Push Backward Compatibility

Delivering WAP Push Functionality to Devices with Openwave™ UP.Notify Technology

About Openwave

Openwave Systems Inc. (Nasdaq: OPWV) is the worldwide leader of open IP-based communication infrastructure software and applications. Openwave is a global company headquartered in Redwood City, California. For more information, please visit www.openwave.com.

Push Backward Compatibility

Delivering WAP Push Functionality to Devices with Openwave™ UP.Notify Technology

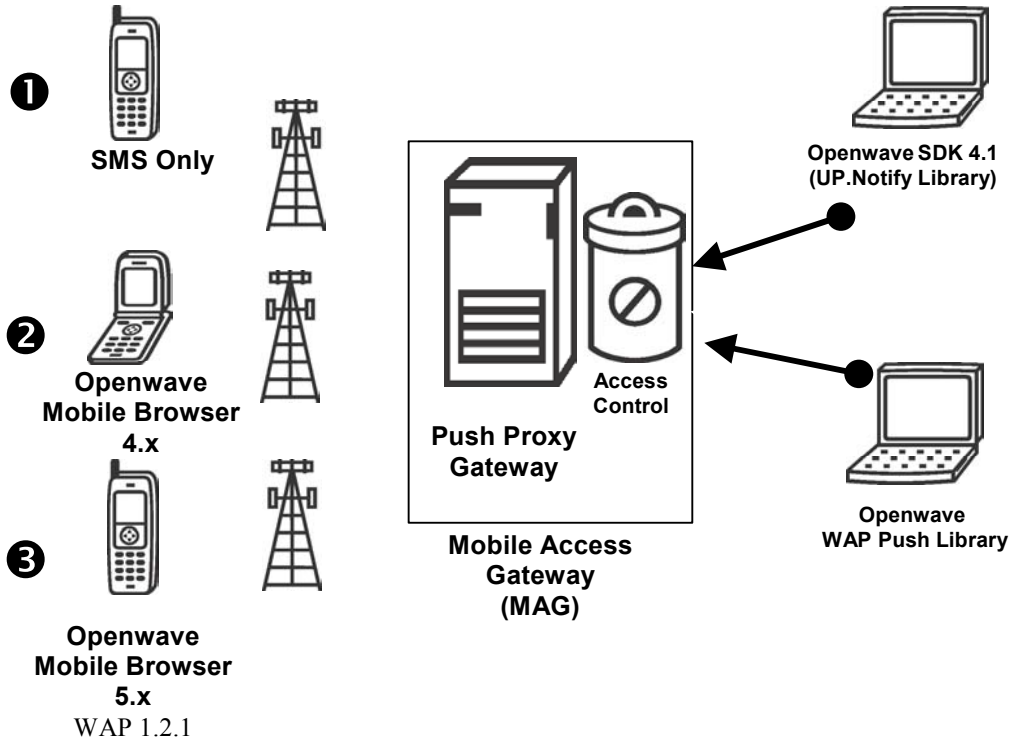
Table of Contents

Introduction

This document compares the earlier Openwave UP.Notify push technology to the functionality offered by WAP Push. UP.Notify is supported by Openwave™ UP.Notify Library (included with Openwave™ SDK 4.1 and earlier), and is used by legacy applications to push content exclusively to devices running Openwave™ Mobile Browser 4.x and earlier. WAP Push is supported by Openwave WAP Push Library, and is used by new applications to push content to any WAP 1.2.1 compatible device. Openwave™ Mobile Access Gateway (MAG) and its Push Proxy Gateway (PPG) component provide backward compatibility so that WAP Push enabled applications may push content to earlier UP.Notify enabled devices.

In summary, Openwave is committed to open standards as well as continuing to support and adapt previously deployed technology in order to preserve past investments of network operators and application developers.

The following illustrates the basic architecture for push:



This diagram illustrates the high degree of interoperability offered by the Openwave™ Push Proxy Gateway (PPG). In ❶ the PPG can communicate to directly to a plain text SMS only device. In ❷ the PPG may communicate using UP.Notify to an Openwave 4.x browser. In ❸the PPG is communicating directly to a WAP Push enabled device.

Note : This document does not discuss the use of Net Alert, a plain text alert system which predates UP.Notify and may be communicated through ❶.

UP.Notify Alert

Openwave UP.Notify is a predecessor of the WAP Push capability and is available in mobile devices running Openwave Mobile Browser 4.x and earlier. This protocol allowed for content to be submitted for delivery to the mobile device (push). The Openwave gateway would accept the content and then initiate contact with the user. Applications that use UP.Notify are able to request the status of or cancel previous submission in a similar manner to WAP Push. However not all WAP Push features have an equivalent in UP.Notify.

In UP.Notify a standard alert has the following semantics when processed by the browser:

- User agents are required to manage incoming alerts by only keeping track of the most recently received alert coming from a given application. This is accomplished by removing from the alert inbox any alert whose HREF attribute matches that of the alert being added to the alert inbox (using a HREF matching algorithm based on RFC2396).

- Depending on its urgency and the preferences specified by the user, user agents may also have to notify the user that an alert has been received through a physical signal (e.g. a beep).

Whether or not an alert is communicated to the user through a physical signal, the only persistent effect an alert has on the client is that it is added to the inbox when it is pushed or pulled to the client. It has no other permanent effect on the state of the client (e.g. its history).

In terms of what is generated over the air, an XML document is converted to its compact binary representation. The following is an example of an XML document that is generated for an alert:

```
<?xml version="1.0"?>

<!DOCTYPE ALERT
PUBLIC "-//OPENWAVE//DTD ALERT 1.0//EN"
      "http://www.openwave.com/DTD/alert.xml">

<ALERT LABEL ="Mobile Mail: You have new e-mail"
 COUNT ="7"
 HREF ="/UP.mail" URGENCY ="HIGH" />
```

This may be transported using SMS or using the user's current session via a specific type of request. A number of HTTP headers may be associated with a request. Of these the salient ones are:

- Content-Location
  This represents the source of the alert. The user agent must ensure that the HREF attribute in the alert has the same scheme and netloc as the Content-Location header.

- Content-Type : `application/vnd.uplanet.alert`

- Content Length: as calculated from tokenized form

- Sequence identifier used to coordinate the sequencing of push messages such as alerts in a one-way push environment.

The core elements of an alert have the following semantics:

- LABEL is a short textual description of the alert. Note the emphasis on the word short. Although no maximum size is specified, browsers are allowed to truncate LABEL arbitrarily.
- HREF which is a required URI to be accessed when the alert is selected. It is the key used by the user agent in removing older alerts coming from the same source.
- COUNT is an optional, application dependent element of information about the state of the application specified in HREF. An example of the use of this value is to indicate the count of the number of outstanding emails for a messaging application.

It should be noted that there are a number of other content types supported by this interface such as cache operations. For more information, refer to the Openwave SDK 4.1 Developer's Guide.

UP.Notify Transport

The transport options offered by UP.Notify may be specified in the type of mechanism used for delivery. This is referred to as `X-Up-Ntfn-channel`. The values that may be chosen are 'push' and 'pull'. If the 'push' option is chosen, it is an indicator to the Messenger component of the Openwave MAG that the method of delivery should be over SMS. If it is pull, then it is an indicator that the mechanism of reusing the user's current session should be used.

The alternate mechanisms for delivery are highly dependent on the specific nature of the underlying network, which is used to communicate to the mobile device. The UP.Notify Library (included with Openwave SDK 4.1) insulates the initiator of the message from these specifics (in the same way as Openwave WAP Push Library).

The UP.Notify mechanism also provides for the ordered delivery of the push messages, but does not allow for the partitioning of a large message over a number of SMS messages on the 'push' channel.

Summary

The following summarizes how an UP.Notify alert works:

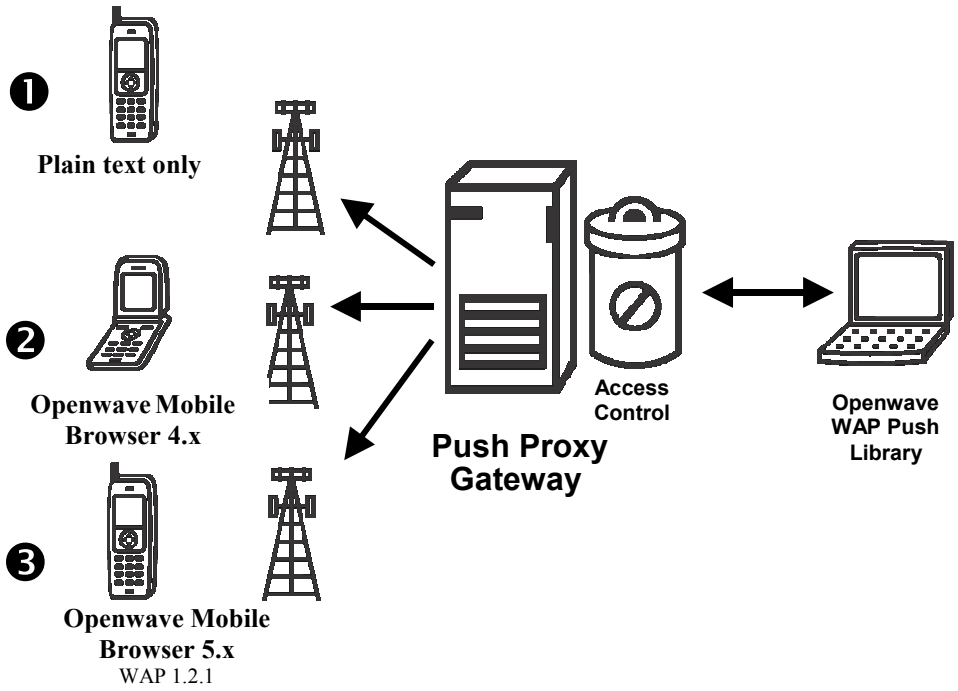
- Alert is sent from the content provider to the Openwave MAG. The alert is received by the Openwave MAG Messenger component.
- Depending on network type the push will be communicated using SMS or over the user's current session. Each alert has a unique identifier. The unique identifier prevents the same alert from being displayed twice.
- The user-agent receives the alert indicator. When the user selects the alert on the browser, the browser will download the alert from the Openwave MAG.

The advantages of using the Openwave UP.Notify push mechanism are:

- Openwave MAG provides content tokenization for efficient over the air transport
- Ordered delivery of push
- The push/pull channel synchronization enables the delivery of push over packet networks and the user's active Session.
- The push content can be delivered directly

WAP Push

Openwave MAG and its Push Proxy Gateway (PPG) component supports the WAP 1.2.1 Push Access Protocol (PAP). Openwave WAP Push Library provides Java APIs that encapsulate PAP, so that



Developers can dramatically simplify the process of creating WAP Push enabled applications.

Openwave PPG uses a combination of the latest provisioning information, the historical transaction data, the session data and the default configuration to determine the best push mechanism for delivery to the target mobile device. For Openwave Mobile Browser 5.0 and higher, as well as any WAP 1.2.1 compliant device, the chosen delivery mechanism is WAP Push.

The PPG is fully compliant with the WAP 1.2.1 PAP Options, WAP Push Content Types and Message specifications. However not all of these options have a direct translation to UP.Notify.

Alerts in WAP Push

In terms of the supported content type the closest analogy to an UP.Notify alert for content and user-agent processing is the Service Indication. The WAP Specification Suite defines the Service Indication content type. The mechanism of submission is also analogous to UP.Notify alert as it uses the HTTP post mechanism. However instead of embedding control information in HTTP headers, the WAP Push mechanism uses the Push Access Protocol, which defines a control entity (an XML document) to be submitted with the content to be sent to the client. The control entity (indicating delivery parameters, priority level, preferred bearer etc.) is not sent to the mobile device, it is stripped out and processed by the PPG.

An example of the Service Indication

```
<?xml version="1.0"?>
<!DOCTYPE si PUBLIC "-//WAPFORUM//DTD SI 1.0//EN"
        "http://www.wapforum.org/DTD/si.dtd">

<si>
    <indication
      href="http://mailservice.openwave.com/email/123/abc.wml"
                  created="1999-06-25T15:23:15Z"
                  si-expires="1999-06-30T00:00:00Z"
                  action="signal-medium">

           Mobile Mail: You have 4 new emails

    </indication>
</si>
```

In addition the following headers (non-exhaustive list) are communicated as part of this push message:

- *Content-Location*
- *Content-Type*
- *Content-Location*
- *x-wap-application-id*

  an identifier to target the alert to the browser or other applications depending on mobile-device support.

- *x-wap-content-uri*

  used a key to the cache in the same manner as x-up-proxy-request-uri

- *x-wap-initiator-uri*

  Identification of the source of the alert, if this header is not present it is assumed to be the same value as the Content-Location header.

The WAP Forum Specification Suite also defines the mechanisms used to transport the Service Indication. This may be over SMS or a WSP Session. The content is transformed into its compact binary representation. Typically on content sizes of the example above the compression ratio is about 60-80% of its textual form.

WAP Push Backward Compatibility

In order to facilitate backward compatibility the Openwave PPG provides a number of transformations for WAP Push defined content types to UP.Notify equivalents.

UP.Notify Conversion

Transformation of PAP to UP.Notify is broken down into two major constituents:

- Control Entity
- Content Entity

There is no equivalent of the capabilities entity in UP.Notify and it will be discarded in the conversion.

Control Entity

The control entity contains address, delivery preference and data pertaining to the initiator of the push message. Not all elements of the control entity have equivalents in UP.Notify and if any of the following elements are included in the control entity for a message targeted to Openwave Mobile Browser 4.x or earlier, the message will be rejected.

PAP Elements/Attributes	Reason
deliver-after-time	No UP.Notify 4.0 equivalent.
ppg-notify-requested-to	No UP.Notify 4.0 equivalent.
progress-notes-requested-to	No UP.Notify 4.0 equivalent.
Network	No UP.Notify 4.0 equivalent. Can be supported if the network-required flag is set false. Can also be supported if the network indicator matches the network for the gateway when the network-required flag is set true.
network-required	No UP.Notify 4.0 equivalent. See above
Bearer	No UP.Notify 4.0 equivalent. Exception in the case of SMS.
bearer-required	No UP.Notify 4.0 equivalent. See above

The following PAP control entity attributes will be mapped to UP.Notify equivalents:

PAP Elements/Attributes	Reason
deliver-before-time	The push-message time can be mapped to time-to-live (X-Up-Ntfn-TTL) header value supported by the UP.Notify 3.0 protocol. TTL can be set to be (deliver-before-time – present time).
product-name	Ignored
push-id	Maps to the X-Up-Ntfn-ID header.
source-reference	Ignored
priority	Generally ignored except in the case of pushes involving Service Indications.
delivery-method	Ignored since Messenger provides a guaranteed method of delivery.

Note: the WAP PAP CANCEL message is converted to a Delete Notification message in UP.Notify.

Content Entity

The content entity may contain WAP defined content types which have equivalence in UP.Notify. The WAP defined content types are:

- Service Indication
- Service Load
- Cache Operation

In the case of Cache Operation it is an almost direct translation to the content type "Application/x-up-cacheop" .

WAP Service Indication

The following is a detailed transformation matrix from the WAP Service Indication (SI) content type to the UP.Notify alert content type, for backward compatibility:

- The content type is changed from text/vnd.wap.si to application/vnd.uplanet.alert.
- Content related HTTP headers will be updated accordingly, as follows:

Add-Push-Ntfn Attributes	Attribute Values
HTTP-Version	"HTTP/1.0"
X-Up-UP.Notify-Version	UP.Notify/4.0
X-Up-Subno	Subscriber ID extracted from a given address-value attribute defined within an address element. The "escaped-value" ABNF rule defined as part of the PAP client address format will contain the subscriber ID value.
X-Up-Ntfn-TTL	Set as in Control Entity section (above)
X-Up-Ntfn-channel	"push"
Content-type	"application/vnd.uplanet.alert"
Content-length	Overall length of the transcoded content, i.e. – the length in bytes

	of the alert.
Content-Base | Content-Location	If these headers where present in original PAP request issued by the Push Initiator then include them in the Add Notification request also. (Content-Location will take precedence)
Entity-body	The alert content body created from the Service Indication.

Additional element equivalents from UP.Notify alert to WAP Service Indication:

Alert Attributes	Attribute Values
Label	#PCDATA specified within the body of the indication element and also (optionally) the contents of the info element if specified.
Href	Value of the href attribute defined within the indication element.
Count	Value of the si-id attribute.
Urgency	Priority defined within the PAP push-message quality-of-service element if specified. This should be defaulted to “medium” otherwise.

Here are the direct mappings that will be done by the PPG for Service Indication values:

SI Attributes	Translated UP.Notify Alert Values
action="signal-low"	urgency="low"
action="signal-none"	urgency="low"
action="signal-medium"	urgency="medium"
action="signal-high"	urgency="high"
action="delete"	this message will be converted into a delete-from-inbox message. Note that if this action type is in a multipart message (i.e. it is part of a multipart message), translation will not occur because messenger does not support delete-from-inbox actions in a multipart.
none specified	urgency="medium" will be used

Any remaining attributes/elements specified within the WAP Service Indication will be lost in the conversion to the UP.Notify equivalent.

Note: Within the alert DTD the *label* and *href* attributes are required. Therefore, if no value can be extracted for either of these from the service indication content then the content transformation will fail and Openwave PPG will respond to the Push Initiator with a status code and description of `3006, Transformation Failure`.

WAP Service Load

The WAP service load (SL) is converted into an UP.Notify alert content type using the PULL notification channel.

- The content type is changed from text/vnd.wap.sl to application/vnd.uplanet.alert.
- Content related HTTP headers will be updated accordingly, as follows:

The following tables detail the exact mappings which take place.

Add-Push-Ntfn Attributes	Attribute Values
HTTP-Version	"HTTP/1.0"
X-Up-UP.Notify-Version	UP.Notify/4.0
X-Up-Subno	Subscriber ID extracted from a given address-value attribute defined within an address element. The "escaped-value" ABNF rule defined as part of the PAP client address format will contain the subscriber ID value.
X-Up-Ntfn-TTL	Set as in Control Entity section (above)
X-Up-Ntfn-channel	"pull"
Content-type	"application/vnd.uplanet.alert"
Content-length	Overall length of the transcoded content, i.e. – the length in bytes of the alert.
Content-Base | Content-Location	If these headers where present in original PAP request issued by the Push Initiator then include them in the Add Notification request also. (Content-Location will take precedence)
Entity-body	The alert content body created from the Service Loading content.

In terms of the displayed alert values the following conversion takes place:

Alert Attributes	Attribute Values
Label	Attribute must be specified although no value within the Service Loading content directly maps. An application label is inserted e.g. – "Push Message"
Href	Value of the href attribute defined within the indication element.

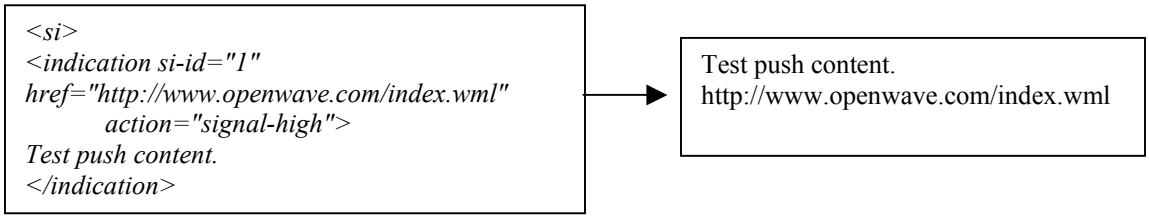
In terms of priority the following conversion takes place:

SL Attributes	Translated UP.Notify Alert Values
action="cache"	urgency="low"
action="execute-low"	urgency="medium"
action="execute-high"	urgency="high"
None	urgency="medium" will be used

Plain Text Conversion

In terms of content, the following translation will be done from WAP Service Indication to plain text SMS since converting other WAP Push content types does not make sense. The text label and HREF are preserved.

An example of this is:



Note: In certain cases, on certain types of device the HREF, if present, will be an active link. In the case of SMS the text keywords in the message can be active links too, dependent on the access policy of the service provider.

The content type *text/plain* will be translated to a plain text SMS if the phone is a non-WAP Push enabled device.

WAP Push Example

A full example of a WAP Push submission

```
--asdlfkjiurwghasf
Content-Type: application/xml; charset=UTF-8
<?xml version="1.0"?>
      <!DOCTYPE pap PUBLIC "-//WAPFORUM//DTD PAP 1.0//EN"
            "http://www.wapforum.org/DTD/pap_1.0.dtd">

<pap product-name="PI Push Submission Test File 1 (Service
Indication)">
      <push-message     push-id="Test Push 1 (SI)"
      source-reference="PDC Push Team"
      ppg-notify-requested-to= "http:Result_Notification_Server"
                  progress-notes-requested="false">
            <address address-value="WAPPUSH=Address
/TYPE=USER@ppg.openwave.com"/>
            <quality-of-service     priority="high"
            delivery-method="confirmed"/>
      </push-message>
</pap>

--asdlfkjiurwghasf
Content-type: text/vnd.wap.si; charset=UTF-8

<?xml version="1.0"?>
<!DOCTYPE si PUBLIC "-//WAPFORUM//DTD SI 1.0//EN"
      "http://www.wapforum.org/DTD/si.dtd">
<si>
<indication href = "
href="http://www.mailservice.com/email/123/abc.wml"">
Mobile Mail: You have new mail.
</indication>
</si>

--asdlfkjiurwghasf
Content-Type: application/xml

<?xml version="1.0"?>

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:prf="http://www.wapforum.org/UAPROF/ccppschema19991014#">

    <rdf:Description>
        <prf:BrowserName>UP.Browser</prf:BrowserName>
        <prf:BrowserVersion>5.0</prf:BrowserVersion>
        <prf:OSName>UNIX</prf:OSName>
        <prf:OSVersion>1.0</prf:OSVersion>
    </rdf:Description>

</rdf:RDF>

--asdlfkjiurwghasf--
```

Summary

Control entity translations from WAP Push to UP.Notify may be summarized at a high level as:

- WAP Service Indications are converted to UP.Notify alerts and sent to the device on the ‘PUSH’ notification channel.
- WAP Service Loads are converted to UP.Notify alerts and sent to the device on the ‘PULL’ notification channel, a default text label is added.
- WAP Cache Operations are converted to UP.Notify Cache Operations and sent to the device on the ‘PUSH’ notification channel.
- WAP Service Indications may be converted to plain text also
- WAP Service Loads and Cache Operations cannot be converted to plain text, as there is no equivalent.
- The Push Access Control Entity is also converted to equivalents in UP.Notify but there are certain attributes, which will cause the message to be rejected.
- There is an equivalent mapping for PAP Cancel in UP.Notify

Application Migration

When migrating applications from UP.Notify to WAP Push, there are a number of issues to consider:.

- It is still possible to push UP.Notify alerts via Openwave MAG to Openwave Mobile Browser 5.0 and higher.
- It is not possible to push UP.Notify alerts to WAP 1.2.1 Push enabled devices not running Openwave Mobile Browser.

However there are a number of additional advantages to migrating applications from UP.Notify to WAP Push:

- WAP Push is an open standard for both developers and device vendors and Openwave supports these standards fully.
- Openwave MAG supports WAP Service Indication, Service Load, CacheOp and the pushing of WML directly to the device.
- WAP Push supports user addressing by subscriber identifier and phone number.
- WAP Push supports multi-recipient addressing
- WAP Push supports Application Addressing using the WINA registered push application identifiers.
- Openwave PPG supports transformation of WAP Push to UP.Notify for earlier versions of Openwave Mobile Browser, thus ensuring maximum compatibility to an existing subscriber community.
- WAP Push Service Indication provides for the removal of duplicate alert messages using the HREF as well as Service indication identifier, thus enhancing the user experience and easing application development.
- The WAP Push interface allows for a greater degree of control over how the message is processed by the PPG (e.g. store & forward, priority level, bearer choice, confirmation, quality of service)
- WAP Push provides for status query and alert cancellation; this is similar to the status and delete operations offered by UP.Notify.
- WAP Push allows for result notification to be sent out, detailing the status of the delivery in manner similar to UP.Notify.
- Openwave MAG allows for multipart push content to be pushed to the mobile device
- WAP Push supports the ability to query a client's capabilities so that pushes may be targeted to specific device types.

There are compelling reasons to migrate applications from UP.Notify to WAP Push, including open standards support, strategic placement of investment, and a richer feature set. The Mobile Access Gateway and its Pus Proxy component support all of these features.

Author
Fergus Wills
Product Technologist
Openwave Systems Inc.

Feedback
whitepaper.feedback@openwave.com

Openwave™ Systems Inc.
1400 Seaport Boulevard
Redwood City
California 94063
U.S.A.
Corporate +1 650 480 8000
Europe +44 1442 458 800
Japan +81 3 5909 6100
http://www.openwave.com