SMART MESSAGING

SPECIFICATION

Revision 3.0.0

2000-12-18

Subject to Change Without Notice

Disclaimer

The information in this document is provided ‘as is’, with no warranties whatsoever, including any warranty of merchantability, fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification, or sample. Furthermore, information provided in this document is preliminary, and may be changed substantially prior to final release. This document is provided for informational purposes only.

Nokia Mobile Phones Ltd. disclaims all liability, including liability for infringement of any proprietary rights, relating to implementation of information presented in this document. Nokia Mobile Phones Ltd. does not warrant or represent that such use will not infringe such rights.

Nokia Mobile Phones Ltd. retains the right to make changes to this specification at any time, without notice.

Third-party brands and names are the property of their respective owners.

License

A license is hereby granted to download and print a copy of this specification for personal use only. No other license, express or implied, by estoppel or otherwise, to any other intellectual property rights is granted herein.

Use of the Smart Messaging Specification shall be governed by the license terms set forth by Nokia in Smart Messaging Specification License Agreement and shall be subject to that you and Nokia execute Smart Messaging Specification License Agreement.

Third-Party Developer Support

Additional information on the Smart Messaging concept can be obtained by contacting Nokia Mobile Phones Ltd. at the following address:

Nokia Mobile Phones Ltd.
ATTN: Forum Nokia
Smart Messaging

P.O. Box 68
FIN-33721 Tampere
Finland

Internet: http://www.forum.nokia.com

< This page intentionally left blank >

Contents

References

GSM_03.38 Digital cellular telecommunications system (Phase 2+), Alphabets and language-specific information, GSM 03.38 version 5.4.0, ETSI, November, 1996. (http://www.etsi.org/)

GSM_03.40 Digital cellular telecommunications system (Phase 2+), Technical realisation of the Short Message Service (SMS), Point-to-Point (PP), GSM 03.40 version 5.4.0, ETSI, November, 1996. (http://www.etsi.org/)

ISO_8859 ISO 8859-1, Information processing — 8-bit single-byte coded graphic character sets — part 1: Latin Alphabet No. 1, International Organization for Standardization, February, 1987. (http://www.iso.ch/)

ISO/IEC_10646 ISO/IEC 10646-1, Information technology — Universal Multiple-Octet Coded Character Set (UCS) — Part 1: Architecture and Basic multilingual plane, May, 1993. (http://www.iso.ch/)

Unicode The Unicode Standard, Version 2.0. Unicode Consortium, 1993.

With the following addition: Unicode Technical Report #8: The Unicode Standard, Version 2.1. Unicode Consortium, 1998. (http://www.unicode.org/)

RFC_1766 Network Working Group, Request for Comments 1766: Tags for the Identification of Languages, UNINETT, H. Alvestrand, March, 1995. (http://www.rfc-editor.org/)

RFC_2425 Network Working Group, Request for Comments 2425: A MIME Content-Type for Directory Information. T. Howes, M. Smith, F. Dawson. September 1998. (http://www.rfc-editor.org/)

RFC_2426 Network Working Group, Request for Comments 2426: vCard MIME Directory Profile. F. Dawson, T. Howes. September 1998. (http://www.rfc-editor.org/)

WAP_WDP Wireless Application Protocol: Wireless Datagram Protocol. Version 1.2.1. WAP Forum, 2000. (http://www.wapforum.org/)

WAP_WTLS Wireless Application Protocol: Wireless Transport Layer Security. Version 1.1. WAP Forum, 1999. (http://www.wapforum.org/)

VCalendar vCalendar, The Electronic Calendaring and Scheduling Exchange Format, Version 1.0, a Versit Consortium Specification, September 18, 1996. (http://www.imc.org/)

VCard vCard, The Electronic Business Card, Version 2.1, a Versit Consortium Specification. (http://www.imc.org/)

Acronyms

ASCII	American Standard Code for Information Interchange
BCD	Binary Coded Decimal
BNF	Backus-Naur Form
CLI	Calling Line Identification
FTP	File Transfer Protocol
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
HTTP	HyperText Transfer Protocol
IANA	Internet Assigned Numbers Authority
IAP	Internet Access Point
IMAP	Internet Message Access Protocol
IMC	Internet Mail Consortium
ISDN	Integrated Services Digital Network
ISO	International Standards Organization
MIDI	Musical Instrument Digital Interface
MIME	Multipurpose Internet Mail Extensions
NBS	Narrow Band Socket
NC	Network Code
OTA	Over The Air
PC	Personal Computer
PCS	Personal Communications Services
POP	Post Office Protocol
RFC	Request For Comments
SAR	Segmentation And Re-assembly
SMS	Short Message Service
SMSC	Short Message Service Centre
TCP	Transmission Control Protocol
UCS-2	Universal Multiple-Octet Coded Character Set, 16-bit representation
UDP	User Datagram Protocol
UID	Unique Identifier
URL	Uniform Resource Locators
USSD	Unstructured Supplemental Services Data
UTC	Universal Time Coordinated
WAP	Wireless Application Protocol
WDP	Wireless Datagram Protocol
WTLS	Wireless Transport Layer Security

1. Introduction

1.1 Background

The emphasis in cellular networks is changing from voice-only communication to a rich combination of voice, messaging and browsing. Where voice communication once was accompanied only by circuit-switched data services, now various kinds of packet-switched communication means exist.

As the emphasis is moving in the direction of messaging, it is clear that close interaction is required between handset vendors, infrastructure vendors, cellular network operators and service providers. The quality of service the users receive depends crucially on how successful this interaction is.

The Smart Messaging specification defines the formats of the messages; this means that the messaging is bearer independent. In order to efficiently utilise the messaging capabilities of today’s and future networks, an open platform is required which enables support for today’s communication needs, as well as those that are still emerging. Today a set of tools and interfaces are needed to bring to the users the solutions they require, and tomorrow as the requirements evolve, the tools and interfaces must be able to grow as well. The tools and interfaces used are independent of this specification.

1.2 DOCUMENT OVERVIEW

This document is divided into three sections. First, the Introduction gives a short overview of the basic philosophy behind the document.

Second, the Smart Messaging architecture is described. In that section an overview is given of what is involved in building applications that interface with the architecture.

In the third section, the Message Formats are presented. That section describes the currently defined set of Smart Message Formats. This set of formats enables cellular network operators and service providers to provide services that work with a wide variety of handsets.

These sections are followed by appendixes A and B. Appendix A details the reserved port addresses. It presents information on the reserved socket port address space, and the port assignments for the currently defined "well-known" protocols. Appendix B includes detailed Smart Messaging examples.

This document does not discuss the extent of smart messaging implementation in any handset or smart phone. Each manufacturer provides separate information on the support in their products.

2. SMART MESSAGING

2.1 SMART MESSAGING ARCHITECTURE

2.1.1 Narrow-Band Sockets

Smart Messaging was first developed to use the Narrow-Band Socket protocol. Today it is most commonly used with a specific port number in the header of the Smart Message. Those port numbers are known as Narrow-Band Socket (NBS) port numbers. The NBS port numbers are included in Appendix A. The actual Narrow-Band Socket protocol is no longer used as originally designed. The WAP Forum's Wireless Datagram Protocol (WDP) is recommended to be used instead of the Narrow-Band Socket protocol when supported by the target device. [WAP_WDP].

Depending on the needs of service providers, they may choose from several platforms on which to provide their services. While smaller or specialised service providers might opt for a PC-based system, service providers focusing on large-scale service implementation can choose from the integrated systems available from network infrastructure vendors.

For operators the main concerns are how to integrate the new services with the existing infrastructure, and how to efficiently handle use of the air interface. For them, closely integrating new services with their existing service centres is a clean and effective solution.

2.1.2 Wireless Datagram Protocol

Since the publication of the earlier versions of this specification, the ideas presented by the Narrow-Band Socket protocol have been incorporated into the WAP Forum's Wireless Datagram Protocol (WDP). In practice, the WDP protocol does not deviate far from the Narrow-Band Socket protocol, and it is recommended to use WDP, if the recipient implements this protocol layer of the Wireless Application Protocol (WAP) stack and is capable of handling Smart Message payloads.

There are certain differences between NBS and WDP protocols when they are implemented over the GSM Short Message Service. In practice, WDP capable peers can receive NBS datagrams, but NBS only capable peers may not be able to receive WDP packets due to the differences in how Short Messages are used underneath these protocols. Peers that have a TCP/IP stack may use UDP to carry Smart Messages, just as they would use WDP. Therefore, WDP shares the TCP/UDP port number space. Therefore, some of the default port numbers mentioned in this specification differ between NBS and WDP implementations. The differences are listed with each message type.

Please note: All future implementations of the Smart Messaging protocol should use WDP and the WDP port numbers, if full interoperability with old NBS implementations is not critical.

Please note: Security services can be added to Smart Messaging by utilising the WTLS (Wireless Transport Layer Security) protocol, which offers security services for WDP. The use of WTLS [WAP_WTLS] is only possible when both communicating peers are using a WAP stack. WTLS cannot be used if either end is only capable of NBS.

Please note: NBS and WDP layers can be bypassed by using the so-called “keyword headers”. In this model, normal GSM Short Messages start with a certain keyword. The recipient detects this keyword and processes the rest of the Short Message as a Smart Message. All future Smart Messaging implementations should use keyword headers only if interoperability with an old implementation is critical.

2.1.3 Protocol Architecture

The WDP protocol is used to deliver Smart Messages; messages can also be delivered with specific port number in the header of the Smart Message. Those delivery methods are not bearer specific. If a peer supports a certain bearer, it does not imply that other bearers would be supported. A peer may support any combination of bearers. In addition to GSM’s Short Message Service, WDP can be adapted to work over, for example, USSD. As mentioned earlier, UDP (using a circuit-switched data call or GPRS) can also be used. It should be noted that Smart Messaging is not dependent on the network technology, and thus is not limited to GSM only.

Exchanging formats specified in the Message Formats section may be done by the applications at the port addresses specifically reserved for this purpose. The NBS port number space is controlled by Nokia. The TCP/UDP port number space (and subsequently, the WDP port number space) is controlled by IANA. For more information on the address space and reserved ports, please refer to Appendix A: Reserved Port Numbers.

2.2 MESSAGE FORMATS

The Smart Messaging architecture provides application developers with an extensible set of message formats. These formats enable any application to communicate with a wide variety of handsets. A list of relevant port numbers can be found in Appendix A: Reserved Port Numbers.

A list of message formats is presented in the Message Formats section. This basic set enables a rich set of applications to be built in the OTA environment. The set of specifications currently covers the following areas:

- Sending or receiving business cards.
- Sending or receiving Internet Access Configuration related information.
- Sending or receiving calendar items.
- Sending and receiving ringing tones and graphical information
- Sending and receiving multipart messages

< This page intentionally left blank >

3. MESSAGE FORMATS

3.1 NOTATION

The notation used in the Message Formats section uses the common elements presented here. The Backus-Naur Form (BNF) notation is used to describe the format of the messages. The BNF format and the extensions used are as follows:

- In BNF notation "::=" means "definition", where a non-terminal symbol is on the left side of the operator "::=", and the definition is on the right side.
- The symbol order in the BNF notation is the same as the syntax symbol order.
- Terminal symbols are enclosed between quotes (**"**), and the symbols are written in **bold**.
- Non-terminal symbols are enclosed between the < and > characters, and the symbols are written in *italics*.
- Textual definitions for non-terminal characters are enclosed between apostrophes (**'**).
- The operator "|" is used as a delimiter between multiple choices.
- Grouping of items is expressed by enclosing them in the meta symbols { and }.
- Optional parts are enclosed in meta symbols [and].
- Kleene's "+" operator is supported, i.e. <A>$^{+}$ denotes repetition of non-terminal <A> from **1** to infinity.
- Kleene's "*" operator is supported, i.e. <A>* denotes repetition of non-terminal <A> from **0** to infinity.
- The semicolon symbol ";" means that the rest of the line contains comments.

3.1.1 Common Text Elements

The used character literals, e.g. "5", represent the corresponding character values encoded according to the currently used character set. The currently used character set is either explicitly indicated as part of the definition of a Smart Message type, or is the default (“native”) character set for the bearer service used (for example GSM Short Message Service). Note that the bearer’s native character set may be the only character set that is usable on a particular bearer (for instance, because the bearer supports only 7-bit data). The native character set of the bearer should be used when there is no explicit information on the availability of other character sets.

In GSM the native character set is the character set defined in [GSM_3.38] and [GSM_3.40]. Existing implementations usually default to this character set unless another one has been explicitly specified.

Character Definitions

<*line-feed*> ::= ‘The line feed character of the currently used character set, corresponding to the character with ISO 8859-1 value 10 decimal.’

<*carriage-return*> ::= ‘The carriage return character of the currently used character set, corresponding to the character with ISO 8859-1 value 13 decimal.’

<*line-delimiter*> ::= <*line-feed*> | <*carriage-return*>

<*space*> ::= ‘The space character of the currently used character set, corresponding to the character with ISO 8859-1 value 32 decimal’

<*default-char*> ::= ‘Any character in the character set that is currently being used.’

<*default-char-not-lf*> ::= ‘Any character in the currently used character set except <*line-feed*>.’

<*default-char-not-ld*> ::= ‘Any character in the currently used character set except <*line-delimiter*>.’

<*default-char-not-space*> ::= ‘Any character in the currently used character set except <*space*>.’

<*unicode-char*> ::= ‘Any 16-bit Unicode (see [Unicode] and [ISO/IEC_10646]) character in UCS-2 encoding (each encoded character is represented in a 16-bit quantity). Implies that the message transport must be eight-bit-clean.’

<*ISO-8859-1-char*> ::= ‘Any 8-bit ISO 8859-1 (ISO Latin 1, see [ISO_8859]) character. Implies that the message transport must be eight-bit-clean.’

Dialling Definitions

<*common-digit*> ::= “**1**” | ”**2**” | ”**3**” | ”**4**” | ”**5**” | ”**6**” | ”**7**” | ”**8**” | ”**9**” | ”**0**”

<*common-tel-char*> ::= <*common-digit*> | “**-**“ | “**#**” | “*” | “**W**” | “**w**” | “**P**” | “**p**” | <*space*>

‘The character “**W**” / “**w**” commands the handset to wait for a dial-tone before dialling the rest of the sequence. The character “**P**” / “**p**” commands the handset to pause for a predefined interval prior to dialling the rest of the string.’

<*common-phone-number*> ::= [“**+**”] <*common-tel-char*>*

Information Encapsulation Definitions

<*common-information-content*> ::= <*information-content-length*> “**:**” <*information-content-char*>*

<*information-content-length*> ::= ‘decimal value encoded in a character string in the currently used character set, i.e. “**42**” for a message length of 42 when using ISO 8859-1. This value counts the number of 7/8-bit bytes, depending on the transmission channel setup. In the case of Unicode characters, this value is twice the number of 16-bit Unicode characters (i.e. the number of octets). This strictly counts the number of <*information-content-char*>:s; this means that <*information-content-length*> and terminal character “**:**” are excluded from the length.’

<*information-content-char*> ::= <*default-char*>

Miscellaneous Definitions

<*common-language*> ::=

<*default-char-not-lf*>$^+$ ["**-**" <*default-char-not-lf*>$^+$]

```
; ‘The language is as defined in RFC1766 [RFC_1766], with a language code followed by an optional country
code. For example, US English is presented as “en-US”, whereas generic English is presented as “en”. The
language codes and the country codes are case-sensitive.’

<common-boolean> ::=      { “t” | “T” } <default-char-not-lf>* |    ; ‘as in “True”’
                          { “f” | “F” } <default-char-not-lf>*      ; ‘as in “False”’

<common-flip-option> ::=
        <common-flip-option-yes> | <common-flip-option-no> |
        <common-flip-option-on> | <common-flip-option-off>

<common-flip-option-yes> ::= “Y”     ; ‘case-insensitive’
<common-flip-option-no> ::= “N”      ; ‘case-insensitive’
<common-flip-option-on> ::= “On”     ; ‘case-insensitive’
<common-flip-option-off> ::= “Off”   ; ‘case-insensitive’


<common-hex-digit> ::= <common-digit> | “A” | “B” | “C” | “D” | “E” | “F”

<common-charset-field> ::=
        <common-charset-gsm-default> |
        <common-charset-ascii> |
        <common-charset-iso8859> |
        <common-charset-ucs2>

<common-charset-gsm-default>   ::= “GSM”          ; ‘Refer to [GSM_03.38], [GSM_03.40]
<common-charset-ascii>         ::= “ASCII”        ; ‘US-ASCII: Deprecated’
<common-charset-iso8859>       ::= “ISO-8859-1”   ; ‘Refer to [ISO_8859]’
<common-charset-ucs2>          ::= “UCS2”         ; ‘Refer to [ISO/IEC_10646] and [Unicode]’

<common-version-field> ::= <common-digit>+ [ “.” <common-digit>+ ]
; ‘Version number. Format: major.minor version, i.e. “1.2“’
```

Date/Time Definitions

```
<common-date> ::= <year><month><day> [ <time> [ <type-designator> ] ]
;’The basic format of ISO 8601 for date and time. for example, 19971031T231210’.

<year> ::= <common-digit> <common-digit> <common-digit> <common-digit>    ; ‘i.e. “1997”’
<month> ::= <common-digit> <common-digit>                                ; ‘i.e. “03”’ for March’
<day> ::= <common-digit> <common-digit>                                  ; ‘i.e. “08”
<time> ::= “T” <hours> <minutes> <seconds>
<hours> ::= <common-digit> <common-digit>                                ; ’24-hour format’
<minutes> ::= <common-digit> <common-digit>
<seconds> ::= <common-digit> <common-digit>
```

<*type-designator*> ::= “**Z**” ; ’This means the time is Universal Time Coordinated (UTC)’

3.2 INTERNET ACCESS CONFIGURATION

Internet access point configuration information can be transmitted to handsets to enable automatic access point configuration for the Internet email and Internet access used. The configuration support can be extended to also cover bookmarks, scripts, FTP, Telnet, terminal and WWW settings. Also, GSM Short Message Centre numbers and Voice Mail numbers can be configured in a similar way.

There are two flavours of Internet access point configuration messages. The simpler type is described in the Basic Configuration Syntax section, whereas the more complicated syntax is presented in the Extended Configuration Syntax section.

If the message is not transferred with the NBS port number in the message header or over WDP, the <*iap-compatibility-header*> should be used, and vice versa. When compatibility with older implementations is critical, the message should not be sent with the NBS port number in the message header or over WDP. Recipients should always be prepared to receive either variant of this message.

The Internet access point configuration data reader is listening on NBS port 5503 decimal (157F hexadecimal).

The default usage of this format is on top of a 7-bit transmission channel. The format can also be transmitted over wider than 7-bit transmission channels. In such cases the highest bits in the representation are set to zero if there is a possibility for ambiguity.

Please note: When configuring Internet access, the configuration messages should *only* configure the set of parameters that are crucial for the correct operation of the Internet access point. Some configurable features are user settings by nature and do not affect the operation of the Internet access point. Such settings should generally not be configured remotely.

Please note: Only use the smallest number of settings that you can. If a setting is not required, leave it out for maximum compatibility between different implementations, unless the product-specific documentation mandates its use.

Please note: Capabilities differ between different products. You should consult the appropriate developer documentation for the product you are planning to configure by using these messages. For information on the capabilities of a specific product, please contact the product vendor.

Please note: Service providers are recommended to take field length limits into consideration when designing the configuration and setup of their devices. If a field identifier is present, but the value of that field is left empty, then the empty value will be updated in the device; check for empty field values.

Please note: For security reasons, passwords should not be sent in the configuration messages.

Several informative examples of the Internet Access Configuration messages can be found in Appendix B.

3.2.1 Basic Configuration Syntax

<*iap-message*> ::= [<*iap-compatibility-header*>] <*notify-text*> <*info-body*>

<*iap-compatibility-header*> ::= "**//SIAP11**" <*line-feed*> ; 'optional compatibility header'
; 'used without NBS headers'

<*notify-text*> ::= <*default-char-not-lf*>* <*line-feed*>

<*info-body*> ::= <*basic-configuration-info*>$^+$ | <*extended-configuration-info*>

<*basic-configuration-info*> ::= <*iap-info*> | <*mail-info*>

3.2.1.1 Basic IAP

```
<iap-info> ::= <iap-info-name> { <iap-parameter> <line-feed> }*
<iap-info-name> ::=  “Iname:” <default-char-not-lf>*  <line-feed> ; ‘maximum length 50 chars’

<iap-parameter> ::=
            <iap-parameter-phone-number> |
            <iap-parameter-user-name> |
            <iap-parameter-prompt-for-password> |
            <iap-parameter-password> |
            <iap-parameter-modem-initialization> |
            <iap-parameter-ip-address> |
            <iap-parameter-primary-nameserver> |
            <iap-parameter-secondary-nameserver> |
            <iap-parameter-network-mask> |
            <iap-parameter-default-gateway>

<iap-parameter-phone-number> ::= “Itel:” [ <common-phone-number> ]   ; ‘maximum length 16 chars’
<iap-parameter-user-name> ::= "Iuid:" <default-char-not-lf>*         ; ‘maximum length 50 chars’
<iap-parameter-prompt-for-password> ::= “Ipwp:” [ “Y” | “N” ]        ; ‘whether to ask for a password from
the’
                                                                     ; ‘user upon connecting’
<iap-parameter-password> ::= "Ipwd:" <default-char-not-lf>*          ; ‘password’
                                                                     ; ‘password should not be included for
                                                                     security reasons’
<iap-parameter-modem-initialization> ::= "Iini:" <default-char-not-lf>*  ; ‘maximum length 50 chars’
<iap-parameter-ip-address> ::= "Iip:" [ <ip-string> ]                ; ‘static IP address, if it exists’
<iap-parameter-primary-nameserver> ::= "Idns1:" [ <ip-string> ]      ; ‘primary name server (DNS) address’
<iap-parameter-secondary-nameserver> ::= "Idns2:" [ <ip-string> ]    ; ‘secondary name server address’
<iap-parameter-network-mask> ::= "Imsk:" [ <ip-string> ]             ; ‘network mask’
<iap-parameter-default-gateway> ::= "Idgw:" [ <ip-string> ]          ; ‘default gateway’

<ip-string> ::= <byte-dec> "." <byte-dec> "." <byte-dec> "." <byte-dec>
<byte-dec> ::= <common-digit> [ [ <common-digit> ] <common-digit> ]  ;‘value in range [0..255] decimal’
```

3.2.1.2 Basic Mail

```
<mail-info> ::= <mail-iap-name> { <mail-parameter> <line-feed> }*

<mail-parameter> ::=
            <mail-parameter-remote-mailbox-username> |
            <mail-parameter-remote-mailbox-password> |
            <mail-parameter-own-email-address> |
            <mail-parameter-receiving-host> |
            <mail-parameter-sending-host> |
            <mail-parameter-remote-mailbox-protocol>
```

```
<mail-iap-name> ::= "Mname:" <default-char-not-lf>* <line-feed>               ;'used IAP name, max. 50 chars'
<mail-parameter-remote-mailbox-username> ::= "Muid:" <default-char-not-lf>*        ; 'remote mailbox user
                                                                                     name, maximum length
                                                                                     50 chars'
<mail-parameter-remote-mailbox-password> ::= "Mpwd:" <default-char-not-lf>*        ; 'remote mailbox password,
                                                                                     maximum length 50 chars'
<mail-parameter-own-email-address> ::= "Madr:" [ <email-address> ]       ; 'the user's email address,
                                                                           maximum length 100 chars'
<mail-parameter-receiving-host> ::= "Mrcv:" [ <iaddress> ]               ; 'host name of the IMAP/POP server'
<mail-parameter-sending-host> ::= "Msnd:" [ <iaddress> ]                 ; 'host name of the SMTP server'
<mail-parameter-remote-mailbox-protocol> ::= "Mpro:" <mail-protocol>     ; 'whether to use IMAP or POP'

<email-address> ::= <default-char-not-lf>*                   ; 'RFC 822 based name@host.domain format '
<iaddress> ::= <ip-string> | <hostname>
<hostname> ::= <default-char-not-lf>*                        ; 'a host name in DNS, maximum length 50 chars'

<mail-protocol> ::=  "IM" |
                     "PO"                                    ; 'IM = IMAP4, PO = POP3'
```

3.2.2 Extended Configuration Syntax

Extended configuration syntax enables more configuration message types.

```
<extended-configuration-info> ::=
        <extended-iap-info> |
        <extended-mail-info> |
        <hotlist-item-info> |
        <script-info> |
        <sms-info> |
        <telnet-info> |
        <terminal-info> |
        <www-info> |
        <ftp-info> |
        <internet-info> |
        <telephone-info>  |
        <www-autofetch-info>
```

3.2.2.1 Extended IAP

```
<extended-iap-info> ::=
        <iap-info-name> { <iap-parameter> <line-feed> | <extended-iap-parameter> <line-feed> }*

<extended-iap-parameter> :: =
        <iap-parameter-extended-phone-number> |
        <iap-parameter-http-no-proxy-for> |
```

<*iap-parameter-http-port*> |
<*iap-parameter-http-proxy*> |
<*iap-parameter-login-customisation*> |
<*iap-parameter-compression*> |
<*iap-parameter-secure-proxy*> |
<*iap-parameter-secure-port*>

<*iap-parameter-extended-phone-number*> ::=
“**Itel:**” [<*common-phone-number*>] ; ‘phone number of the dial-up pool, maximum length 30 chars’

<*iap-parameter-http-no-proxy-for*> ::= “**Inop:**” <*default-char-not-lf*>* ; ‘domains for which no HTTP proxy will be used’
<*iap-parameter-http-port*> ::= “**Ippo:**” <*common-digit*>* ; ‘HTTP proxy port, maximum length 5 digits’
<*iap-parameter-http-proxy*> ::= “**Iprx:**” <*ip-string*> ; ‘HTTP proxy address’
<*iap-parameter-login-customisation*> ::= “**Ilgn:**” <*login-type*> ; ‘whether to use login customisation’
<*iap-parameter-compression*> *::= "* **Icmp:**" [“**Y**” | “**N**”] ; ' whether to use PPP compression'
<*iap-parameter-secure-proxy*> ::= "**Isrx:**" <*ip-string*> ; 'SSL tunnelling proxy IP address'
<*iap-parameter-secure-port*> ::= "**Ispo:**" <*common-digit*> ; 'SSL tunnelling proxy port, maximum length 5 digits’
<*login-type*> ::= “**0**” | “**1**” | <*default-char-not-lf*>* ; ‘ “**0**” for none, “**1**” for manual,
; or name of the script.’

3.2.2.2 Extended Mail

Extended mail settings control the features of the device’s email client.

<*extended-mail-info*> ::=
[<*mail-info-name*>] { <*mail-parameter*> <*line-feed*> | <*extended-mail-parameter*> <*line-feed*> }*

<*extended-mail-parameter*> :: =
<*mail-parameter-copy-to-own-email-address*> |
<*mail-parameter-delete-fetched*> |
<*mail-parameter-fetch-attachments*> |
<*mail-parameter-fetch-headers*> |
<*mail-parameter-mime-encoding*> |
<*mail-parameter-remote-mailbox-folder*> |
<*mail-parameter-selected-font*> |
<*mail-parameter-send-mail*> |
<*mail-parameter-show-header-fields*> |
<*mail-parameter-define-remote-mailbox*>

<*mail-parameter-copy-to-own-email-address*> ::= "**Mcpy:**" <*common-flip-option*> ; ‘whether to send a copy to own address’
<*mail-parameter-delete-fetched*> ::= "**Mdel:**" <*common-flip-option*> ; ’whether to delete fetched mail from remote server’

```
<mail-parameter-fetch-attachments> ::= "Matt:" <common-flip-option>          ; 'whether to retrieve
                                                                              attachments'
<mail-parameter-fetch-headers> ::= "Mhdr:" <fetch-option>                     ; 'which headers to retrieve'
<mail-parameter-mime-encoding> ::= "Mime:" <common-flip-option>               ; 'whether to use MIME
                                                                               encoding in outgoing mail'
<mail-parameter-remote-mailbox-folder> ::= "Mbox:" <default-char-not-lf>*     ; 'remote mailbox folder
                                                                              name, maximum length 30
                                                                              chars'
<mail-parameter-selected-font> ::= "Mfnt:" <font-selection>                   ; 'mail editor typeface
                                                                              ; (use not recommended)'
<mail-parameter-send-mail> ::= "Msen:" <mail-send-option>                     ; 'when to send mail'
<mail-parameter-show-header-fields> ::= "Mshf:" <mail-show-headers-option>    ; 'which header fields to
                                                                              show'
<mail-parameter-define-remote-mailbox> ::= "Mid:" <default-char-not-lf>*      ; 'remote mailbox name'

<fetch-option> ::= <fetch-option-all> | <fetch-option-recent>
<fetch-option-all> ::= { "a" | "A" } <default-char-not-lf>*          ; 'as in "All"'
<fetch-option-recent> ::= { "r" | "R" } <default-char-not-lf>*       ; 'as in "Recent"'

<font-selection> ::= <font-urw-mono> | <font-urw-roman> | <font-urw-sans>
<font-urw-mono> ::= { "m" | "M" } <default-char-not-lf>*             ; 'as in "Mono"'
<font-urw-roman> ::= { "r" | "R" } <default-char-not-lf>*            ; 'as in "Roman"'
<font-urw-sans> ::= { "s" | "S" } <default-char-not-lf>*             ; 'as in "Sans"'

<mail-send-option> ::=
          <mail-send-option-later> |
          <mail-send-option-next> |
          <mail-send-option-now>

<mail-send-option-later> ::= { "u" | "U" } <default-char-not-lf>*    ; 'as in "Upon request".'
<mail-send-option-next> ::= { "d" | "D" } <default-char-not-lf>*     ; 'as in "During next connection".'
<mail-send-option-now> ::= { "i" | "I" } <default-char-not-lf>*      ; 'as in "Immediately".'

<mail-show-headers-option> ::=
          <mail-show-headers-options-none> |
          <mail-show-headers-options-basic> |
          <mail-show-headers-options-all>

<mail-show-headers-options-none> ::= { "n" | "N" } <default-char-not-lf>*   ;'as in "None".'
<mail-show-headers-options-basic> ::= { "b" | "B" } <default-char-not-lf>*  ;'as in "Basic".'
<mail-show-headers-options-all> ::= { "a" | "A" } <default-char-not-lf>*    ;'as in "All".'
```

3.2.2.3 WWW Hotlist Item Settings

WWW hotlist item settings control the addition of new items to the bookmark (hotlist) of the device's World Wide Web browser.

<hotlist-item-info> ::= <hotlist-item-group>$^{+}$

<hotlist-item-group> ::=

    <hotlist-item-name> <line-feed>

    <hotlist-item-url> <line-feed>

    [<hotlist-autoselect-iap> <line-feed>]

    [<hotlist-access-point> <line-feed>]

    [<hotlist-folder> <line-feed>]

<hotlist-item-name> ::= “**Hname:**” <default-char-not-lf>* ; ‘the name of the hotlist item’

<hotlist-item-url> ::= “**Hurl:**” <default-char-not-lf>* ; ‘location (URL) of the document’

<hotlist-autoselect-iap> ::= “**Haap:**” <common-flip-option> ; ‘whether to automatically select IAP
; when fetching’

<hotlist-access-point> ::= “**Hiap:**” <default-char-not-lf>* ; ‘which IAP to use’

<hotlist-folder> ::= "**Hdir:**" <default-char-not-lf>* ; ‘in which folder to place the hotlist item’

3.2.2.4 Script Settings

Script settings control the addition of new login scripts to log into an IAP.

<script-info> ::=

    <script-name> <line-feed>

    <script-type> [<space> <script-version>]<line-feed>

    <script-data>

<script-name> ::= “**Pname:**” <default-char-not-lf>* ; ’filename, maximum length 32‘

<script-type> ::= “**Ptype:**” <script-type-identifier> ; ‘for what the script is used for’

<script-type-identifier> ::= **“PPPS”** | **"PPP" |** <default-char-not-lf-or-space>* ; ‘see product-specific documentation’

<script-version> ::= <common-version-field> ; ‘script version number’

<script-data> ::= “**Pdata:**” <common-information-content> ; ‘the script itself’

<script-addition> ::= “**Padd:**” <common-information-content> ; ‘additional data for an existing script’

3.2.2.5 SMS Settings

SMS settings control the features of the device’s Short Messaging application.

<sms-info> ::= {<sms-item> <line-feed> }$^{+}$

<sms-item> ::= [<sms-service-center-name>] <sms-service-center-number>

<sms-service-center-name> ::= "**Sname:**" <default-char-not-lf>* ; ‘GSM SMSC name’

<sms-service-center-number> ::= “**Stel:**” [<common-phone-number>] ; ‘GSM SMSC telephone number’

3.2.2.6 Telnet Settings

Telnet settings control the features of the device’s Telnet application.

<telnet-info> ::=

    { <telnet-connection-name> <line-feed> { <telnet-item> <line-feed> }* }$^{+}$

<*telnet-connection-name*> ::= “**Tname:**” <*default-char-not-lf*>*   ; ‘Telnet connection name’

<*telnet-item*> ::=

  <*telnet-backspace-key*> |

  <*telnet-destination-host*> |

  <*telnet-internet-access*>

<*telnet-backspace-key*> ::= “**Tdel:**” <*backspace-key*>   ; ‘which backspace code to send’

<*telnet-destination-host*> ::= “**Thst:**” <*default-char-not-lf*>*   ; ‘destination host name’

<*telnet-internet-access*> ::= “**Tiap:**” <*default-char-not-lf*>*   ; ‘which IAP to use’

<*backspace-key*> ::= <*key-backspace*> | <*key-del*>

<*key-backspace*> = “**BS**”   ; ‘send backspace (ISO 8859-1 8_{10})’

<*key-del*> ::= “**DEL**”   ; ‘send delete (ISO 8859-1 127_{10})’

3.2.2.7 Terminal Settings

Terminal settings control the features of the device’s terminal application.

<*terminal-info*> ::=

  { <*terminal-connection-name*> <*line-feed*> { <*terminal-item*> <*line-feed*> }* }$^{+}$

<*terminal-connection-name*> ::= “**Rname:**” <*default-char-not-lf*>*   ; ‘terminal connection name’

<*terminal-item*> ::=

  <*terminal-backspace-key*> |

  <*terminal-data-bits*> |

  <*terminal-line-end-convention*> |

  <*terminal-local-echo*> |

  <*terminal-modem-init*> |

  <*terminal-parity*> |

  <*terminal-phone-number*> |

  <*terminal-stop-bits*> |

  <*terminal-incoming-echo*> |

  <*terminal-incoming-line-end*>

<*terminal-backspace-key*> ::= “**Rdel:**” <*backspace-key*>   ; ‘which backspace code to send’

<*terminal-data-bits*> ::= “**Rdat:**” <*terminal-databits*>   ; ‘number of data bits’

<*terminal-line-end-convention*> ::= “**Rend:**” <*terminal-line-end*>   ; ‘which line-end convention to use’

<*terminal-local-echo*> ::= “**Rech:**” <*common-flip-option*>   ; ‘whether to locally echo characters’

<*terminal-modem-init*> ::= “**Rini:**” <*default-char-not-lf*>*   ; ‘modem initialisation string to use’

<*terminal-parity*> ::= “**Rpar:**” <*terminal-parity*>   ; ‘parity to be used’

<*terminal-phone-number*> ::= “**Rtel:**” <*common-phone-number*>   ; ‘phone number to call’

<*terminal-stop-bits*> ::= “**Rstp:**” <*terminal-stopbits*>   ; ‘number of stop bits’

<*terminal-incoming-echo*> *::= "* **Reci:**" <*common-flip-option*>   ; ‘whether to locally echo characters during incoming data calls’

<*terminal-incoming-line-end*> *::="* **Reni**:" <*terminal-line-end*>   ; ‘which line-end convention to use during incoming data calls’

<*terminal-databits*> ::= "**7**" | "**8**" ; '7 or 8 data bits'

<*terminal-line-end*> ::= "**CR**" | "**LF**" | "**CRLF**" ; 'line end: CR (ISO 8859-1 13_{10}), LF (ISO 8859-1 10_{10}) or CR+LF pair'

<*terminal-parity*> ::= <*terminal-parity-none*> | <*terminal-parity-odd*> | <*terminal-parity-even*>

<*terminal-parity-none*> ::= { "**n**" | "**N**" } <*default-char-not-lf*>* ; 'as in "**None**"'

<*terminal-parity-odd*> ::= { "**o**" | "**O**" } <*default-char-not-lf*>* ; 'as in "**Odd**"'

<*terminal-parity-even*> ::= { "**e**" | "**E**" } <*default-char-not-lf*>* ; 'as in "**Even**"'

<*terminal-stopbits*> ::= "**1**" | "**2**" ; '1 or 2 stop bits'

3.2.2.8 WWW Settings

WWW settings control the features of the device's World Wide Web browser.

<*www-info*> ::= {<*www-item*> <*line-feed*> }$^{+}$

<*www-item*> ::= <*www-access-point*>

<*www-access-point*> ::= "**Wiap:**" <*default-char-not-lf*>* ; 'default IAP to use'

3.2.2.9 FTP Settings

FTP settings control the features of the device's File Transfer Protocol application.

<*ftp-info*> ::=

{ <*ftp-connection-name*> <*line-feed*> { <*ftp-item*> <*line-feed*> }* }$^{+}$

<*ftp-connection-name*> ::= <*ftp-parameter-name*> ::= **"Fname:"** <*default-char-not-lf*>*

; 'FTP connection name'

<*ftp-item*> ::=

<*ftp-parameter-access-point*> ::= **"Fiap:"** <*access-point*> | ; 'which IAP to use'

<*ftp-parameter-ip-address*> ::= **"Fip:"** <*ip-string*> | ; 'IP address of the FTP server'

<*ftp-parameter-password*> ::= **"Fpwd:"** <*default-char-not-lf*>* | ; 'FTP server password'

<*ftp-parameter-port*> ::= **"Fprt:"** <*common-digit*> | ; 'FTP server port'

<*ftp-parameter-username*> ::= **"Fuid:"** <*default-char-not-lf*>* ; 'FTP server user name'

3.2.2.10 Internet Settings

Internet settings describe the settings for the data connection.

<*internet-info*> ::= {<*internet-item*> <*line-feed*>}$^{+}$

<*internet-item*> ::= <*internet-parameter-compression*> | <*internet-parameter-modem-initialization*>

<*internet-parameter-compression*> ::= "**Iv42:**" <*common-flip-option*> ; 'whether to use V.42bis'

<*internet-parameter-modem-initialization*> ::= "**Imdm:**" <*internet-modem*> ; 'modem type to use'

<*internet-modem*> ::=

*"**0**"* | ; ' for "Autobauding"

*"***2***"* | ; ' for "Fixed 9600 b/s"

```
                "4"  |                                                  ; ' for "Fixed 14400 b/s"
                <default-char-not-lf>                                   ; ' for "Custom"
```

3.2.2.11 Telephone Settings

Telephone settings configure the voice mailbox number.

```
<telephone-info> ::= <telephone -item>

<telephone-item> ::= <telephone–voice-mailbox-number>

<telephone–voice-mailbox-number> ::= "Tbox:" <common-phone-number>          ; ‘phone number to call’
```

3.2.2.12 WWW Autofetch Settings

WWW Autofetch settings control how the terminal downloads material from the World Wide Web automatically, without user intervention.

```
<www-autofetch-info> ::=
                { <www-autofetch-url> <line-feed> { <www-autofetch-item> <line-feed> }* }+
<www-autofetch-item> ::=
                <www-autofetch-body> |
                <www-autofetch-headers> |
                <www-autofetch-iap> |
                <www-autofetch-method> |
                <www-autofetch-name>

<www-autofetch-url> ::= "Aurl:" <default-char-not-lf>*                  ; ‘location (URL) to fetch’
<www-autofetch-body> ::= "Abdy:" <default-char-not-lf>*                 ; ‘send this data as HTTP POST
                                                                        body part’
<www-autofetch-headers> ::= "Ahdr:" <default-char-not-lf>*              ; ‘additional HTTP headers’
<www-autofetch-iap> ::= "Aiap:" <default-char-not-lf>*                  ; ‘IAP to use’
<www-autofetch-method> ::= "Amth:"  <www-autofetch-method-parametres>       ; ‘which HTTP method to use’
<www-autofetch-name> ::= "Aname:" <default-char-not-lf>*                ; ‘bookmark name’
<www-autofetch-method-parametres> ::= <method-get> | <method_post> | <method_put>
<method-get> ::= "1"                                                    ; ' default method (HTTP GET)'
<method-post> ::= "2"                                                   ; ‘HTTP POST (form submission)’
<method-put> ::= "3"                                                    ; ‘HTTP PUT’
```

3.3 EMAIL NOTIFICATION

Email servers can use this message to notify cellular terminals of the existence of new messages in the mail server. This message format is a legacy format.

When not transferring this message with the NBS port number in the message header or over WDP, the *<email-compatibility-header>* must be used, and vice versa. If compatibility with existing products is critical, the message should not be sent with the NBS port number in the message header or over WDP. However, the recipients should be prepared to receive both variants of the message.

The email notification reader is listening on NBS port 5512 decimal (1588 hexadecimal).

The default usage of this format is on top of a 7-bit transmission channel. The format can also be transmitted over wider channels. In such cases the highest bits in the representation are set to zero if there is a possibility for ambiguity.

The Extended Email notification chapter specifies the format of smart message that can be used to build the mail headline view without establishing a connection to the remote mailbox or just to notify the user of new mail, if a legacy phone is used. Two different modes are described: restricted mode and non-restricted mode. In the restricted mode, the total length of the smart message MUST NOT be more than 160 characters (or 140 characters, if the smart message contains 8-bit characters). This is achieved by cutting the subject field so that the size limit is not exceeded and/or by leaving unimportant fields out of the mail notification. In the non-restricted mode, no size limit is defined. There are no obligatory fields in either mode.

3.3.1 Simple Email Notification

3.3.1.1 Syntax

<*simple-notify*> ::= <*mail-header*> <*total-count-of-new-messages*>

<*mail-header*> ::= <*email-compatibility-header*> | <*nbs-header*>

<*email-compatibility-header*> ::= "**//MLAP11**" <*line-feed*>

<*nbs-header*> ::= "//**SCKL1588**" <*line-feed*>

<*total-count-of-new-messages> ::= <common-digit>*$^+$; 'Total count of the messages in the server.'

3.3.2 Extended Email notification

3.3.2.1 Syntax

<*extended-notify*> ::= <*header*> <*new-amount*> [<*extended-mail-notify-item*>$^+$]

<*header*> ::= (<*mail-header*> | <*nbs-header*>) <*line-feed*>

<*new_amount*> ::= <*nz-digit*> <*common-digit*> * [<*comment*>] <*line-feed*>

<*extended-mail-notify-item* >::=

<*from*> |

<*subject*> |

<*size*> |

<uid> |
<server-id> |
<attachments> |
_<to>_ |
_<cc>_ |
<date> |
<folder> |
<sender> |
<reply-to> |
<uid-validity> |
<extension-field>

<from> ::= “**from:**” _<ws-char>_* _<rfc822-address>_ _<line-feed>_

<subject> ::= “**subject:**” _<ws-char>_* _<char>_* _<line-feed>_

<size> ::= “**size:**” _<ws-char>_* _<nz-digit>_ _<common-digit>_ * _<ws-char>_* (“**lines**” | “**kB**”) _<line-feed>_

<uid> ::= _<imap-uid>_ | _<pop-uid>_

<imap-uid> ::= “**iuid:**” _<ws-char>_* _<nz-digit>_ _<common-digit>_ * _<line-feed>_ ; max 10 digits

<pop-uid> ::= “**puid:**” _<ws-char>_* _<7-bit-char>_$^+$ _<line-feed>_ ; max 70 chars

<server-id> ::= “**sid:**” _<ws-char>_* _<common-digit><common-digit><common-digit><line-feed>_

<attachments> ::= “**att:**” _<ws-char>_* _<common-digit>_ $^+$ _<line-feed>_

_<to>_ ::= “**to:**” _<ws-char>_* _<rfc822-address>_ (“**,**” _<rfc822-address>_)* [_<truncate-comment>_] _<line-feed>_

_<cc>_ ::= “**cc:**” _<ws-char>_* _<rfc822-address>_ (“**,**” _<rfc822-address>_)* [_<truncate-comment>_] _<line-feed>_

<date> ::= ”**date:**” _<ws-char>_* _<rfc822-date>_ _<line-feed>_

<folder> ::= “**fldr:**” _<ws-char>_* _<7-bit-char>_$^+$ _<line-feed>_ ; ' default INBOX '

<sender> ::= “**sender:**” _<ws-char>_* _<rfc822-address>_ (“**,**” _<rfc822-address>_)* _<line-feed>_

<reply-to> ::= “**reply-to:**” _<ws-char>_* _<rfc822-address>_ (“**,**” _<rfc822-address>_)* _<line-feed>_

<uid-validity> ::= “**uidv:**” _<ws-char>_* _<nz-digit>_ _<common-digit>_ * _<line-feed>_

<extension-field> ::= _<extension-name>_ _<colon>_ _<extension-value>_ _<line-feed>_

<extension-name> ::= _<field-name-char>_$^+$

<field-name-char> ::= ‘any char except _<colon>_ and _<line-feed>_’

<extension-value> ::= _<char>_*

<comment> ::= _<ws-char>_$^+$ _<char>_*

<username> ::= _<7-bit-char>_*

<port-number> ::= _<nz-digit>_ _<common-digit>_ *

<mail-header> ::= “//**MLAP11**”

<nbs-header> ::= “//**SCKL1588**” | “//**SCKL15881588**” _<common-hex-digit>_ _<common-hex-digit>_ _<common-hex-digit>_ _<common-hex-digit>_ _<common-hex-digit>_ _<common-hex-digit>_

<colon> ::= “**:**”

<crlf> ::= _<carriage-return>_ | _<line-feed>_ | _<carriage-return>_ _<line-feed>_

<char> ::= _<default-char-not-lf>_*

<7-bit-char> ::= ‘any 7-bit char except _<line-feed>_’

<non-digit> ::= ‘any char except _<common-digit>_ and _<line-feed>_’

<nz-digit> ::= “**1**” | “**2**” | “**3**” | “**4**” | “**5**” | “**6**” | “**7**” | “**8**” | “**9**”

<rfc822-address> ::= _<mailbox>_ | _<group>_ ; see RFC822

<*rfc822-date*> ::= [<*day*> “**,**”]<*common-digit*> [<*common-digit*>] <*space*> <*email-notify-month*> <*space*> <*email-notify-year*> <*space*> <*email-notify-time*>

<*email-notify-year*> ::= <*common-digit*> <*common-digit*> <common*digit*> <*common-digit*>

<*email-notify-month*> ::= “**Jan**” | “**Feb**” | “**Mar**” | “**Apr**” | “**May**” | “**Jun**” | “**Jul**” | “**Aug**” | “**Sep**” | “**Oct**” | “**Nov**” | “**Dec**”

<*email-notify-time*> ::= <*hour*> <*space*> <*zone*>   ; see RFC822

<*email-notify-day*> ::= “**Mon**” | “**Tue**” | “**Wed**” | “**Thu**” | “**Fri**” | “**Sat**” | “**Sun**”

<*truncate-comment*> ::= “**(**“ <*char*>* “**)**”

<*ws-char*> ::= <*space*> | ‘ISO 8859-1 code 9$_{10}$ ‘

Each field may appear only once within one mail notification smart message. Field names are case insensitive and linear whitespace characters are allowed after colons (:) as well as folding of header fields (see RFC822). It is recommended that fields that contain information significant to the user (i.e. from, subject, size and so on) are located as near the beginning of the message as possible. The service provider can freely define its own fields, as long as they match the field format defined in this specification (see <extension_field> in chapter 3).

If mail headers contain quoted-printable or base64 encoded fields, it is highly recommended that they are decoded and converted to the ISO 8859-1 charset [ISO_8859], which is the default character set in this message type.

3.3.2.2 Description of the Fields

3.3.2.2.1 Number of New Mail Messages

The <*new-amount*> field determines the number of new mail messages in the remote mailbox. The number of new mail messages is given here for backward compatibility with the older notification format that told only the amount of new mail messages and nothing more. There can be an optional comment after the number of new messages.

3.3.2.2.2 From

The from field determines the sender of the mail message. When the length of the message matters, it is highly recommended that only the sender’s email address (username@domain) is used. Otherwise, all address formats defined in RFC 822 can be used. This field is important to the user, so it is highly recommended that this field is included in the Mail Notification Smart Message.

3.3.2.2.3 Subject

The subject field determines the subject of the mail message. The length of the subject may have to be cut in order to compress the whole message into a single Short Message. If the subject is truncated, it SHOULD be indicated by three dots (…) at the end of the subject line. This field is important to the user, so it is highly recommended that this field is included in the Mail Notification Smart Message.

3.3.2.2.4 Size

This field determines the size of the body of the mail message in lines or kilobytes. This field contains information important to the user, so it is highly recommended that this field is included in the Mail Notification Smart Message. The headers of the mail message are ignored when calculating the size.

3.3.2.2.5 UID

These fields identify the mail message in the server. The name of this field defines the protocol to be used (IMAP or POP). This field provides one way to fetch or delete mail message without fetching all headers first.

3.3.2.2.6 Server ID

This field contains a hash value, which identifies the server to which the message arrived.

Before the hash value can be calculated, the characters must be converted to the ISO 8859-1 [ISO_8859] charset. The hash value is calculated as follows:

- a concatenated string of length *i*: *S* is formed which contains the user name, host name and port number (port number being a string which expresses the port number in decimal), in this order. S_i represents the ISO 8859-1 character value of the *i*th character of string *S*.
- a 32-bit number *N* is created by multiplying the ISO 8859-1 value of the first character of *S* by one, the second character value by two, and adding all of these together: $N = \sum_{j=1}^{i} jS_j$
- the hash value *H* is this number modulo 512: *H=N (mod 512)*

3.3.2.2.7 Attachments

This field determines the number of attachments in the mail message.

3.3.2.2.8 To

This field determines the recipients of the mail message (listed on the To: header line). This field can be truncated so that all To: addresses are not included in the Mail Notification Smart Message. This should be indicated by inserting a comment in parentheses as the last address of the To: field, for example ‘(...)’.

3.3.2.2.9 Cc

This field determines the recipients of the mail message (listed on the Cc: header line). This field can be truncated so that all Cc: addresses are not included in the Mail Notification Smart Message. This should be indicated by inserting a comment in parentheses as the last address of the Cc: field, for example ‘(...)’.

3.3.2.2.10 Date

This field determines the date when the mail message was sent.

3.3.2.2.11 Folder

This field determines the folder from which the mail is fetched. If this field is missing, the default folder is “INBOX”.

3.3.2.2.12 Sender

This field determines the sender of the mail message.

3.3.2.2.13 Reply-To

This field determines the address to which replies are sent. If this field is missing the replies are sent to the address on the From: line.

3.3.2.2.14 UID Validity

This field is used to determine if the UID is still valid or not.

3.4 Business Card

Business card information transfer is based on the Versit vCard specification. The vCard specification defines a format for electronic business cards. This format is suitable to be used as an interchange format between applications or systems, and it is independent of the method used to transport it [vCard].

The vCard reader is listening to on WDP port 9204 decimal (23F4 hexadecimal) or WTLS-secured WDP port 9206 decimal (23F6 hexadecimal).

The default usage of this format is on top of a 7-bit transmission channel; however, the vCard [vCard] specification enables transmission over both 7-bit and 8-bit transmission channels.

The set of supported fields and how they are interpreted may vary between different products. Please contact the product vendor for more information.

Please note: It is strongly recommended that the vCard parser is also able to handle messages that conform to vCard 3.0, which is the text/directory MIME type specification [RFC_2425], [RFC_2426].

An Informative example can be found in Appendix B.

3.5 CALENDAR

Calendar information transfer is based on the Versit vCalendar specification. The vCalendar specification defines a format for electronic calendaring and scheduling. This format is suitable to be used as an interchange format between applications or systems, and it is independent of the method used to transport it.

The vCalendar enables exchange of event and to-do types of calendaring and scheduling events. An event represents a scheduled amount of time on a calendar, and a to-do item represents an action-item or assignment. [vCalendar]

The vCalendar reader is listening to on WDP port 9205 decimal (23F5 hexadecimal) or WTLS-secured WDP port 9207 decimal (23F7 hexadecimal).

The default usage of this format is on top of a 7-bit transmission channel; however, the vCalendar [vCalendar] specification enables transmission over both 7-bit and 8-bit transmission channels.

The set of supported fields may vary between different products. Please contact the product vendor for more information.

An Informative example can be found in Appendix B.

3.6 RINGING TONES

The ringing tone format enables ringing tones to be sent to a wide variety of handsets. Depending on the handset implementation, it may be possible for the user to create ringing tones and then send them to other handsets.

The ringing tone format is handset independent, and describes only the audio related information. It enables transmission of both basic songs and temporary songs. A basic song is intended to be saved in a handset while the temporary songs can be used together with an alert router to implement message notification with a special ringing tone.

The ringing tone reader is listening to on NBS port 5505 decimal (1581 hexadecimal).

Ringing tones are designed to use an 8-bit communication channel. If only a 7-bit communication channel is available, the ringing tone format must be converted to 7-bit format. The conversion is 7-bit communication channel dependent and is not included in this specification.

The default character set, if not otherwise indicated, is ISO 8859-1 [ISO_8859].

The building blocks of the ringing tone messages are expressed as bit strings (base-2 values). These are concatenated and the resulting bit string is transferred in octets (8 bits in each byte).

An informative example can be found in Appendix B.

3.6.1 Syntax

<*ringing-tone-programming-language*> ::= <*command*>$^+$

<*command*> ::=

<*command-length*> <*command-part*>$^+$ |
<*command-end*>

<*command-length*> ::= ‘binary [00000001 .. 11111111], indicates how many command parts there are in the command. If necessary, filler bits are added to the end of each command part to ensure that the <*command-part*> is always octet-aligned.’

<*command-end*> ::= ‘binary [00000000]. This indicates the end of the ringing tone programming language.’

<*command-part*> ::=

<*ringing-tone-programming*> |
<*unicode*> |
<*cancel-command*> <*cancel-command-specifier*> |
<*sound*> <*sound-command-specifier*>

The Ringing Tone programming requires that the order of the command parts is as follows: <*ringing-tone-programming*>, [<*unicode*> ,] <*sound*>.

Table 3.6-1. Command-Part Encoding

Command	*Value (binary)*
<cancel-command>	*0000 101*
<ringing-tone-programming>	*0100 101*
<sound>	*0011 101*
<unicode>	*0100 010*

<cancel-command-specifier> ::= *<unicode>*

<sound-command-specifier> ::=

<basic-song-type> <basic-song> |
<temporary-song-type> <temporary-song> |
<midi-song-type> <midi-song> |
<digitised-song-type> <digitised-song> |
<polyphonic-song-type> <polyphonic-song>

Table 3.6-2. Song Type Encoding

Song type	*Value (binary)*
<basic-song-type>	*001*
<temporary-song-type>	*010*
<midi-song-type>	*011*
<digitised-song-type>	*100*
<polyphonic-song-type>	*110*

<basic-song> ::= *<song-title> <temporary-song>*

<song-title> ::= *<text-length> <text>*

<text> ::= *<default-char>*$^+$ | ; 'Unicode disabled'
<ISO-10646-char>$^+$; 'Unicode enabled'

<text-length> ::= 'binary [0000 .. 1111] indicating how many characters are used for the following text. For example, in the case of *Unicode*, this counts the number of 16-bit *Unicode* characters.'

<temporary-song> ::= *<song-sequence-length> <song-sequence>*

<song-sequence-length> ::= 'binary [00000000 .. 11111111]; Indicates how many song patterns follow.'

<song-sequence> ::= *<song-pattern>*$^+$

<song-pattern> ::=

<pattern-header> |
<pattern-header> <pattern-instruction>$^+$

<pattern-header> ::= *<pattern-header-id>* *<pattern-id> <loop-value> <pattern-specifier>*

Table 3.6-3. <*pattern-id*> Encoding

Pattern ID	*Value*
A-part	*binary 00*
B-part	*binary 01*
C-part	*binary 10*
D-part	*binary 11*

<*loop-value*> ::= ‘binary [0000 .. 1111]; Indicates how many times the pattern should be repeated. The value zero means no repeat. The value binary 1111 means infinite.’

<*pattern-specifier*> ::= <*already-defined-pattern*> | <*length-of-the-new-pattern*>

<*already-defined-pattern*> *::*= ‘binary [00000000]; This indicates that an already defined pattern is used again.’

<*length-of-the-new-pattern*> ::= ‘binary [00000001 .. 11111111]; Indicates how many pattern instructions there are in the song pattern. The value zero is illegal.’

<*pattern-instruction*> ::=

<*note-instruction*> | <*scale-instruction*> | <*style-instruction*> |
<*tempo-instruction*> | <*volume-instruction*>

<*midi-song-type*> ::= ‘MIDI data. Reserved for future extension.

<*digitised-song-type*> ::= ‘Digitised sound data. Reserved for future extension.’

<*polyphonic-song-type*> ::= 'Polyphonic song data. Reserved for future extension.'

Table 3.6-4. Instruction Encoding

Instruction	*Sequence*
<note-instruction>	*<note-instruction-id> <note-value> <note-duration> <note-duration-specifier>*
<scale-instruction>	*<scale-instruction-id> <note-scale>*
<style-instruction>	*<style-instruction-id> <style-value>*
<tempo-instruction>	*<tempo-instruction-id> <beats-per-minute>*
<volume-instruction>	*<volume-instruction-id> <volume>*

Table 3.6-5. Instruction ID Encoding

Instruction ID	*Value (binary)*
<pattern-header-id>	*000*
<note-instruction-id>	*001*
<scale-instruction-id>	*010*
<style-instruction-id>	*011*
<tempo-instruction-id>	*100*
<volume-instruction-id>	*101*

Table 3.6-6. Note-Value Encoding

Note value	*Value (binary)*
pause	*0000*
C	*0001*
Cis 'i.e. Des'	*0010*
D	*0011*
Dis 'i.e. Es'	*0100*
E	*0101*
F	*0110*
Fis 'i.e. Ges'	*0111*
G	*1000*
Gis 'i.e. As'	*1001*
A	*1010*
Ais 'i.e. B'	*1011*
H	*1100*
RESERVED	*1101*
RESERVED	*1110*
RESERVED	*1111*

Table 3.6-7. Note-Duration Encoding

Note duration	*Value (binary)*
Full-note	*000*
1/2-note	*001*
1/4-note	*010*
1/8-note	*011*
1/16-note	*100*
1/32-note	*101*
RESERVED	*110*
RESERVED	*111*

Table 3.6-8. Note-Duration-Specifier Encoding

Note duration specifier	*Value (binary)*
No special duration	*00*
Dotted note	*01*
Double dotted note	*10*
2/3 length	*11*

Table 3.6-9. Note-Scale Encoding

Note Scale	*Value (binary)*
Scale-1 (i.e. Note A is 440 Hz)	*00*
Scale-2 (i.e. Note A is 880 Hz), default	*01*
Scale-3 (i.e. Note A is 1.76 kHz)	*10*
Scale-4 (i.e. Note A is 3.52 kHz)	*11*

Table 3.6-10. Style-Value Encoding

Style	*Value (binary)*
Natural Style (rest between notes), default	*00*
Continuous Style (no rest between notes)	*01*
Staccato Style (shorter notes and longer rest period)	*10*
RESERVED	*11*

Table 3.6-11. Beats-per-Minute Encoding

Beats per Minute	*Value (binary)*
25, i.e., length of ¼ note = 2.40 sec.	*0000 0*
28, i.e., length of ¼ note = 2.14 sec.	*0000 1*
31, i.e., length of ¼ note = 1.90 sec.	*0001 0*
35, i.e., length of ¼ note = 1.70 sec.	*0001 1*
40, i.e., length of ¼ note = 1.51 sec.	*0010 0*
45, i.e., length of ¼ note = 1.35 sec.	*0010 1*
50, i.e., length of ¼ note = 1.20 sec.	*0011 0*
56, i.e., length of ¼ note = 1.07 sec.	*0011 1*
63, i.e., length of ¼ note = 0.95 sec, default	*0100 0*
70, i.e., length of ¼ note = 0.85 sec.	*0100 1*
80, i.e., length of ¼ note = 0.76 sec.	*0101 0*
90, i.e., length of ¼ note = 0.67 sec.	*0101 1*
100, i.e., length of ¼ note = 0.60 sec.	*0110 0*
112, i.e., length of ¼ note = 0.54 sec.	*0110 1*
125, i.e., length of ¼ note = 0.48 sec.	*0111 0*
140, i.e., length of ¼ note = 0.43 sec.	*0111 1*
160, i.e., length of ¼ note = 0.38 sec.	*1000 0*
180, i.e., length of ¼ note = 0.34 sec.	*1000 1*
200, i.e., length of ¼ note = 0.30 sec.	*1001 0*
225, i.e., length of ¼ note = 0.27 sec.	*1001 1*
250, i.e., length of ¼ note = 0.24 sec.	*1010 0*
285, i.e., length of ¼ note = 0.21 sec.	*1010 1*
320, i.e., length of ¼ note = 0.19 sec.	*1011 0*
355, i.e., length of ¼ note = 0.17 sec.	*1011 1*
400, i.e., length of ¼ note = 0.15 sec.	*1100 0*
450, i.e., length of ¼ note = 0.13 sec.	*1100 1*
500, i.e., length of ¼ note = 0.12 sec.	*1101 0*
565, i.e., length of ¼ note = 0.10 sec.	*1101 1*
635, i.e., length of ¼ note = 0.09 sec.	*1110 0*
715, i.e., length of ¼ note = 0.08 sec.	*1110 1*
800, i.e., length of ¼ note = 0.07 sec.	*1111 0*
900, i.e., length of ¼ note = 0.06 sec.	*1111 1*

Table 3.6-12. Volume Encoding

Volume	*Value (binary)*
tone-off	*0000*
level-1	*0001*
level-2	*0010*
level-3	*0011*
level-4	*0100*
level-5	*0101*
level-6	*0110*
level-7, default	*0111*
level-8	*1000*
level-9	*1001*
level-10	*1010*
level-11	*1011*
level-12	*1100*
level-13	*1101*
level-14	*1110*
level-15	*1111*

3.7 GRAPHICAL LOGOS AND ICONS

The OTA bitmap format enables graphical information to be sent to a wide variety of handsets. Depending on the handset implementation, it may be possible for the user to create graphical objects and then send them to other handsets. Various applications can use this information to create a more illustrative and attractive outlook for the application.

The OTA bitmap format is handset independent, and describes only the graphical information. In addition to the OTA bitmap format two different applications using OTA bitmap format are specified here.

The Calling Line Identification (CLI) icon is a bitmap, which can be attached to some number or numbers in the handset's phonebook (a caller group). When the caller is identified, the attached CLI icon is shown alongside other appropriate information such as the name and/or number of the caller. The CLI icon format doesn't contain any phonebook information so the linking between the phonebook entry and the CLI icon must be done in the handset.

The CLI Icon Reader is listening to on NBS port 5507 decimal (1583 hexadecimal).

The Operator Logo is a bitmap, which can be shown alongside the operator identification when the display of the handset is in idle mode. The Operator Logo format contains operator identification information. It is up to handset implementation how to this information is used.

The Operator Logo Reader is listening on NBS port 5506 decimal (1582 hexadecimal). Both the CLI Icon and Operator Logo require an 8-bit communication channel. If only a 7-bit communication channel is available, the message formats must be converted to 7-bit format. The conversion is 7-bit communication channel dependent and is not included in this specification.

The default character set is ISO 8859-1.

The OTA bitmap is usually used as a building block in other Smart Messages. As the OTA bitmap may contain palette information whose length cannot be known in advance, the encapsulating short message should either place the OTA bitmap at the end of the Smart Message or have a length field which can be utilised using parsing to determine the end of the OTA bitmap.

Informative examples of the OTA bitmap, CLI Icon and Operator Logo can be found in Appendix B.

3.7.1 OTA Bitmap Syntax

```
<ota-bitmap> ::= <header> <image- data> [<palette>]
<header>::=<infofield> [<extfield>]* <width> <height> <depth>
<infofield>::= 'Octet which is defined in Table 3.7-2'
<extfield>::= 'Octet which is defined in Tables 3.7-3 and 3.7-4'
<width> ::= <common-hex-digit> <common-hex-digit>
            [ <common-hex-digit> <common-hex-digit> ]
            ; 'Horizontal width of the bitmap in pixels. Field width depends on <infofield> bit 4.'
<height> ::= <common-hex-digit> <common-hex-digit>
            [ <common-hex-digit> <common-hex-digit> ]
            ; 'Vertical height of the bitmap in pixels. Field width depends on <infofield> bit 4.'
<depth> ::= <common-hex-digit> <common-hex-digit> ; 'Number of colours or grey shades'
```

<*image-data*> ::= <*main-image*> [<*animated- image*>]* ;'There can be 0 to 15 animated images'

<*main-image*> ::= 'Bitmap formed according to image data structure description below'

<*animated-image*> ::= 'Bitmap formed according to image data structure description below'

<*palette*> ::= 'Optional palette information format, not defined in this version of the specification. Present if <*infofield*> bit 5 is set.'

Table 3.7-1. Length of Header Parts

Data Type	*Length in Bits*
InfoField	*8*
ExtField	*8*
Width	*8 or 16*
Height	*8 or 16*
Depth	*8*

Table 3.7-2. Infofield description

InfoField, description	*Bit*
Concatenation flag, 1 = More will follow, 0 = Last octet	*7*
Compression, 1 = Compression, 0 = No compression	*6*
External palette, 1 = Used, 0 = Not used	*5*
Max. size of icon, 1 = 16 bit, 0 = 8 bit	*4*
Number of animated icons, msb	*3*
Number of animated icons	*2*
Number of animated icons	*1*
Number of animated icons, lsb	*0*

Table 3.7-3. Extended Field 0 description

ExtField 0, description	*Bit*
Concatenation flag, 1 = More will follow, 0 = Last octet	*7*
Bitmap Version information	*6*
Bitmap Version information	*5*
Bitmap Version information	*4*
Reserved	*3*
Reserved	*2*
Reserved	*1*
Reserved	*0*

Table 3.7-4. Extended Fields 1-15 description

ExtField 1..n, description	*Bit*
Concatenation flag, 1 = More will follow, 0 = Last octet	*7*
Reserved	*6*
Reserved	*5*
Reserved	*4*
Reserved	*3*
Reserved	*2*
Reserved	*1*
Reserved	*0*

3.7.1.1 Coding Rules

The format described here is a pure bitmap format; no data stream type handling is supported; the bitmap is assumed to be accessible as a whole for the decoder. Also no error correction or checksums are included for this format but the decoder is assumed to have access to error-free data.

In this version of the specification definition of the Palette, Compression and Animation scheme is left open to be defined in future, i.e. only static black and white images are fully defined in this version of the specification.

To ensure backward compatibility between today's simple black and white only devices and more advanced devices in the future the following decoding and encoding recommendations are given:

- If the bitmap fits into 8-bit boundaries, the encoder must use 8-bit presentation
- The Encoder takes care of colour reduction so that the first plane of the multicolour bitmap contains the black and white information.
- Pure black and white only capable decoders show only the first plane of the multicolour bitmaps.
- Decoders that are capable of processing exclusively non-animated bitmaps show only the first bitmap of the animation sequence.
- Decoders not capable of decompressing don't show the compressed icon at all.
- If the bitmap is bigger than display the bitmap is cropped so that the bitmap is shown starting from the upper left corner to the lower right corner of the display.

3.7.1.2 Image Data Structure

An image comprises <Depth> planes where the first plane is a black and white mask and other planes give additional colour information defined in the palette. Scanning of the pixels in a plane will be done row by row. Rows will be scanned from top to bottom and pixels inside the row will be scanned from left to right.

In order to fully utilise the available bandwidth no filler bits are used in the end of the row; fillers are put at the end of the whole bitmap if the size of the bitmap is not divisible by eight. Filler bits assume a value of zero.

3.7.2 CLI Icon Syntax

<cli-icon> ::= *<cli-header> <ota-bitmap>*

<cli-header> ::= *<cli-version> <cli-header-body>*

<cli-version> ::= "**0**" ;'Identifier for CLI icon version, current version is zero (0).

<cli-header- body> ::= 'With current version 0 this field will be skipped altogether because no additional data is supported'

3.7.3 Operator Logo Syntax

<operator-logo> ::= <operator-logo-header> <line-feed> <ota-bitmap>

<operator-logo-header> ::= <operator-logo-Version> <operator-logo-header-body>

<*operator-logo-version*> ::= “**0**” ;‘Identifier for Operator Logo version, current version is zero (0). '

<operator-logo-header-body> ::= <operator-information>

<*operator-information*> ::= <*mcc*> “**,**” <*mnc*> ; ‘For GSM only; other systems have another version’

<*mcc*> ::= <*common-hex-digit*> <*common-hex-digit*> <*common-hex-digit*> <*common-hex-digit*>
; ‘GSM Mobile Country Code, little-endian BCD,
filled with F_{16}’

<*mnc*> ::= <*common-hex-digit*> <*common-hex-digit*>

; ‘GSM Mobile Network Code, little-endian BCD,

filled with F_{16}’

3.8 MULTIPART MESSAGE FORMAT

The Multipart Message format can be used for sending picture messages and whole profiles to mobile phones. The Multipart Message reader is listening on NBS port 5514 decimal (158A hexadecimal).

Multipart messages are designed to use an 8-bit communication channel. If only a 7-bit communication channel is available, the multipart message format must be converted to 7-bit format. The conversion is 7-bit communication channel dependent and is not included in this specification.

The default character set, if not otherwise indicated, is ISO 8859-1 [ISO_8859].

3.8.1 Picture Message

A Picture Message is a message format that consists of a picture and a text part. In the message, the text and the picture may be in either order (i.e. text part first, or picture first). Neither part is optional. The Picture Message syntax is described in the Multipart Message syntax below.

An informative example can be found in Appendix B.

3.8.2 Downloadable Profile

A Downloadable Profile is a message format that allows the sending of user profiles to mobile phones. A Downloadable Profile consists of a profile name, a 'screen saver' and a ringing tone. These three parts may be in any order, and all parts are optional. The Downloadable Profile syntax is described in the Multipart Message syntax.

An informative example can be found in Appendix B.

3.8.3 Multipart Message syntax

Here is the syntax for the Picture Message and Downloadable Profile.

```
<Multipart-Message> ::= <version><item>+
<Version> ::= "0"     ; 'Identifier for version, current version is (ASCII-)zero "0" .
                              If it is not "0", stop the processing of the message.'
<Item> ::=             "00" < Item-length> <ISO-8859-1-char>*
                      | "01" <Item-length> <UNICODE-char>*
                      | "02" <Item-length> <OTA-bitmap>
                      | "03" <Item-length> <Ringing-tone-programming-language>
                      | "04" <Item-length> <Profile-name>
                      | "05" <Item-length> RESERVED
                      | "06" <Item-length> < Screen-saver>
                      | <common-hex-digit> <common-hex-digit> <Item-length> <default-char>*
                      ; 'Two hex characters which specify the item type, if not 00...06.
                              Tokens 07 ... FF are reserved. If one of these values is present,
                              the application may either stop processing or skip that item.'
```

<Item-length> ::= *<common-hex-digit> <common-hex-digit>*
<common-hex-digit> <common-hex-digit>
; 'Number of octets in the following field, in hexadecimal.
Note that there are two octets per Unicode character.'
<Profile-name> ::= <UNICODE-char>* ;' Name for Profile'
<Screen-saver> ::= *<OTA-bitmap>*

4. Appendix A: Reserved Port Numbers

4.1 Introduction

4.1.1 NBS port numbers

The NBS port numbers can be used in text headers as well as in binary headers (using the concept of User-Data-Headers). All future Smart Messaging implementations should use text headers only if interoperability with an old implementation is critical.

Text headers are positioned at the beginning of each short message. The NBS text header ends with a NBS-delimiter (space or line-delimiter) character. Text headers are specified as follows:

<*NBS-text-socket-header*> ::= <*NBS-keyword*> <*NBS-port-information*> [<*NBS-other-header*>] <*NBS-delimiter*>

<*NBS-delimiter*> *::= <space>*

<*NBS-keyword*> ::= "**//SCK**"

<*NBS-port-information*> ::=

<*NBS-short-destination-address*> |

<*NBS-short-destination-address*> <*NBS-short-source-address*> |

<*NBS-short-destination-address*> <*NBS-short-source-address*> <*NBS-SAR-information*> |

"**L**" <*NBS-long-destination-address*> |

"**L**" <*NBS-long-destination-address*> <*NBS-long-source-address*> |

"**L**" <*NBS-long-destination-address*> <*NBS-long-source-address*> <*NBS-SAR-information*>

<*NBS-other-header*> ::= “***//***” <*default-char-not-space*>*

<*NBS-short-destination-address*> ::= <*common-hex-digit*> <*common-hex-digit*>

; ‘Destination NBS port in ISO 8859-1 coded hexadecimal [00..FF], i.e. decimal [0..255]. When the short destination address presentation is used alone, then the source address of the message is defaulted to be the same as the destination address.’

<*NBS-short-source-address*> ::= <*common-hex-digit*> <*common-hex-digit*>

; ‘Source NBS port in ISO 8859-1 coded hexadecimal [00..FF], i.e. decimal [0..255].’

<*NBS-long-destination-address*> ::=

<*common-hex-digit*> <*common-hex-digit*> <*common-hex-digit*> <*common-hex-digit*>

; ‘Destination NBS port as a string of ISO 8859-1 characters, hexadecimal [0000..FFFF].’

<*NBS-long-source-address*> ::=

<*common-hex-digit*> <*common-hex-digit*> <*common-hex-digit*> <*common-hex-digit*>

; ‘Source NBS port as a string of ISO 8859-1 characters, hexadecimal [0000..FFFF].’

<*NBS-SAR-information*> ::=

<*NBS-SAR-reference*> <*NBS-SAR-total-fragments*> <*NBS-SAR-current-fragment*>

<*NBS-SAR-reference*> ::= <*common-hex-digit*> <*common-hex-digit*>

; ‘Concatenated message reference number as a string of ISO 8859-1 characters, hexadecimal [00..FF].’

<*NBS-SAR-total-fragments*> ::= <*common-hex-digit*> <*common-hex-digit*>

; ‘Concatenated message total fragment count as a string of ISO 8859-1 characters, hexadecimal [01..FF].’

<*NBS-SAR-current-fragment*> ::= <*common-hex-digit*> <*common-hex-digit*>

; ‘Concatenated message segment index as a string of ISO 8859-1 characters, hexadecimal [01..FF].’

For GSM, the binary headers use the concept of the User Data Header as defined in [GSM_03.40]. The information elements 00h for concatenation and 04h and 05h for port addressing are used.

4.1.2 WDP

The WDP (Wireless Datagram Protocol) is defined by the WAP Forum [WAP_WDP]. WDP specifications are available from the WAP Forum World Wide Web site.

4.2 RESERVED PORT NUMBERS

An up-to-date listing of reserved TCP/UDP port numbers is available from IANA (http://www.iana.org/). The following port numbers have been reserved from the NBS port number space.

Table 4-1. Reserved Port Numbers in NBS port number space

Port Number (decimal)	*Port Number (hexadecimal)*	*Application/Protocol*
0	*0*	*Default port for transparent (legacy) messages*
80	*50*	*WWW Server (HTTP)*
226	*E2*	*Business Card exchange (MIME vCard) Card reader*
228	*E4*	*Calendar Items (MIME vCalendar) Calendar reader*
5501	*157D*	*Compact Business Card reader (not specified in this document)*
5502	*157E*	*Service Card reader (not specified in this document)*
5503	*157F*	*Internet Access Configuration Data reader*
5504	*1580*	*<RESERVED>*
5505	*1581*	*Ringing Tone reader*
5506	*1582*	*Operator Logo*
5507	*1583*	*CLI Logo*
5508	*1584*	*Dynamic Menu Control Protocol (not specified in this document)l*
5509	*1585*	*<RESERVED>*
5510	*1586*	*<RESERVED>*
5511	*1587*	*Message Access Protocol*
5512	*1588*	*Simple Email Notification*
5513	*1589*	*<RESERVED>*
5514	*158A*	*<RESERVED>*
5580	*15CC*	*Character-mode WWW Access (TTML) (not specified in this document)*
5601	*15E1*	*<RESERVED>*
5603	*15E3*	*<RESERVED>*
8500	*2134*	*<RESERVED>*
8501	*2135*	*<RESERVED>*
8502	*2136*	*<RESERVED>*

Table 4-2. Smart Messaging-Related Port Numbers in TCP/UDP Port Number Space

Port Number (decimal)	*Port Number (hexadecimal)*	*Application/Protocol*
9204	*23F4*	*WAP vCard*
9205	*23F5*	*WAP vCalendar*
9206	*23F6*	*WAP vCard Secure*
9207	*23F7*	*WAP vCalendar Secure*

5. APPENDIX B: EXAMPLES

5.1 SMART MESSAGING EXAMPLES

5.1.1 Examples of Internet Access Configuration format

An example with minimum information to setup:

```
Your RNET access granted !
Iname:Company_access
Iuid:Username
Ipwd:secretpwd
Itel:+123456789012345
```

This message defines an Internet access point named Company_access and adds it to the Internet access point list. The username for this new access point is "Username”, the password is "secretpwd”and the phone number to call is "+123456789012345".

Typical situation:

```
Welcome !
Iname:Company
Iuid:User
Ipwd:secret
Itel:+123456789012345
Iip:123.123.123.123
Idns1:123.123.123.123
Idns2:123.123.123.124
```

This message defines an Internet access point named Company and adds it to the Internet access point list. The username for this new access point is "User”, the password is "secret” and the phone number to call is "+123456789012345". The IP address will be set as: 123.123.123.123, the Primary Nameserver as the same and the Secondary Nameserver as: 123.123.123.124.

A special situation that arises when no network mask autoconfiguration is available from the service provider. (The strings have to be shortened if all information is to be sent in one SMS):

```
Iname:Comp
Iuid:Us3
```

```
Ipwd:secr56
Itel:+123456789012345
Iip:123.123.123.123
Idns1:123.123.123.123
Idns2:123.123.123.124
Imsk:255.255.255.252
```

This message defines an Internet access point named Comp and adds it to the Internet access point list. The username for this new access point is "Us3”, the password is "secr56”and the phone number to call is "+123456789012345". The IP address will be set as: 123.123.123.123, the Primary Nameserver as the same and the Secondary Nameserver as: 123.123.123.124. The Network mask will be set as: 255.255.255.252.

Typical mail service information:

```
Note Text
Mname:Company_access
Muid:Username
Mpwd:secretpw
Madr:Username@serv.provid.net
Mrcv:imapserver.provid.net
Msnd:smtpserver.provid.net
Mpro:IM
```

The message requires that the Internet access point Company_access exists. The username is "Username" and the password is "secretpw". The defined settings are e-mail address (Madr), receiving host (Mrcv) and sending host (Msnd). Also the mailbox protocol that is wanted to use, will be defined (Mpro).

Sample script message:

```
Pname:the_script
Ptype:PPP
Pdata:15:
"" /r
```

This message installs a script in the Internet settings. The script will be named: the_script. The type can be used to give more information to the user about the sent script; the script type sent here is: "PPP". Pdata is followed by new script data and may be followed by a number defining the length of the script data.

WWW Bookmark message:

```
Nokia web page
Hname:hostname
Hurl:www.wheretogo.com
Hiap:Company_access
```

This message will create a Hotlist item called "hostname", which will retrieve the URL www.wheretogo.com. This Hotlist item will use a specific Internet access point called Company_access to retrieve the page.

Telnet connection:

```
Telnet connection
Tname:My connection
Tiap:My access
Thst:computer.company.com
Tdel:DEL
```

This short message will configure a connection to the Telnet application named "My connection”. The connection will use the Internet access point "My access” and will connect to the host, computer.company.com, using the backspace key as a delete.

Terminal connection:

```
Terminal configuration
Rname: My connection
Rtel:+123456789
Rdat:8
Rstp:1
Rech:On
Rini:ats35=0
Rdel:BS
```

This smart message will define a basic 8N1 ISDN connection named "My connection”.

5.1.2 Examples of Email Notification format

5.1.2.1 Examples of simple Email Notification format

Indication of 5 new email messages:

```
//MLAP11
5
```

Indication of 42 new email messages:

```
//SCKL1588
42
```

5.1.2.2 Examples of extended Email Notification format

5.1.2.2.1 Restricted mode version for legacy phones

```
//SCKL1588
1 new mail message
from: name.surname@nokia.com
subject: testing
size: 42 lines
```

This message notifies that one new mail message has been received. The sender is name.surname@nokia.com and the subject is "testing". The size of the message is 42 lines.

5.1.2.2.2 Restricted Mode Version for Smart Phones

```
//SCKL1588
1
subject:testing
from: name.surname@nokia.com
size:42 lines
date:10 Oct 1997 11:42 +0000
iuid:1234567891
sid: 007
```

This message notifies that one new mail message has been received. The sender is name.surname@nokia.com and the subject is "testing". The size of the message is 42 lines. The "iuid" (imap uid) field identifies the mail message in the server and the "sid" field identifies the server to which the new mail message has arrived.

5.1.2.2.3 Non-restricted Mode Version for a Legacy Phone

```
//SCKL1588
1 new mail message
From: Surname Name <name.surname@nokia.com>
Subject: test only, please ignore!
Size: 42 KB
att:1
To: Mark Smith <mark.smith@nmp.nokia.com>, Teemu Teekkari
<t123456@students.tut.fi> (…)
Cc: John Doe <avs@example.org>
Date:10 Oct 1997 +0000
sender: name.surname@nokia.com
reply-to: name.surname@nokia.com
fldr: user.smartme.this.folder.is.for.junk.mail
```

This message notifies that one new mail message has been received. The sender is name.surname@nokia.com and the subject is "test only, please ignore!". The size of the message is 42 kB and there is also one attachment included. Recipients´ e-mail addresses are defined (To and Cc) and also the folder from which the mail is fetched is determined.

5.1.2.2.4 Non-restricted Mode Example for Smart Phones

```
//SCKL1588
1 new mail message received
from: name.surname@nokia.com
subject: test only, please ignore
size: 4242 KB
date:10 Oct 1997 11:42 +0000
att: 42
to: Surname Name <name.surname@nokia.com>, Another User
<another.user@big.corporation.com>
cc: tester@mail.org
fldr: user.tester.this.folder.is.for.junk.mail
sender: name.surname@nokia.com
reply-to: name.surname@nokia.com
sid:007
iuid:1234567892
uidv:1020304050
```

This message notifies that one new mail message has been received. The sender is name.surname@nokia.com and the subject is "test only, please ignore". The size of the message is 4242 kB and there are 42 attachments included. Recipients´ e-mail addresses are defined (To and Cc) and also the folder from which the mail is fetched is determined. The "iuid" (imap uid) field identifies the mail message in the server and the "sid" field identifies the server to which the new mail message has arrived. The "uidv" field is used to determine if the UID is still valid or not

5.1.3 An example of Business Card format

This is an example of the Business Card (vCard) message format. The Business Card reader is listening on WDP port 9204 decimal (23F4 hexadecimal).

The Business Card message content is:

```
BEGIN:VCARD
VERSION:2.1
N:Smith;Mike
TEL;PREF:+55512345
END:VCARD
```

The following is a hex dump of the Business Card message format.

```
42 45 47 49 : 4E 3A 56 43 : 41 52 44 0D : 0A 56 45 52
53 49 4F 4E : 3A 32 2E 31 : 0D 0A 4E 3A : 53 6D 69 74
68 3B 4D 69 : 6B 65 0D 0A : 54 45 4C 3B : 50 52 45 46
3A 2B 35 35 : 35 31 32 33 : 34 35 0D 0A : 45 4E 44 3A
56 43 41 52 : 44 0D 0A
```

Table 5-1 describes the Business Card message format byte by byte.

Table 5-1. Description of the Business Card (vCard) example

Component	*Hex values*	*Description*
Vcard content	*42*	*B (ISO 8859-1 value)*
	45	*E (ISO 8859-1 value)*
	47	*G (ISO 8859-1 value)*
	49	*I (ISO 8859-1 value)*
	4E	*N (ISO 8859-1 value)*
	3A	*: (ISO 8859-1 value)*
	56	*V (ISO 8859-1 value)*
	43	*C (ISO 8859-1 value)*
	41	*A (ISO 8859-1 value)*
	52	*R (ISO 8859-1 value)*
	44 0D 0A 56 45 52…	*The rest of the vCard data*

5.1.4 An example of Calendar format

This is an example of Calendar (vCalendar) message format. The Calendar reader is listening to on WDP port 9205 decimal (23F5 hexadecimal). The Calendar message content is: A meeting in Portal on 6th of September 2000 from 10 am to 12 PM.

The Calendar message content is:

```
BEGIN:VCALENDAR
VERSION:1.0
BEGIN:VEVENT
DESCRIPTION:Steering Group meeting in Portal
DTSTART:20000906T100000
DTEND:20000906T120000
END:VEVENT
END:VCALENDAR
```

The following is a hex dump of the Calendar message format.

```
42 45 47 49 : 4E 3A 56 43 : 41 4C 45 4E : 44 41 52 0D
0A 56 45 52 : 53 49 4F 4E : 3A 31 2E 30 : 0D 0A 42 45
47 49 4E 3A : 56 45 56 45 : 4E 54 0D 0A : 44 45 53 43
52 49 50 54 : 49 4F 4E 3A : 53 74 65 65 : 72 69 6E 67
20 47 72 6F : 75 70 20 6D : 65 65 74 69 : 6E 67 20 69
6E 20 50 6F : 72 74 61 6C : 0D 0A 44 54 : 53 54 41 52
54 3A 32 30 : 30 30 30 39 : 30 36 54 31 : 30 30 30 30
30 0D 0A 44 : 54 45 4E 44 : 3A 32 30 30 : 31 30 39 30
36 54 31 32 : 30 30 30 30 : 0D 0A 45 4E : 44 3A 56 45
56 45 4E 54 : 0D 0A 45 4E : 44 3A 56 43 : 41 4C 45 4E
44 41 52 0D : 0A
```

Table 5-2 describes the vCalendar message format byte by byte.

Table 5-2. Description of the Calendar (vCalendar) example

Component	*Hex values*	*Description*
vCalendar content	*42*	*B (ISO 8859-1 value)*
	45	*E (ISO 8859-1 value)*
	47	*G (ISO 8859-1 value)*
	49	*I (ISO 8859-1 value)*
	4E	*N (ISO 8859-1 value)*
	3A	*: (ISO 8859-1 value)*
	56	*V (ISO 8859-1 value)*
	43	*C (ISO 8859-1 value)*
	41	*A (ISO 8859-1 value)*
	4C	*L (ISO 8859-1 value)*
	45	*E (ISO 8859-1 value)*
	4E	*N (ISO 8859-1 value)*
	44 41 52…	*The rest of the vCalendar data*

5.1.5 An example of Ringing tone format

The following is a hex dump of a sample Ringing tone message format. The ringing tone reader is listening on NBS port 5505 decimal (1581 hexadecimal).

```
02 4A 3A 51 : D1 95 CD D0 : 04 00 1B 20 : 55 05 90 61
05 60 55 85 : 50 54 85 40 : 82 08 49 90 : 00 00
```

The data has been encoded from the bit string which is explained in Table 5-3. The whole bit string must be divided into octets and then transferred to hex code.

Table 5-3. Ringing tone bit string

Bit string	*Description*	*Value*
00000010	*<command-length>*	*Number of command part presents*
01001010	*<ringing-tone-programming>*	*Command part 1 (with filler bit)*
0011101	*<sound>*	*Command part 2*
001	*<basic song type>*	
0100	*<song title length>*	*4 characters (ISO-8859-1)*
01110100	*the first character*	*t*
01100101	*the second character*	*e*
01110011	*the third character*	*s*
01110100	*the fourth character*	*t*
00000001	*<song sequence length>*	*1 song patterns*
000	*<pattern header>*	*pattern header id*
00	*<pattern id>*	*A-part*
0000	*<loop value>*	*no loop*
00001101	*<pattern specifier>*	*<length of the new pattern> 13 pattern instructions*
100	*<tempo instruction id>*	
10000	*<beats per minute>*	*160 (i.e. length of 1/4 note = 0,38 sec.)*
001	*<note instruction id>*	
0101	*<note value>*	*note E*
010	*<note duration>*	*¼ note*
00	*<note duration specifier>*	*no special duration*
001	*<note instruction id>*	
0110	*<note value>*	*note F*
010	*<note duration>*	*¼ note*
00	*<note duration specifier>*	*no special duration*
001	*<note instruction id>*	
1000	*<note instruction>*	*note G*
010	*<note duration>*	*¼ note*
00	*<note duration specifier>*	*no special duration*
001	*<note instruction id>*	
0101	*<note value>*	*note E*
100	*<note duration>*	*1/16 note*
00	*<note duration specifier>*	*no special duration*
001	*<note instruction id>*	
0101	*<note value>*	*note E*
011	*<note duration>*	*1/8 note*
00	*<note duration specifier>*	*no special duration*
001	*<note instruction id>*	
0101	*<note value>*	*note E*
010	*<note duration>*	*¼ note*
00	*<note duration specifier>*	*no special duration*
001	*<note instruction id>*	
0101	*<note value>*	*note E*
001	*<note duration>*	*½ note*

00	*<note duration specifier>*	*no special duration*
001	*<note instruction id>*	
0101	*<note value>*	*note E*
000	*<note duration>*	*full note*
00	*<note duration specifier>*	*no special duration*
010	*<scale instruction id>*	
00	*<note scale>*	*Scale-1 (i.e. Note A is 440 Hz)*
001	*<note instruction id>*	
0000	*<note value>*	*pause*
010	*<note duration>*	*¼ note*
00	*<note duration specifier>*	*no special duration*
010	*<scale instruction id>*	
01	*<note scale>*	*Scale-2 (i.e. Note A is 880 Hz), default*
001	*<note instruction id>*	
1001	*<note value>*	*Gis i.e. As'*
000	*<note duration>*	*full note*
00	*<note duration specifier>*	*no special duration*
0000000		*filler bits*
00000000	*<command end>*	*The end of the ringing tone data*

5.1.6 An example of OTA Bitmap format

The following is a hex dump of a sample of an OTA Bitmap. In this sample image, there are two black lines and several black dots.

```
00 48 0E 01 : FF FF FF FF : FF FF FF FF : FF 00 00 00
00 00 00 00 : 00 00 FF FF : FF FF FF FF : FF FF FF 00
00 00 00 00 : 00 00 00 00 : 10 F0 00 00 : 00 00 00 00
00 00 00 00 : 00 00 00 00 : 00 00 00 00 : 00 00 00 00
00 00 00 00 : 00 00 00 00 : 00 00 00 00 : 00 00 00 00
00 00 00 00 : 00 00 00 00 : 00 00 00 00 : 00 00 00 00
00 00 00 00 : 00 00 00 00 : 00 00 00 00 : 00 00 00 00
00 00 00 00 : 00 00 00 00 : 00 00 00 00 : 00 00 00 00
00 01
```

Table 5-4 describes the OTA bitmap format byte by byte.

Table 5-4. Description of the OTA Bitmap example

Component	*Hex values*	*Description*
Header	*00*	*Infofield*
	48	*Width of the OTA bitmap*
	0E	*Height of the OTA bitmap*
	01	*Depth, numbers of colours or grey shades*
Image data	*FF FF FF FF…*	*Main image*

5.1.7 An example of CLI Icon format

This logo is the default logo for the “family” caller group. The recipient associates the given logo with one of the caller groups in the phone. The CLI Icon reader is listening on NBS port 5507 decimal (1583 hexadecimal).

```
30 00 48 0E : 01 00 00 00 : 00 00 00 00 : 00 00 03 8F
71 E0 00 00 : 00 00 00 04 : 71 8E 30 00 : 00 00 00 00
09 21 24 18 : 00 00 00 00 : 00 0A 01 40 : 18 00 00 00
00 00 0A 01 : 40 18 00 00 : 00 00 00 08 : 01 00 18 00
00 00 00 00 : 04 01 80 30 : 00 00 00 00 : 00 02 03 40
60 00 00 00 : 00 00 01 06 : 20 C0 00 00 : 00 00 00 00
8C 11 80 00 : 00 00 00 00 : 00 58 0B 00 : 00 00 00 00
00 00 30 06 : 00 00 00 00 : 00 00 00 00 : 00 00 00 00
00 00 00
```

Table 5-5 describes the Cli Icon message format byte by byte.

Table 5-5. Description of the CLI Icon example

Component	*Hex values*	*Description*
Version	*30*	*The version, character '0'*
Header	*00*	*Infofield*
	48	*Width of the OTA bitmap*
	0E	*Height of the OTA bitmap*
	01	*Depth, numbers of colours or grey shades*
Image data	*00 00 00 00…*	*Main image*

5.1.8 An example of Operator logo format

The following is a hex dump of a sample Operator logo message format. The Operator logo reader is listening on NBS port 5506 decimal (1582 hexadecimal).

```
30 42 F4 50 : 0A 00 48 0E : 01 00 00 00 : 00 00 00 00
00 00 00 00 : 00 00 00 00 : 00 00 00 1E : 00 00 00 00
02 00 00 00 : 21 00 00 01 : 00 02 00 00 : 00 21 00 00
01 00 02 00 : 00 00 21 3C : 73 39 9C C2 : 07 1E 70 21
22 8A 05 22 : 82 08 A2 88 : 21 22 FA 3D : 22 82 08 A2
```

```
88 21 22 82 : 45 22 82 08 : A2 88 21 22 : 8A 45 22 82
08 A2 88 1E : 3C 72 3C 9C : 83 E7 1E 70 : 00 20 00 00
00 00 00 02 : 00 00 20 00 : 00 00 00 00 : 3C 00 00 00
00 00 00 00 : 00 00 00
```

Table 5-6 describes the Operator logo message format byte by byte.

Table 5-6. Description of the Operator logo example

Component	*Hex values*	*Description*
Version	*30*	*The version, character '0'*
Operator logo header	42F4	*Mobile Country Code, little endian BCD filled, with F_{16} (ie. 244 is 42F4 hex.)*
	50	*Mobile Network Code, little endian BCD filled, with F_{16} (ie. 05 is 50 hex.)*
Line feed	0A	*ISO 8859-1 value*
OTA Bitmap	00480E0100…	*OTA Bitmap data.*

5.1.9 An example of Picture Message format

The following is a hex dump of a sample picture message format. The picture message consists of two parts: a picture bitmap, and message text. Both parts start with a Type (**bold underlined**) and an item length (*italic underlined*).

```
30 00 00 04 : 54 65 73 74 : 02 01 00 00 : 48 1C 01 66
66 66 66 66 : 66 66 66 66 : 99 99 99 99 : 99 99 99 99
99 80 00 00 : 00 00 00 00 : 00 01 40 00 : 00 00 60 00
E0 00 02 40 : 00 00 0E 90 : 03 10 00 02 : 80 00 00 31
08 0C F3 B8 : 01 80 00 00 : 40 04 11 04 : 44 01 40 00
00 FF FE 2F : 8B 12 02 40 : 00 00 00 00 : 53 8C AA 02
80 00 00 00 : 00 62 89 C4 : 01 80 00 00 : 00 00 41 41
40 01 40 00 : 00 00 00 01 : 42 80 02 40 : 00 20 00 00
01 42 80 02 : 80 01 F0 00 : 00 00 A2 80 : 01 80 0F FE
00 00 00 A5 : 00 01 5F FF : FF FF FF FE : A5 7F FA 40
0A AA 00 00 : 00 55 00 02 : 82 01 50 04 : 40 01 5D 08
A1 88 10 24 : 80 00 40 FF : 02 01 40 41 : 00 01 00 03
AB E0 02 44 : 00 00 08 20 : 0D 55 58 82 : 80 10 14 40
00 1A AA AC : 01 80 00 00 : 00 00 35 55 : 56 01 40 01
00 00 80 6A : AA AB 02 40 : 00 00 00 00 : 55 55 55 02
80 00 00 00 : 00 00 00 00 : 01 99 99 99 : 99 99 99 99
99 99 66 66 : 66 66 66 66 : 66 66 66
```

Table 5-7 describes the picture message byte by byte.

Table 5-7. Description of the Picture Message example

Component	*Hex values*	*Description*
Version	*30*	*The version, character '0'*
Message's text part	***00***	*Type 00 (Message text ISO-8859-1-char)*
	00 04	*Item length (amount of the characters in the message)*
	54 65…	*Hex-coded (ISO-8859-1) characters*
Message's picture part	***02***	*Type 02 (OTA-bitmap)*
	01 00	*Item length (256 bytes)*
	00 48 1C 01 ...	*OTA bitmap format, 72x28 bitmap*

5.1.10 An example of a Downloadable Profile message format

The following is a hex dump of a sample downloadable profile format. The profile consists of three parts: the profile name, ringing tone and screen saver bitmap. Each part starts with a Type (**bold underlined**) and an item length (*italic underlined*).

```
30 04 00 10 : 00 53 00 4d : 00 53 00 20 : 00 54 00 65
00 73 00 74 : 03 00 90 02 : 4a 3a 65 89 : c9 a5 85 b9
89 bd c9 d4 : 04 00 c9 20 : a2 ac 22 d4 : 9c 81 a6 1a
42 8a b0 8b : 52 72 06 98 : 69 0a 26 c4 : 9c 69 a8 18
61 84 28 9b : 12 71 a6 a0 : 61 86 10 a2 : ac 22 d4 9c
81 a6 1a 42 : 8a b0 8b 52 : 72 06 98 69 : 26 98 a2 2c
26 c2 a8 26 : c2 2c 49 a2 : 10 61 86 18 : 62 88 b0 9b
0a a0 9b 0a : a0 9b 0a b4 : 9c 12 71 86 : 18 71 8a 22
c2 68 49 c6 : 28 9a 12 71 : 8a 26 84 9c : 61 a6 28 8b
09 b0 aa 09 : b0 aa 08 b0 : aa 52 69 8a : 22 c2 6c 2a
82 6c 22 c4 : 9a 20 00 06 : 01 00 00 48 : 1c 01 80 00
0b ff ff ff : d0 00 01 41 : 00 12 00 00 : 00 48 00 82
21 00 22 ff : ff ff 44 00 : 84 11 fc 42 : 80 00 01 42
3f 88 09 00 : 82 bf ff fd : 41 00 90 05 : 01 02 a0 00
05 40 80 a0 : 00 00 02 af : ff f5 40 00 : 00 00 00 02
a8 00 15 40 : 00 00 00 98 : 02 ab ff d5 : 40 3f 80 01
24 02 aa 00 : 55 40 24 80 : 11 24 02 aa : ff 55 40 24
80 39 24 02 : aa 81 55 40 : 20 80 54 c8 : 02 aa bd 55
40 20 80 10 : 00 02 aa a5 : 55 40 00 00 : 10 00 02 aa
a5 55 40 00 : 00 11 04 02 : aa bd 55 40 : 13 00 11 04
02 aa 81 55 : 40 24 80 11 : 24 02 aa ff : 55 40 24 80
01 24 02 aa : 00 55 40 24 : 80 01 fc 02 : ab ff d5 40
19 00 00 00 : 02 a8 00 15 : 40 00 00 00 : 00 02 af ff
f5 40 00 00 : 05 01 02 a0 : 00 05 40 80 : a0 09 00 82
bf ff fd 41 : 00 90 11 fc : 42 80 00 01 : 42 3f 88 21
00 22 ff ff : ff 44 00 84 : 41 00 12 00 : 00 00 48 00
82 80 00 0b : ff ff ff d0 : 00 01
```

Table 5-8 describes the profile byte by byte.

Table 5-8. Description of the Downloadable Profile example

Component	*Hex values*	*Description*
Version	*30*	*The Version, character '0'*
Profile name	***04***	*Type 04 (profile name in Unicode)*
	00 10	*Item length (16 bytes - i.e. 8 Unicode characters)*
	00 53, 00 4d ...	*8 Unicode characters, most significant byte first (zero because their Unicode codes are < 256)*
Ringing tone	***03***	*Type 03 (ringing tone)*
	00 90	*Item length (16*9 bytes)*
	02 4A 3A 65 ...	*Ringing tone data*
Screen saver	***06***	*Type 06 (screen saver)*
	01 00	*Item length (256 bytes)*
	00 48 1C 01 ...	*OTA bitmap format, 72x28 bitmap*

Smart Messaging Specification