WAP Service Indication

Version 08-Nov-1999

Wireless Application Protocol
Service Indication Specification

Contents

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*" [WAP].

This specification defines the Service Indication (SI) content type, which is an application of the Extensible Markup Language (XML) 1.0 [XML]. The content type provides a means to notify a client that an external asynchronous event has occurred and indicate a service that can be loaded in order to react to that event. This is accomplished by sending a message to the client that informs the end-user about the event, and a URI from where the appropriate service can be loaded. For example, the message could read "You have new voice mails", and the URI points to the voice mail service.

The SI content type does also allow the level of user-intrusiveness to be controlled, deletion of SIs stored on a client (both manually and automatically by using the concept of expiration), and replacement of stored SIs. A mechanism to resolve race conditions is also specified. Finally, WBXML [WBXML] tokens are defined to allow for efficient over-the-air transmission.

2. Document Status

This document is available online in the following formats:

- PDF format at http://www.wapforum.org/.

2.1 Copyright Notice

2.2 Errata

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

2.3 Comments

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

3. References

3.1 Normative references

[HTML4] "HTML 4.0 Specification, W3C Recommendation, revised on 24-Apr-1998", D. Raggett et al., April 24 1998. URL: http://www.w3.org/TR/1998/REC-html40-19980424

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

[WAE] “Wireless Application Environment Overview”, WAP Forum, 04-Nov-1999, URL: http://www.wapforum.org/

[WBXML] “WAP Binary XML Content Format”, WAP Forum, 04-Nov-1999, URL: http://www.wapforum.org/

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

[WML] “Wireless Markup Language”, WAP Forum, 04-Nov-1999, URL: http://www.wapforum.org/

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

3.2 Informative references

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

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

[PushOTA] “WAP Push OTA Specification”, WAP Forum, 08-Nov-1999 URL: http://www.wapforum.org/

[PushPAP] “WAP Push Access Protocol Specification”, WAP Forum, 08-Nov-1999 URL: http://www.wapforum.org/

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

[WAP] “Wireless Application Protocol Architecture Specification”, WAP Forum, 30-Apr-1998 URL: http://www.wapforum.org/

4. Definitions and Abbreviations

4.1 Definitions

The following are terms and conventions used throughout this specification.

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 by [RFC2119].

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.

Context – an execution space where variables, state and content are handled within a well-defined boundary.

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.

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).

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*

4.2 Abbreviations

For the purposes of this specification, the following abbreviations apply.

CPI	Capability and Preference Information
DNS	Domain Name Server
DTD	Document Type Definition
HTTP	Hypertext Transfer Protocol
IANA	Internet Assigned Numbers Authority
IP	Internet Protocol
OTA	Over The Air
PAP	Push Access Protocol
PI	Push Initiator
PPG	Push Proxy Gateway
QOS	Quality of Service
RDF	Resource Description Framework
RFC	Request For Comments
SGML	Standard Generalized Markup Language
SI	Service Indication
SIA	Session Initiation Application
SIR	Session Initiation Request
SL	Service Loading
SSL	Secure Socket Layer
TLS	Transport Layer Security
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTC	Universal Time Co-ordinated
WAP	Wireless Application Protocol
WDP	Wireless Datagram Protocol
WSP	Wireless Session Protocol
WBXML	WAP Binary XML
WINA	WAP Interim Naming Authority
WTLS	Wireless Transport Layer Security
XML	Extensible Mark-up Language

5. Introduction

The Service Indication (SI) content 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.

The example below illustrates the procedure and one possible user interface:

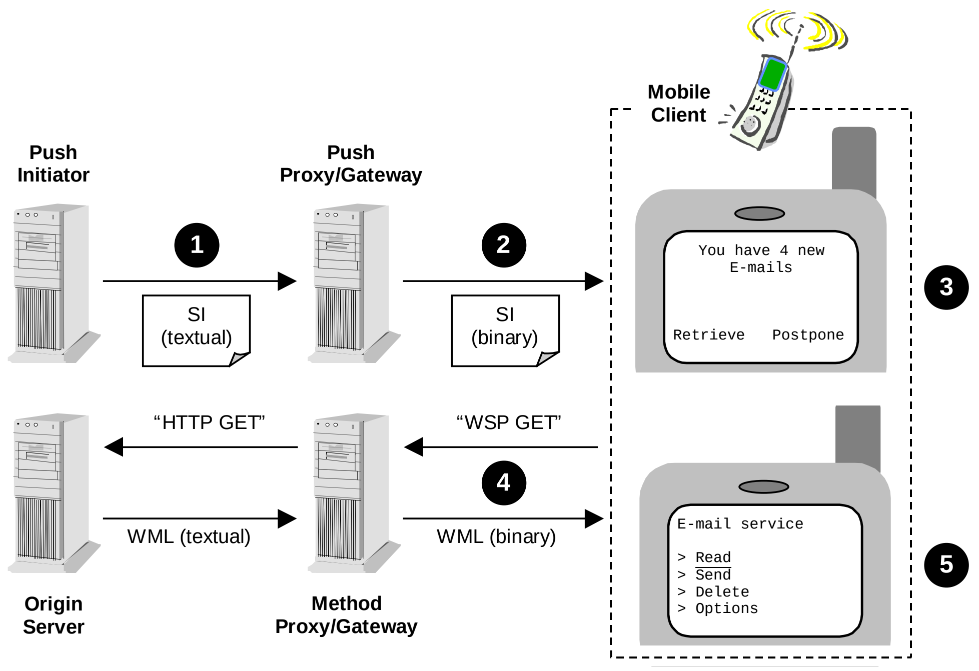

Figure 1 : Service Indication - the basic concept

The example illustrates how an end-user is notified about new e-mails and how the appropriate service (in the form of a WML deck) is started. The following steps are involved:

1. The Push Initiator, in this case the e-mail provider, instructs the Push Proxy/Gateway to push an SI to the mobile client using the Push Access Protocol [PushPAP]. The Push Initiator provides the SI with an appropriate message and a URI to the e-mail service.
2. The Push Proxy/Gateway sends the SI to the mobile client using the Push OTA Protocol [OTA].
3. The mobile client receives the push containing the SI, and the message is presented to the end-user. The client provides the end-user with a means to choose whether the e-mail service should be started immediately, or if the SI should be postponed. In this example, the end-user chooses to start the e-mail service immediately.
4. The e-mail service indicated by the SI's URI is retrieved ("pulled") from the origin server via the Method Proxy/ Gateway or optionally from the client's cache memory.
5. The e-mail service starts executing on the mobile client.

In addition to the basic functionality described above, the SI content type also provides various mechanisms to improve the end-user experience. These include:

- **User-intrusiveness levels**
  It is possible to assign different levels of user-intrusiveness to SIs in order to affect the client's behaviour when a SI is about to be presented to the end-user.
- **Deletion**
  The service provider can delete SIs that become invalid for some reason (e.g. an e-mail notification becomes invalid if the e-mails have been read using other means than the mobile client). This is accomplished by issuing a special SI to delete the now invalid SI.
- **Replacement**
  In most cases, it is of no use to store multiple SIs indicating the same service on a client (e.g. one SI saying that there is one new e-mail and then another one saying that there are two new e-mails, both indicating the same e-mail service). This is avoided by providing a means to replace an old SI with a new one.
- **Handling of out of order delivery**
  Due to the unpredictable availability of a wireless client, it can not be ensured that content always is delivered in the same order as it was sent (race conditions). It would be inappropriate to apply rules for replacement of SIs in such case (e.g. one SI saying that there is one new e-mail can arrive after one saying that there are two new e-mails, both indicating the same e-mail service). This is avoided by silently discarding a received SI if it is older than any similar SI stored on the client.
- **Expiration**
  The service indicated by an SI is in many cases only valid for a certain amount of time (e.g. voice mails are usually automatically deleted after a couple of days), and hence the SI will indicate void content after that time. This is addressed by allowing the author of an SI to specify the date and time when an SI should expire, i.e. be automatically deleted from a client.

6. The Service Indication Content Format

This section defines the content format used to represent the Service Indication (SI), which is an application of XML version 1.0 [XML]. The complete Service Indication DTD, which an implementation conforming to this specification MUST support, is defined in chapter 8.

6.1 Service Indication Character Set

The SI content type MUST use the same character set rules as specified in [WML], except the rules for meta-information placed within the content, since such information is not supported in SI.

6.2 The SI Element

```
<!ELEMENT si(indication,info?)>
```

The `si` element specifies two elements that describe a Service Indication.

6.2.1 The Indication Element

```
<!ELEMENT indication (#PCDATA)>
<!ATTLIST indication
  href        %URI;                              #IMPLIED
  si-id       CDATA                              #IMPLIED
  created     %Datetime;                         #IMPLIED
  si-expires  %Datetime;                         #IMPLIED
  action      (signal-none|signal-low|
               signal-medium|signal-high|delete)  “signal-medium”
>
```

Attributes

`href=%URI`

This attribute specifies the URI that is used to access the service. If `href` is empty, or omitted, the SI corresponds to a notification (no service can be initiated).

`si-id=CDATA`

This attribute may provide the SI with an identity in order to make it possible to distinguish between different SIs. If this attribute is not specified, its value is considered to be the same as the value of the `href` attribute.

In order avoid conflicts between SIs, it is RECOMMENDED that content developers use an address (e.g. URL) within their control combined with an identifier for the SI as the value for `si-id` (for example: `“www.wapforum.org/siid/123”` or `“123@siid.wapforum.org”`).

`created=%Datetime`

This attribute may be used to specify the date and time associated with the creation or last modification of the content indicated by `href`, which may differ from the date and time when the SI was created.

If used, the attribute value MUST be expressed in a date/time representation based on [ISO8601] as specified in [HTML4]. However, SI does not allow use of time zones; the time MUST always be expressed in Co-ordinated Universal Time (UTC), a 24-hour timekeeping system (indicated by the "`Z`"). The format is:

```
YYYY-MM-DDThh:mm:ssZ
```

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

Note: "`T`" and "`Z`" appear literally in the string.

Example: "`1999-04-30T06:40:00Z`" means 6.40 in the morning UTC on the 30th of April 1999.

`si-expires=%Datetime`

This attribute may be used to specify the date and time when the SI expires and thereby be automatically deleted or marked as "expired". If this attribute is not specified, the SI never expires and is thus not subject to automatic deletion.

The representation for this attribute value is the same as the one used for the `created` attribute.

`action=(signal-none|signal-low|signal-medium|signal-high|delete)`

This attribute may contain a text string specifying the action to be taken when the SI is received.

Attribute value	Description reference
`signal-none`	Section 7.2.
`signal-low`	Section 7.2.
`signal-medium`	Section 7.2.
`signal-high`	Section 7.2.
`delete`	Section 7.2.

If the attribute is not specified, the value "`signal-medium`" is used.

6.2.2 The Info Element

```
<!ELEMENT info (item+)>

<!ELEMENT item (#PCDATA)>
<!ATTLIST item
  class        NMTOKEN      #REQUIRED
>
```

The `info` element provides a means to specify additional information not provided by the attributes of the `indication` element. The element contains one or more `item` elements that specify the additional information. Each `item` element contains a `class` attribute describing what information the content of the `item` element contains. How a client uses this information is not specified within this specification and is thus implementation dependent. A client MAY discard the `info` element.

Values for the `class` attribute SHOULD be registered with the WINA registrar [WINA] (class name, intended usage, intended user-agent, and attribute value prefix WBXML token).

Attributes

`class=NMTOKEN`

This attribute specifies the name of the class, i.e. the kind of information carried in the `item` element.

7. Semantics

7.1 Introduction

When a client receives an SI it must process it in order to determine what action(s) should be carried out. Both actions that the end-user will become explicitly aware of and actions that the end-user will not become explicitly aware of are carried out.

The actions that the end-user does not become explicitly aware of are all carried out upon reception, possibly before the SI is made known to the end-user, and include deletion, expiration handling, resolving of race conditions, and replacement. These actions are treated in section 7.2 (Reception).

Handling of information specified in the `info` element MAY result in actions that the end-user becomes aware of. This is also described in section 7.2 (Reception).

There is one possible outcome of an SI that the end-user will become explicitly aware of; the SI is presented to the end-user as described in section 7.3 (Presentation). The presentation of an SI may, after end-user interaction, result in a service being loaded, which is described in section 7.4 (Service Invocation).

Expiration handling is not only carried out upon reception of an SI as mentioned above; it is also used for invalidating SIs stored on the client. This is treated in section 7.5 (Expiration).

7.2 Reception

The white boxes in the figure below show the steps involved in processing an SI upon reception, none of which is made explicitly known to the end-user. This processing should be carried out as soon as possible upon reception. However, a client MUST NOT start the processing if it currently is presenting another SI to the end-user as described in section 7.3.

The grey box show actions that the end-user MAY become aware of, while the black box shows actions that are made known to the end-user, as described in section 7.3.

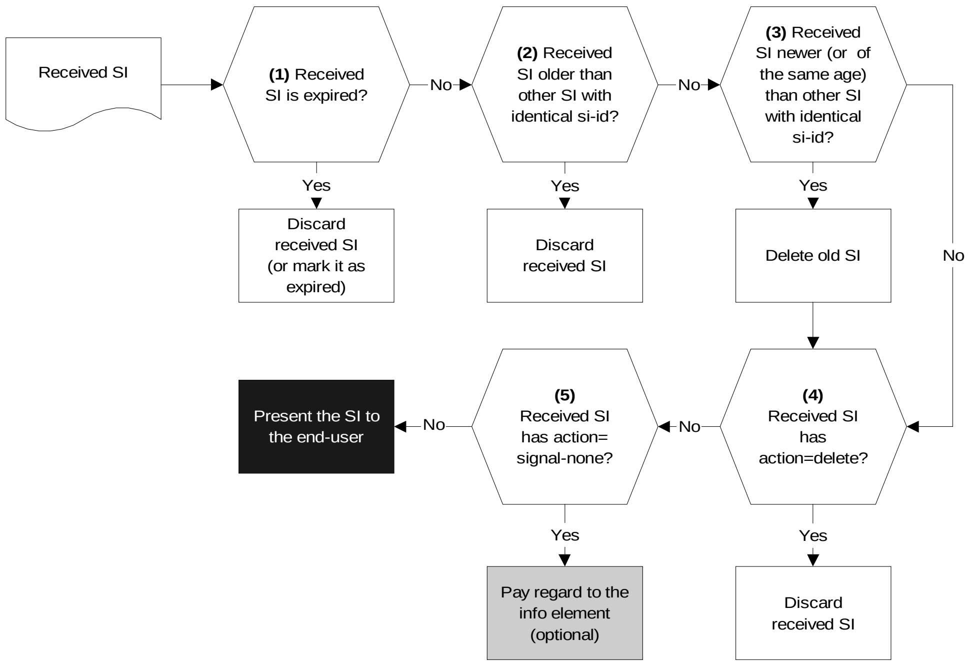
Received SI
(1) Received SI is expired?
No
(2) Received SI older than other SI with identical si-id?
No
(3) Received SI newer (or of the same age) than other SI with identical si-id?
Yes
Discard received SI (or mark it as expired)
Yes
Discard received SI
Yes
Delete old SI
No
Present the SI to the end-user
No
(5) Received SI has action= signal-none?
No
(4) Received SI has action=delete?
Yes
Pay regard to the info element (optional)
Yes
Discard received SI

Figure 2 : Steps involved in processing a received SI

The numbered items in the figure above constitute the steps involved in processing a received SI that need to be carried out, before the SI possibly is made known to the end-user, each explained below:

1. **Expiration**

   A client capable of determining the time MUST use the `si-expires` attribute to determine if the received SI is expired. If so, the received SI MUST be silently discarded or ignored, or marked as "expired" by means provided by the client.

   If a client does not have a real-time clock, or is not by other means able to determine the time, this operation MUST NOT be carried out.

2. **Handling of out of order delivery**

   The `created` attribute is used to determine the age of the SI in order to resolve race conditions between SIs that arrive in an order different from the one in which the push initiator sent them. If the value of the `created` attribute in a received SI is older than this attribute value in any other SI with identical `si-id`, the received SI MUST be silently discarded.

   SIs are subject to this operation only if both criteria stated below are fulfilled:

   - the `created` attribute has an explicitly assigned value
   - the `si-id` attribute or the `href` attribute, or both, has an explicitly assigned value

3. **Replacement**

   The `created` attribute is used to determine the age of an SI, which combined with the value of the `si-id` attribute provides the information needed for replacement. A received SI containing an `si-id` identical to the `si-id` of any older SI MUST replace the old SI. If the received SI and another SI with identical `si-id` contain identical values for the `created` attribute, the received SI MUST be silently discarded.

   SIs are subject to this operation only if both criteria stated below are fulfilled:

   - the `created` attribute has an explicitly assigned value
   - the `si-id` attribute or the `href` attribute, or both, has an explicitly assigned value

4. **Deletion**

   If the received SI's `action` attribute equals "`delete`", both the received SI and any other (if any) SI with identical `si-id` MUST be deleted. An SI with the `action` attribute set to "`delete`" MUST have an explicitly assigned value for `si-id`.

   In the figure above, this procedure is equal to discarding the received SI since any other SI with identical `si-id` is deleted in step 3.

5. **Handling of information specified in the info element**

   If the received SI's `action` attribute equals "`signal-none`", the SI MUST NOT:

   - be presented (see section 7.3) to the end-user
   - be postponed (see section 7.3.1)
   - imply that a service is being executed without user intervention

   However, the client MAY use the information carried in the `info` element to perform certain tasks. For information about how the `info` element is used, see section 6.2.2.

   Note! The client MAY also make use of the information carried in the `info` element even if the SI's `action` attribute does not equal "`signal-none`". If so, this SHOULD be done between steps 3 and 4 in Figure 2.

7.2.1 Reception of Multiple Service Indications

A client receiving SIs that for some reason are not processed as described in section 7.2 upon reception (e.g. if the client is busy with other activities) MUST treat them in the order described below when they are about to be processed:

1. Process all received SIs according to step 1-4 in Figure 1, section 7.2 (white boxes).
2. Process all SIs with `action=signal-none` (step 5) in the order in which they were received (OPTIONAL).
3. Present the remaining SIs to the user, sorted by:
    1. The `action` attribute value, starting with "`signal-high`", then "`signal-medium`", and finally "`signal-low`".
    2. The order in which they were received.

A client MUST be able to maintain an implementation dependent number of SIs that are not processed upon reception. The number MUST be greater or equal to one, but a value below three is NOT RECOMMENDED. A RECOMMENDED minimum storage space for each of these SIs is 500 octets.

7.3 Presentation

When an SI is presented, text specified in the `indication` element MUST be made known to the end-user. Further, the end-user MUST be provided with the opportunity to load the service indicated by the `href` attribute immediately, or postpone the SI for later handling. Postponement of SIs is treated in section 7.3.1.

When an SI is about to be presented, the client MUST treat it according to the value of the `action` attribute, which expresses one of the following actions ("`delete`" or "`signal-none`" are not possible values when a SI is about to be presented):

- `signal-low`
- `signal-medium`
- `signal-high`

This specification does not mandate any exact behaviour for these different `action` attribute values since clients are assumed to have varying capabilities. For example, if a very "thin" client should present an SI immediately, it might need to interrupt an executing service before the received SI is presented. A more powerful client might on the other hand be able to present the received SI, possibly load the indicated service, and then return to the previous executing service.

The following rules MUST however be followed:

- `action`= "`signal-low`"

  The SI MUST be postponed without user intervention (see section 7.3.1for information about postponing SIs).

- `action`= "`signal-medium`"

  The SI MUST be presented as soon as the implementation allows that to be carried out in a non-user-intrusive manner.

- `action`= "`signal-high`"

  The SI MUST be presented as soon as the implementation allows that to be carried out in a non-user-intrusive manner, or earlier if considered appropriate (which MAY result in a user-intrusive behaviour). This decision can either be based on user preference settings or be carried out at the discretion of the implementation.

It is RECOMMENDED that push initiators use `action`="`signal-medium`" (default value) to the fullest possible extent.

It is also RECOMMENDED that a client provide the end-user with a means to associate different `action` attribute values mentioned above with various logical indicators (beep, melody, flashing light, vibration etc).

7.3.1 Postponement

When an SI is postponed, it is stored in the client for later handling. Since the possibility to postpone several SIs exists, the client MUST provide the end-user with a means to act upon them (load the service indicated by the URI provided by the SI's `href` attribute into the intended user agent) at a later time. How this is accomplished is implementation dependent, but it is RECOMMENDED that the user can make selections based on the text specified in the `indication` element. The end-user SHOULD also be given the possibility to delete any postponed SI by means provided by the client.

A client MUST be able to maintain an implementation dependent number of postponed SIs. The number MUST be greater or equal to one, but a value below ten is NOT RECOMMENDED. A RECOMMENDED minimum storage space for each of these SIs is 500 octets. It is implementation dependent how a client handles an SI that is about to be postponed, when this causes the maximum number of postponed SIs to be exceeded.

7.4 Service Invocation

When an SI is presented to the end-user, he or she may choose to load the service indicated by the SI. If so, the service identified by the URI provided by the SI's `href` attribute MUST be loaded in the same way as the user agent otherwise performs method requests initiated by the end-user [WAE]. This implies that service content is fetched either from an origin server or from the client's cache memory, if available. Once the method request is successfully completed, the user agent loads the service into a clean user agent context and executes it.

The client SHOULD somehow indicate to the end-user that service loading is in progress and provide the user with the possibility to abort that loading. If the end-user aborts a service that is being loaded, the SI should be postponed automatically.

It is implementation specific whether an SI should be automatically deleted from the client after the service has been executed, or if it should be stored on the client so the end-user can load the service at a later point of time as well.

7.5 Expiration

A client capable of determining the time MUST apply the following rules:

- After the date and time expressed by the `si-expires` attribute, the SI is said to be expired and MUST either be:
  - deleted from the client, or
  - marked as "expired" by means provided by the client
- A client receiving an already expired SI MUST either:
  - silently discard or ignore that SI as described in section 7.2, or
  - process the SI as described in section 7.2, but mark the SI as "expired" by means provided by the client

If a client does not have a real-time clock, or is not by other means able to determine the time, these operations MUST NOT be carried out.

8. SI Reference Information

Service Indication (SI) is an application of [XML] version 1.0.

8.1 Document Identifiers

8.1.1 SGML Public Identifier

Editor's note: This identifier has not yet been registered with the IANA or ISO 9070 registrar

```
-//WAPFORUM//DTD SI 1.0//EN
```

8.1.2 SI Media Type

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

Textual form:

```
text/vnd.wap.si
```

Tokenised form:

```
application/vnd.wap.sic
```

8.2 Document Type Definition (DTD)

```
<!--
Service Indication (SI) Document Type Definition.
SI is an XML language.  Typical usage:
   <?xml version="1.0"?>
   <!DOCTYPE si PUBLIC "-//WAPFORUM//DTD SI 1.0//EN"
          "http://www.wapforum.org/DTD/si.dtd">
   <si>
   ...
   </si>
-->


<!ENTITY % Datetime “CDATA”>           <!-- ISO date and time -->
<!ENTITY % URI      “CDATA”>           <!-- URI designating a
                                            hypertext node    -->


<!--===================== The SI Element =====================-->
<!ELEMENT si(indication,info?)>


<!--================= The indication Element =================-->
<!ELEMENT indication (#PCDATA)>
<!ATTLIST indication
  href        %URI;                           #IMPLIED
  si-id       CDATA                           #IMPLIED
  created     %Datetime;                      #IMPLIED
  si-expires  %Datetime;                      #IMPLIED
  action      (signal-none|signal-low|
               signal-medium|signal-high|delete)  “signal-medium”
>


<!--==================== The INFO Element ====================-->
<!ELEMENT info (item+)>

<!ELEMENT item (#PCDATA)>
<!ATTLIST item
  class        NMTOKEN                        #REQUIRED
>
```

9. A Compact Binary Representation of Service Indication

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

9.1 Extension Tokens

9.1.1 Tag Tokens

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

9.1.2 Attribute Tokens

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

9.2 Encoding Semantics

9.2.1 Document Validation

XML document validation (see [XML]) SHOULD occur during the process of tokenising an SI and, if done, it MUST be based on the DOCTYPE declared in the SI. When validating the source text, the tokenisation process MUST accept any DOCTYPE or public identifier, if the document is identified as an SI media type (see section 8.1.2).

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

9.2.1.1 Validate %Datetime;

The WML tokenisation process SHOULD validate that attribute values defined as `%Datetime;` follow the syntax defined in section 6.2.1.

9.2.2 Encoding of %Datetime;

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

For example, “1999-04-30T06:40:00Z” is encoded into six octets as follows:

Number	“1”	“9”	“9”	“9”	“0”	“4”	“3”	“0”	“0”	“6”	“4”	“0”	“0”	“0”
Binary value	0001	1001	1001	1001	0000	0100	0011	0000	0000	0110	0100	0000	0000	0000
Octet (hex)	00011001 (19)		10011001 (99)		00000100 (04)		00110000 (30)		00000110 (06)		01000000 (40)		omitted	

9.3 Numeric Constants

9.3.1 Tag Tokens

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

Tag Name	***Token***
si	5
indication	6
info	7
item	8

9.3.2 Attribute Start Tokens

The following token codes represent the start of an attribute in code page zero (0). All numbers are in hexadecimal.

Attribute Name	***Attribute Value Prefix***	***Token***
action	signal-none	5
action	signal-low	6
action	signal-medium	7
action	signal-high	8
action	delete	9
created		A
href		B
href	http://	C
href	http://www.	D
href	https://	E
href	https://www.	F
si-expires		10
si-id		11
class		12

Tokens B0-FF are reserved to be used as attribute value prefixes for the `class` attribute (registered with the WINA registrar [WINA]).

9.3.3 Attribute Value Tokens

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

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

10. Example

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

```
<?xml version="1.0"?>
     <!DOCTYPE si PUBLIC "-//WAPFORUM//DTD SI 1.0//EN"
                  "http://www.wapforum.org/DTD/si.dtd">
<si>
     <indication  href=”http://www.xyz.com/email/123/abc.wml”
                  created=”1999-06-25T15.23.15Z”
                  si-expires=”1999-06-30T00.00.00Z”>
                  You have 4 new e-mails
     </indication>
</si>
```

The tokenised form of the example above (numbers in hexadecimal), using the WBXML encoding defined in section 9, is found below. This example assumes an UTF-8 character encoding and NULL terminated strings.

In this example the textual SL consists of 269 octets, while the encoded form consists of 76 octets.

```
00  05  6A  00  45  C6  0D  03  'x' 'y' 'z' 00  85  03  'e' 'm'
'a' 'i' 'l' '/' '1' '2' '3' '/' 'a' 'b' 'c' '.' 'w' 'm' 'l' 00
0A  C3  07  19  99  06  25  15  23  15  10  C3  04  19  99  06
30  01  03  'Y' 'o' 'u' ' ' 'h' 'a' 'v' 'e' ' ' '4' ' ' 'n' 'e'
'w' ' ' 'e' '-' 'm' 'a' 'i' 'l' 's' 00  01  01
```

In an expanded and annotated form:

Token Stream	***Description***
`00`	Version number - WBXML version 1.0
`05`	SI 1.0 Public Identifier
`6A`	Charset=UTF-8 (MIBEnum 106)
`00`	String table length
`45`	`si`, with content
`C6`	`indication`, with content and attributes
`0D`	`href=”http://www.”`
`03`	Inline string follows
`‘x’, ‘y’, ‘z’, 00`	String
`85`	`“.com/”`
`03`	Inline string follows
`‘e’, ‘m’, ‘a’, ‘i’, ‘l’, ‘/’, ‘1’, ‘2’, ‘3’, ‘/’, ‘a’, ‘b’, ‘c’, ‘.’, ‘w’, ‘m’, ‘l’, 00`	String
`0A`	`created=`
`C3`	`OPAQUE` data follows
`07`	Length field (7 bytes)
`19, 99, 06, 25, 15, 23, 15`	Data
`10`	`si-expires=`
`C3`	`OPAQUE` data follows
`04`	Length field (4 bytes)
`19, 99, 06, 30`	Data
`01`	`END` (of `indication` attribute list)
`03`	Inline string follows
`‘Y’, ‘o’, ‘u’, ‘ ’, ‘h’, ‘a’, ‘v’, ‘e’, ‘ ’, ‘4’, ‘ ’, ‘n’, ‘e’, ‘w’, ‘ ’, ‘e’, ‘-’, ‘m’, ‘a’, ‘i’, ‘l’, ‘s’, 00`	String
`01`	`END` (of `indication` element)
`01`	`END` (of `si` element)

11. Static Conformance Requirements

This static conformance clause defines a minimum set of features that should be implemented to support Service Indication. A feature can be optional (O), mandatory (M) or conditional (C). Whether a feature applies or not is in some cases dependent on the status of another item, e.g. the SI_SEM_C002 feature is only applicable, and mandatory (status X is equal to status X:M), only if CLOCK is supported. If optional features have labels (O.<n>), support of at least one in the group of options labelled by the same numeral is required.

11.1 Client Features

11.1.1 Predicates

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

Item	Functionality	Reference	Status
CLOCK	The client is able to determine the time.	-	O

11.1.2 Character Set and Encoding

Item	Functionality	Reference	Status
SI_CSE_001	UTF-8 Encoding.	6.1, [WML]	O
SI_CSE_002	UTF-16 Encoding.	6.1, [WML]	O
SI_CSE_003	UCS-4 Encoding.	6.1, [WML]	O
SI_CSE_004	Other character encoding.	6.1, [WML]	O
SI_CSE_005	Reference processing (no meta-information).	6.1, [WML]	M
SI_CSE_006	Character entities.	6.1, [WML]	M

11.1.3 Content Format and Tokenisation

Item	Functionality	Reference	Status
SI_CF_001	Support for the SI DTD.	8	M
SI_CF_002	Support for class(es) in the INFO element.	6.2.2	O
SI_CF_003	Support for SI in textual form (text/vnd.wap.si).	8	O
SI_CF_004	Support for SI in tokenised form (application/vnd.wap.sic).	9	M
SI_CF_005	Syntactical check of attribute values defined as `%Datetime;`.	9.3.1.1	O
SI_CF_006	Support for `%Datetime;` encoded as OPAQUE data.	9.3.2	SI_CF_004:M

11.1.4 Semantics

Item	Functionality	Reference	Status
SI_SEM_001	A received SI is not processed if another SI is currently being presented to the user.	7.2	M
SI_SEM_002	Handling of SIs that are expired.	7.2, 7.5	CLOCK
SI_SEM_003	Handling of SIs that arrive out of order.	7.2	M
SI_SEM_004	Replacement of SIs.	7.2	M
SI_SEM_005	Deletion of SIs.	7.2	M
SI_SEM_006	Handling of information specified in the `info` element.	7.2	O
SI_SEM_007	Handling of one or multiple SIs that are not processed upon reception.	7.2.1	M
SI_SEM_008	Ability to maintain at least one SI that can not be processed directly upon reception.	7.2.1	M
SI_SEM_009	The text specified in the `indication` element is made known to the end-user when an SI is presented.	7.3	M
SI_SEM_010	When a SI is presented, the end-user can choose to load the indicated service immediately, or postpone the SI for later handling.	7.3	M
SI_SEM_011	Handle the SI according to its `action` attribute value.	7.2, 7.3	M
SI_SEM_012	Different `action` attribute values can be associated with logical indicators.	7.3	O
SI_SEM_013	The client provides the end-user with the ability to act upon postponed SIs.	7.3.1	M
SI_SEM_014	Ability to maintain at least one postponed SIs.	7.3.1	M
SI_SEM_015	When the end-user chooses to load the service indicated by the SI, that service is loaded in the same way as the user agent otherwise performs method requests initiated by the end-user.	7.4	M
SI_SEM_016	The end-user is given the possibility to abort a service that is being loaded.	7.4	O

11.2 Push Proxy Gateway Features

11.2.1 General

Item	**Functionality**	**Reference**	**Status**
SI_PPG_001	Support for SI in textual form (text/vnd.wap.si).	8	M
SI_PPG_002	Support for encoding an SI into tokenised form (application/vnd.wap.sic).	9	M
SI_PPG_003	Support for the SI token table.	9.4	M

11.2.2 Validation

Item	**Functionality**	**Reference**	**Status**
SI_VAL_001	XML well-formed.	9.3.1	M
SI_VAL_002	XML validation.	9.3.1	O
SI_VAL_003	SI validation.	SCR-11.2.3	O

11.2.3 SI Document

Item	**Functionality**	**Reference**	**Status**
SI_DOC_001	An SI with the `action` attribute set to "`delete`" MUST have an explicitly assigned value for `si-id`.	7.2	SI_VAL_003:M