Universal Gateway & Wireless Service Management – Application Developers Guide – Development of Wib Services* [2].

The WIG WML used in the examples is supported by Delivery Platform version 6.1 and later DP versions.

Document number: 90-290
Revision: 1.3 2010-03-23

2 References

[1] RFC 2616. Hypertext Transfer Protocol - HTTP/1.1. June 1999.

[2] *Universal Gateway & Wireless Service Management – Application Developers Guide – Development of Wib Services*, SmartTrust.

[3] *Universal Gateway Request Protocol – Interface Specification*, SmartTrust.

[4] *WIG WML v. 5 – Specification*, SmartTrust.

[5] Wireless Application Protocol Forum. Push Access Protocol. 29 April 2001. Available: http://www.openmobilealliance.org/

[6] Wireless Application Protocol Forum. WAP Push Architectural Overview. 3 July 2001. Available: http://www.openmobilealliance.org/

[7] S@T 01.23 V3.0.0 (Release 2007) S@T Push Commands

3 Definitions and abbreviations

Acronym	Definition
DTD	Document Type Definition
DP	SmartTrust Delivery Platform
PAP	Push Access Protocol
S@T	SIM Alliance Toolbox
UG	Universal Gateway
WAP	Wireless Application Protocol
Wib	SmartTrust Wib™
WIG	Wireless Internet Gateway
WML	Wireless Markup Language
XML	eXtensible Markup Language

4 WAP Push Access Protocol

This chapter introduces the WAP Push Access Protocol. The UG supports a subset of the WAP Push Access Protocol.

In the WAP technology a push operation [6] occurs when a Push Initiator (Push Client) transmits content to a WAP client using the *WAP Push Access Protocol (PAP)* [5] and the Push Over-The-Air (OTA) Protocol. The normal situation is that the Push Client is on the Internet and that the WAP client is in the WAP domain and therefore a translating Gateway is needed. The Push Access Protocol then defines the communication protocol between a Push Client on the Internet and the Gateway. The Push Access Protocol is designed to be independent of the underlying transport protocol, and a PAP message contains a control entity, and in the case of a push submission also a content entity and optionally a capability entity.

The WAP Push Access Protocol supports five operations:

- Push submission
- Result notification
- Push cancellation (Not supported by UG)
- Status query (Not supported by UG)
- Client capabilities query (Not supported by UG)

4.1 Addressing

There are three addresses involved in the Push Access Protocol.

4.1.1 Push Proxy Gateway Address

The IP address and port number to access the Gateway.

4.1.2 Mobile Station Address

The push destination address given by the Push Client. This is the MSISDN of the Mobile Station.

4.1.3 Result Notification Address

The Result notification address is a URL to be used for notifications from the Gateway. This is given by the Push Client in the `ppg-notify-requested-to` attribute in PAP.

5 UG Push Overview

This Chapter gives an overview of the Push features supported by the UG.

The protocol between the Push Client and the UG supports a subset of the features described in the *WAP Push Access Protocol* [5], namely the Push submission and Result notification parts. HTTP [1] is used as the underlying transport protocol.

To handle secure push transactions the Secure Socket Layer (SSL) may be used between the UG and the Push Client.

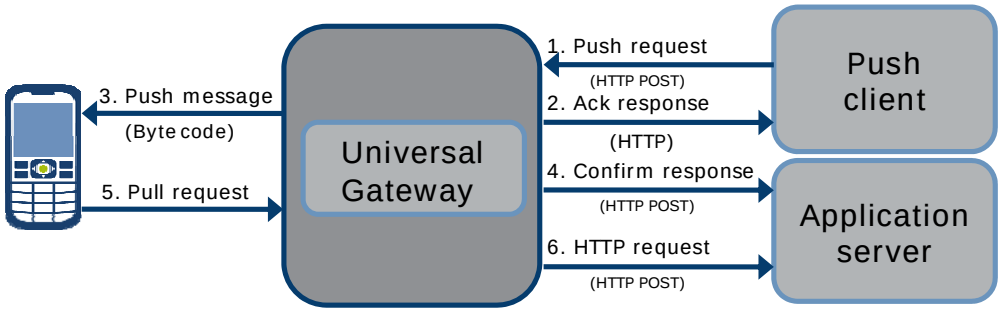

Figure 1. UG Push Architecture. Steps 4 - 6 are optional.

The above figure shows the basic system configuration for the UG, the Push Client and the Content Provider Application Server.

1. A *Push Request* is sent from the Push Client to the UG. This corresponds to the *Push submission* in the Push Access Protocol (PAP). See Chapter 6 .
2. The UG directly responds with an *Acknowledge Response*. See Chapter 7 .
3. The UG processes the Push Request and sends it to Wib.
4. Optionally a *Confirm Response* can be sent from the UG when the push message has been delivered to Wib. This corresponds to the *result notification* specified by PAP. For details on the *Confirm Response*, see Chapter 8 .
5. If the Push Request contains a go element to a URL, for example something like
   `<go href="http://www.content-provider.com/test.wml"/>`,
   this will generate a Wib request to the UG.
6. The UG will transform the Wib request to an HTTP request and send it to the given URL.

The UG supports multiple addressing. This should be used when a Push Client wants to send an identical message to multiple subscribers.

6 Push Request

The UG supports the HTTP request protocol to transfer the Push Request from the Push Client. The Push Request shall contain a valid document according to this specification.

The Push Request has the following structure:

```
Request = Request-Line
*(entity-header)
CRLF
[ message-body ]

Request-Line = Method SP Request-URI SP HTTP-Version CRLF
```

6.1 Request-Line

```
Method = "POST"
Request-URI = "/dummy.jsp"
HTTP-Version = "HTTP/1.1"
```

Note that the value for the Request-URI is just an example. The Request-URI is currently not used by the UG.

Example (1)

```
POST /notused.jsp HTTP/1.1
```

6.2 Entity-header

The headers recognised by the UG are described in the following subsections. Note that each entity-header has to be followed by the `CRLF` (Carriage Return Line Feed) characters.

6.2.1 Content-Length

The `Content-Length` entity-header field indicates the size of the message-body in decimal number of octets according to HTTP [1].

6.2.2 Content-Type

The content-type used for Push Requests is the following:

```
Content-Type: multipart/related; boundary=BOUNDARY;
  type="application/xml"
```

The `BOUNDARY` may be changed to any document unique character string. It is recommended to use `asdlfkjiurwghasf`.

Example (2)

Content-Type: multipart/related; boundary=asdlfkjiurwghasf;
type="application/xml"

6.2.3 Host

This header is currently not used by the UG.

Example (3)

```
Host: wigserverhost:5012
```

6.2.4 X-WAP-Payment-Info

The `X-WAP-Payment-Info` is an optional header and indicates the tariff class to be used for billing purposes by DP. For further discussion on that parameter please see *UG & WSM – Application Developers Guide – Development of Wib Services* [2].

Example (4)

```
X-WAP-Payment-Info: content-value-class=42
```

6.3 Message-body

The message-body of the HTTP request contains the following.

```
--BOUNDARY
Control section
--BOUNDARY
Message section
--BOUNDARY
```

`BOUNDARY` should be the same as defined in the `Content-Type` header. See Section 6.2.2 . The WAP PAP also specifies a *client capabilities section*, but that section is not supported by the UG.

6.3.1 Control Section

The Control section contains a `Content-Type: application/xml` and a PAP control entity according to Section 6.4 .

Example (5)

Example of a control section:

```
Content-Type: application/xml
<?xml version="1.0"?>
<!DOCTYPE pap PUBLIC "-//WAPFORUM//DTD PAP 2.0//EN"
"http://www.wapforum.org/DTD/pap_2.0.dtd">
<pap>
...PAP content...
</pap>
```

6.3.2 Message section

The Message section contains a Content-Type: text/vnd.wap.wml header and a WML document according to *the WIG WML Specification* [4].

Example (6)

Example of a message section:

```
Content-Type: text/vnd.wap.wml

<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE wml PUBLIC "-//SmartTrust//DTD WIG-WML 4.0//EN"
  "http://www.smarttrust.com/DTD/WIG-WML4.0.dtd">
<wml>
...wml content...
</wml>
```

6.4 PAP control entity

The PAP control entity is based on XML and uses a Document Type Definition (DTD) document, `pap_2.0.dtd` [5], specified by the WAP Forum. The DTD defines all allowed fields in the control section and in the capabilities section.

The PAP control entity is used to define the following data, among other things.

- MSISDN - The phone number of the destination Mobile Station.
- Confirmation flag and the URL where the confirmation shall be sent.
- A unique ID for the Push Request.

The PAP control entity contains a Prologue and the four different elements described below.

6.4.1 Prologue

The PAP control entity should start with an XML declaration and a document type declaration referring to the WAP PAP DTD [5].

```
<?xml version="1.0"?>
<!DOCTYPE pap PUBLIC "-//WAPFORUM//DTD PAP 2.0//EN"
"http://www.wapforum.org/DTD/pap_2.0.dtd">
```

6.4.2 pap Element

```
<!ELEMENT pap (push-message)>
<!ATTLIST pap>
```

Description

The `pap` element contains the `push-message` element.

Contained elements

`push-message`

Syntax

`<pap>content</pap>`
No attributes.

6.4.3 push-message Element

```
<!ELEMENT push-message (address+, quality-of-service?) >
<!ATTLIST push-message
 push-id                  CDATA  #REQUIRED
 ppg-notify-requested-to  CDATA  #IMPLIED
>
```

Description

The `push-message` is used to contain the `address` and `quality-of-service` elements.

Contained elements

```
address +
quality-of-service ?
```

Syntax

`<push-message>content</push-message>`

Attribute	Explanation	
`push-id`	The ID of the push message is given by the Push Client. The ID shall preferably be unique for each push message. A recommendation is to use the format `<number>@<content-provider>`	M
`ppg-notify-requested-to`	The URL to be used by the UG for the Confirm Response.	O

6.4.4 quality-of-service Element

<!ELEMENT quality-of-service EMPTY >
<!ATTLIST quality-of-service
 delivery-method (confirmed|unconfirmed) "unconfirmed"
 priority (high|low) "high"

Description

The `quality-of-service` element contains the attribute for specifying the delivery method.

Contained elements

`NONE.`

Syntax

`<quality-of-service/>`

Attribute	Explanation	
`delivery-method`	This attribute specifies if the push message shall be confirmed. `confirmed` - The UG will send a Confirm Response when the push message has been delivered to the Mobile Station. `unconfirmed` - The UG will not send a Confirm Response. (Default)	M
`priority`	This attribute specifies the priority of the push. Only applicable for S@T, for more information please see [7]. `high` – High priority push,establishing a session (Default) `low` – Low priority push, no session established	O

6.4.5 address Element

```
<!ELEMENT address EMPTY >
<!ATTLIST address
 address-value  CDATA   #REQUIRED
>
```

Description

The `address` element contains the attribute for specifying the push destination. This element can be used multiple times for multiple addressing.

Contained elements

`NONE.`

Syntax

`<address/>`

Attribute	Explanation	
`address-value`	The address of the Mobile Station, i.e. the MSISDN.	O

6.4.6 sat-push Element

```
<!ELEMENT sat-push EMPTY >
<!ATTLIST sat-push
 sat-push-name  CDATA   #IMPLIED
>
```

Document number: 90-290
Revision: 1.3 2010-03-23

Description

The sat-push element contains SAT specific information. This element is an extension of the pap protocol.

Contained elements

NONE.

Syntax

< sat-push**/>**

Attribute	Explanation	
sat-push-name	This attribute specifies the SAT push name according to specification [7] The name will show up in the display of the mobile terminal if it is supported.	O

6.5 Push Request example

Example (7)

This example illustrates a Push Request that will generate a push message to be sent to two Mobile Stations. The UG will generate two Confirm Responses.

```
POST /dummy.jsp HTTP/1.1
Host: wigserver:5012
Content-Type: multipart/related; boundary=asdlfkjiurwghasf;
 type="application/xml"
Content-Length: 832
X-WAP-Payment-Info: content-value-class=167

--asdlfkjiurwghasf
Content-Type: application/xml
<?xml version="1.0"?>
<!DOCTYPE pap PUBLIC "-//WAPFORUM//DTD PAP 2.0//EN"
"http://www.wapforum.org/DTD/pap_2.0.dtd">
<pap>
  <push-message
    push-id="9fjeo39jf084@content-provider.com"
    ppg-notify-requested-to="www.content-provider.com:8080/pushapp.jsp">
    <address address-value="+45700000000"/>
    <address address-value="+45700000001"/>
    <quality-of-service delivery-method="confirmed"/>
  </push-message>
</pap>

--asdlfkjiurwghasf
Content-Type: text/vnd.wap.wml
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE wml PUBLIC "-//SmartTrust//DTD WIG-WML 4.0//EN"
  "http://www.smarttrust.com/DTD/WIG-WML4.0.dtd">
<wml>
  <card>
    <p>
      Hello! This message is sent to two mobile stations...
    </p>
  </card>
</wml>

--asdlfkjiurwghasf
```

7 Acknowledge Response

The UG generates two types of responses to a Push Request. This chapter describes the Acknowledge Response that is always sent to the Push Client. Optionally a Confirm Response can be sent when the UG gets a notification that the Push Request has been delivered to the Mobile Station. See Chapter 8 .

The UG implementation differs from the WAP-Specification in that only one Acknowledge Response will be created and sent to the Push Client even if multiple Wib addressing is used.

The Acknowledge Response is always formatted as a regular HTTP response. Three different status codes are used by the UG:

- 200
- 408
- 500

7.1 OK acknowledge

Status-Code 200 indicates that the UG has received the push message.

Example (8)

This is the normal response by the UG.

```
HTTP/1.1 200 OK
Server: DP 6.1
Connection: close
Date: Mon, 15 Oct 2001 12:52:51 GMT
Content-length: 32

<wml>
Push message OK
</wml>
```

7.2 Time-out acknowledge

Status-Code 408 indicates that the push connection has timed out. If a Push Client initiates a connection, but does not send all the data within a certain time limit, the UG will generate this response and close the connection.

Document number: 90-290
Revision: 1.3 2010-03-23

Example (9)

The UG generates this response if the push connection times out.

```
HTTP/1.1 408 Request Time-out
Server: DP 6.1
Connection: close
Date: Mon, 15 Oct 2001 12:54:51 GMT
Content-length: 48

<wml>
Time-out receiving push message
</wml>
```

7.3 Error acknowledge

Status-Code 500 indicates that an internal server error has occurred.

Example (10)

The UG generates this response if an internal error has occurred.

```
HTTP/1.1 500 Internal Server Error
Server: DP 6.1
Connection: close
Date: Mon, 15 Oct 2001 12:58:51 GMT
Content-length: 33

<wml>
WIG Server error
</wml>
```

8 Confirm Response

The UG generates two types of responses to a Push Request. This chapter describes the Confirm Response that is sent when the UG gets a notification that the Push Request has been delivered to the Mobile Station. The UG will generate one Confirm Response for each delivered push message, in case of multiple addressing.

The Confirm Response is always formatted as a regular HTTP POST request, and the message-body contains the control section of the initial Push Client Request. The control section will contain exactly one `address` element with the `address-value` attribute set to the MSISDN of the Mobile Station subject to the delivery. Other `address` elements present in the initial Push Client Request will be removed by the UG before the Confirm Response is sent back to the Application Server.

The Confirm Response is sent to the URL specified in the `ppg-notify-requested-to` attribute. If no URL has been specified, no Confirm Response will be sent. See Section 6.4.3 .

Example (11)

Example of a Confirm Response.

```
POST /pushapp.jsp HTTP/1.1
Host: www.content-provider.com:8080
Connection: close
Content-Type: multipart/related; boundary=asdlfkjiurwghasf;
 type="application/xml"
Content-Length: 471

--asdlfkjiurwghasf
Content-Type: application/xml
<?xml version="1.0"?>
<!DOCTYPE pap PUBLIC "-//WAPFORUM//DTD PAP 2.0//EN"
"http://www.wapforum.org/DTD/pap_2.0.dtd">
<pap>
  <push-message
    push-id="9fjeo39jf084@content-provider.com"
    ppg-notify-requested-to="www.content-provider.com:8080/pushapp.jsp">
    <address address-value="+45700000000"/>
    <quality-of-service delivery-method="confirmed"/>
  </push-message>
</pap>
--asdlfkjiurwghasf
```

9 Mobile Station response

If the Push Request requires a response from the Mobile Station, the WML document shall contain a `<go href="..."/>` element or any other WML element that generates a normal Wib request to the UG. See *WIG WML Specification* [4].

Appendix A Example of a Push Session

This Appendix illustrates the 6 steps used for communication between the Push Client, the UG, a Wib and a Web Server. The 6 steps are illustrated in figure A1.

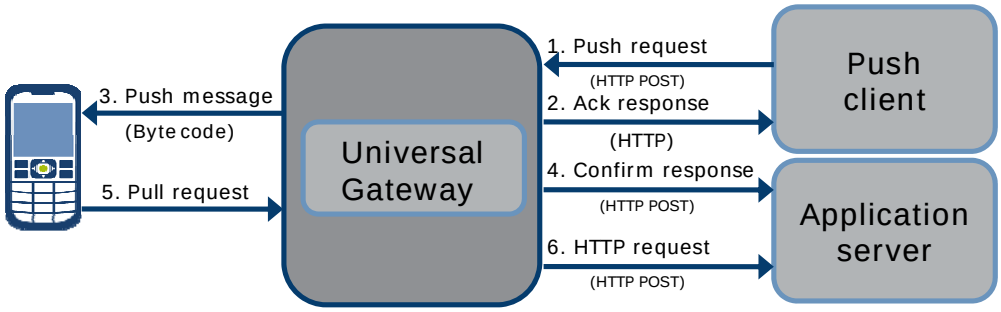

Figure A1. UG Push Architecture. Steps 4 - 6 are optional.

Step 1 - Push Request

The Push Client sends a request to the UG. In this example only one MSISDN is used. This is the syntax for the HTTP POST request used:

```
POST /SendToWIB.jsp HTTP/1.1
Content-Type: multipart/related; boundary=asdlfkjiurwghasf;
  type="application/xml"
Host: wigserver:5012
Content-Length: 791
X-WAP-Payment-Info: content-value-class=167

--asdlfkjiurwghasf
Content-Type: application/xml
<?xml version="1.0"?>
<!DOCTYPE pap PUBLIC "-//WAPFORUM//DTD PAP 2.0//EN"
"http://www.wapforum.org/DTD/pap_2.0.dtd">
<pap>
  <push-message push-id="9fjeo39jf084@content-provider.com"
    ppg-notify-requested-to="www.content-provider.com/pushlog.jsp">
    <address address-value="+46123456789"/>
    <quality-of-service delivery-method="confirmed"/>
  </push-message>
</pap>

--asdlfkjiurwghasf
Content-Type: text/vnd.wap.wml
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE wml PUBLIC "-//SmartTrust//DTD WIG-WML 4.0//EN"
  "http://www.smarttrust.com/DTD/WIG-WML4.0.dtd">
<wml>
  <card>
    <p>
      You have received a push message!
      <go href="www.content-provider.com/pushresponse.jsp"/>
    </p>
  </card>
</wml>

--asdlfkjiurwghasf
```

Step 2 - Acknowledge Response

The UG immediately sends an Acknowledge Response to the Push Client.

```
HTTP/1.1 200 OK
Server: DP 6.1
Connection: close
Date: Mon, 15 Oct 2001 12:52:51 GMT
Content-length: 32

<wml>
Push message OK
</wml>
```

Step 3 - Delivery of push message

The bytecode is transmitted to Wib.

Step 4 - Confirm Response

Since the `delivery-method` attribute had been set to `confirmed` in the initial Push Request in Step 1, a Confirm Response is sent to the URL specified in the `ppg-notify-requested-to` attribute. This response will be sent when the push message has been delivered to the Mobile Station. This is the syntax for the HTTP POST request used:

```
POST /pushlog.jsp HTTP/1.1
Host: www.content-provider.com
Connection: close
Content-Type: multipart/related; boundary=asdlfkjiurwghasf;
 type="application/xml"
Content-Length: 429

--asdlfkjiurwghasf
Content-Type: application/xml
<?xml version="1.0"?>
<!DOCTYPE pap PUBLIC "-//WAPFORUM//DTD PAP 2.0//EN"
"http://www.wapforum.org/DTD/pap_2.0.dtd">
<pap>
  <push-message push-id="9fjeo39jf084@content-provider.com"
    ppg-notify-requested-to="www.content-provider.com/pushlog.jsp">
    <address address-value="+46123456789"/>
    <quality-of-service delivery-method="confirmed"/>
  </push-message>
</pap>
--asdlfkjiurwghasf
```

Step 5 - Wib Request

Since the pushed WML document in Step 1 contains a `<go href="..."/>` element, Wib will send a new request to the UG.

Step 6 - HTTP Request

The UG will transform the Wib request to an HTTP request and send it to the given URL. See also *UG Request Protocol Specification* [3]. The request will have this syntax:

```
GET /pushresponse.jsp HTTP/1.1
Accept: text/*
Accept-Charset: iso-8859-1, UTF-8
Accept-Encoding: identity
User-Agent: WIG Browser/1.2
Connection: close
Host: www.content-provider.com
```