SMART MESSAGING

SPECIFICATION

Revision 2.0.0

1999-05-17

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.

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: Third Party Developer Support
Smart Messaging

P.O. Box 68
FIN-33721 Tampere
Finland

Phone:	+358-10-5051
	(main switchboard)
Fax:	+358-10-505 6888
	(attn: Third Party Developer Support / Smart Messaging)
Email:	smart.messaging@nokia.com

< This page intentionally left blank >

Contents

Overview of changes from 1.0 to 2.0

A few minor errors in previous specification have been corrected, including some ambiguities in message format specifications. There are also many additions to Internet configuration messages. Extended Email Notification message has been introduced. WDP has been introduced as an alternative to NBS.

This specification does not address product-specific differences. Each product should have its own documentation, which is intended to be read in conjunction with this specification.

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 realization 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_8601	ISO 8601, Data elements and interchange formats—Information interchange—Representation of dates and times, International Organization for Standardization, June, 1988. ISO 8601, Technical Corrigendum 1, Data elements and interchange formats—Information interchange—Representation of dates and times, International Organization for Standardization, May, 1991. (http://www.iso.ch/)
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/)
NBS_SPEC	Narrowband Sockets Specification, Nokia Telecommunications, Intel Corporation, March, 1997. (http://www.intel.com/ial/nbs/)
RFC_822	RFC 822: Standard for the Format of ARPA Internet Text Messages, Dept. of Electrical Engineering, University of Delaware, David H. Crocker, August 13, 1982. (http://www.rfc-editor.org/)
RFC_1738	Network Working Group, Request for Comments 1738: Uniform Resource Locators (URL), CERN, T. Berners-Lee; Xerox Corporation, L. Masinter; University of Minnesota, M. McCahill, December,1994. (http://www.rfc-editor.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_2060	Network Working Group, Request for Comments 2060: Internet Message Access Protocol - VERSION 4rev1, University of Washington, M. Crispin, December, 1996. RFC 2060. (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 1.1. WAP Forum, 1999. (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

AC	Access Control
ASCII	American Standard Code for Information Interchange
BCD	Binary Coded Decimal
BNF	Backus-Naur Form
CI	Cell Identity
CLI	Calling Line Identification
DCS	Digital Cellular System
DMCP	Dynamic Menu Control Protocol
DTMF	Dual-Tone Multi-Frequency (tone)
FTP	File Transfer Protocol
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IANA	Internet Assigned Numbers Authority
IAP	Internet Access Point
IETF	Internet Engineering Task Force
IMAP	Internet Message Access Protocol
IMC	Internet Mail Consortium
IMEI	International Mobile Equipment Identity
ISDN	Integrated Services Digital Network
ISO	International Standards Organization
LAC	Location Area Code
LAN	1. Local Area Network, 2. Language
LGS	Localized GSM Services
MAP	Message Access Protocol
MIDI	Musical Instrument Digital Interface
MIME	Multipurpose Internet Mail Extensions
MS	Mobile Station
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
RT	Ringing Tone
SAP	Service Access Point
SAR	Segmentation And Re-assembly
SMS	Short Message Service
SMSC	Short Message Service Center
TCP	Transmission Control Protocol
TTML	Tagged Text Markup Language
UCS-2	Universal Multiple-Octet Coded Character Set, 16-bit representation
UDP	User Datagram Protocol
UI	User Interface
UID	Unique Identifier
UP	Unwired Planet
URL	Uniform Resource Locators
USSD	Unstructured Supplemental Services Data
UTC	Universal Time Coordinated
UU	User to User data
WAP	Wireless Application Protocol
WDP	Wireless Datagram Protocol
WTLS	Wireless Transport Layer Security
WWW	The World Wide Web

1. INTRODUCTION

1.1 BACKGROUND

The emphasis in cellular networks is changing from voice-only communication to a rich combination of voice and messaging. Where voice communication once was accompanied only with circuit-switched data services, now various kind of (narrow-band) 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, and operators. The quality of service the users receive depends crucially on how successful this interaction is.

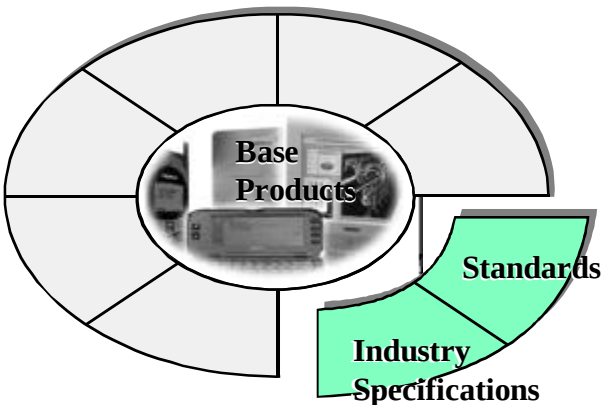

Figure 1-1. The whole product model

In order to efficiently utilize the messaging capabilities of networks such as GSM/DCS/PCS1900, 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 requirements evolve, the tools and interfaces must be able to grow as well.

This document is the first part in a family of documents that cover the Smart Messaging concept. The intention of this family of documents is to share up-to-date information within the telecommunication industry. Third parties are provided the means, i.e., the infrastructure on which they can build their services. Standards, were they official or agreed on within the industry, are an important part in the whole product model (Fig.1-1). They ensure on their part that the whole product fulfills the needs of the customer.

1.2 DOCUMENT OVERVIEW

This document is divided to four sections. First, the Introduction gives short overview on the basic ideology behind the document family.

Second, Smart Messaging architecture is described. In that section an overview is given on what is involved in building applications that seamlessly interface with the architecture. References are made to relevant specifications that will delve deeper into the technical details of 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 operators to provide services that work with a variety of handsets.

In the fourth section, relevant Protocol Specifications are presented. Based on these formats it is possible to create services that enable users to browse the Internet using handsets, or enable dynamic update of menu items in the handsets.

These sections are followed by an appendix on the reserved port addresses. This appendix presents information on the reserved socket port address space, and the port assignments for the currently defined "well-known" protocols.

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

2. SMART MESSAGING

2.1 SMART MESSAGING ARCHITECTURE

2.1.1 Narrow-Band Sockets

One of the main design criteria in Smart Messaging Architecture is to provide application developers with a flexible interface that hides communication channel idiosyncrasies. This is achieved by a joint effort of Nokia and Intel, in Narrow-Band Socket (NBS) specification. The NBS specification [NBS_SPEC] enables applications to access various network data bearer services using standard socket interface. This interface is available both in high-end handsets as well as in server elements in the network, making it possible for application developers to utilize their existing knowledge and code base. Over the Air (OTA) applications that rely on this architecture will benefit from the short time to market.

Depending on the needs of service providers, they may choose from several platforms to provide their services on. While smaller or specialized 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.

A solution in PC based systems is considered very important, as new markets are opened for small businesses, and PC platforms are challenging the performance of powerful workstations. Also the availability of a Winsock2 NBS interface in PC systems will help small businesses enter market.

For operators the main interests 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 centers 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 in NBS have been taken into use in WAP Forum's Wireless Datagram Protocol (WDP) [WAP_WDP]. In practice, the WDP protocol does not deviate far from NBS, and whenever NBS is mentioned as a suitable transport, it is possible 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 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 how Short Messages are used underneath these protocols . Peers which have a TCP/IP stack may use UDP to carry Smart Messages, just as if 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 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 utilizing 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 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 architecture is shown in Figure 2-1. The applications communicate with the NBS and WDP layers, which provide addressing capability and hide the packet boundaries of the underlying communication channel by providing a segmentation and reassembly (SAR) functionality. The upper interfaces of NBS and WDP may be implemented as sockets. The protocols 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 NBS and WDP are not dependent of the network technology, and thus are 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. It is also possible to build tailored systems in which vendor-specific formats are exchanged. In the Smart Messaging model, vendors reserve a port number for their own use, and the port number identifies the target application. 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 procedure on how to reserve a port number, please contact Nokia Third Party Developer Support. For more information on the address space and reserved ports, please refer to Appendix A: Reserved Port Numbers.

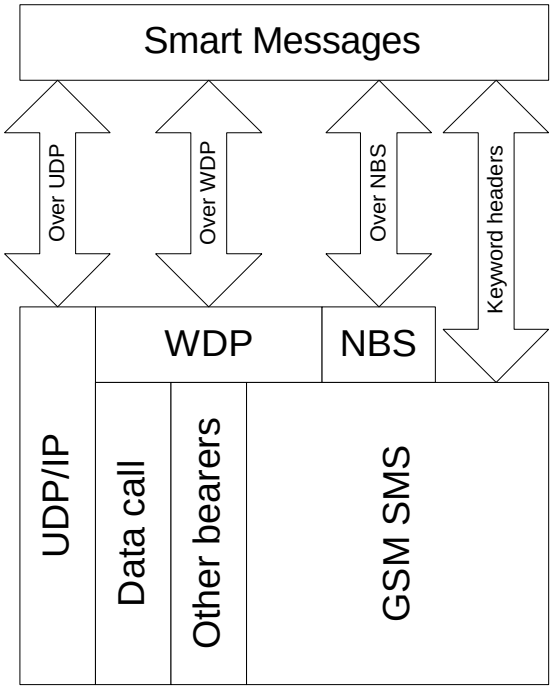

Figure 2-1. The Smart Messaging Architecture

2.2 MESSAGE FORMATS

The Smart Messaging architecture provides application developers with an extendible set of message formats . These formats enable any application to communicate with a wide variety of handsets. The addressing capability of Narrow-Band Sockets enables easy implementation of client/server applications; a set of narrow-band socket addresses are reserved to be solely used for exchanging the known message formats. A list of relevant socket addresses 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 service cards that enable easy access to touch tone (DTMF) based services.
- Sending or receiving Internet Access Configuration related information.
- Sending or receiving calendar items.
- Sending and receiving ringing tones and graphical information
- Enabling access to operator-specific supplementary services.
- Enabling access to a wide variety of value added services.

2.3 PROTOCOL SPECIFICATIONS

The Protocol Specifications section handles various kinds of protocols. These protocols, like the message formats, function in certain NBS ports. In this version of the document three protocols are made available. A short overview is given on the protocols next.

Dynamic Menu Control Protocol (DMCP)

The dynamic menu control protocol enables over the air update of the handset menu structure. Depending on the handset implementation, applicable parts of DMCP are supported. Typical to all implementations is the menu structure, which is divided to sub-menus. The control of these sub-menus can be given to authorized parties.

As users become familiar with the concept of dynamic menus, this capability is expected to be present in all handsets. The dynamic menus enable customization of the menu structure based on the needs of the user or based on the set of services that the operator provides.

When a menu item is selected, it may cause a number of different actions to take place. For example, a TTML based service can be started, a phone call can be initiated, or a message can be sent. Access to supplementary services can be made automatic, as users are shielded from the supplementary service string required to initiate a network service.

DMCP enables handsets to have any number of menus. The menus are divided into these basic categories: home operator, roaming operator, local, user's favorite and manufacturer-specific menus.

Tagged Text Markup Language (TTML)

The TTML specification enables users of legacy cellular phones, as well as users of browser capable phones to access services on the Internet. Enabling the legacy cellular phones to utilize the same concept enables the service providers to target a large user base from day one. This user base is likely to use keyword-based TTML service, whereas users with a browser capable phone will fully utilize the TTML forms available to access services.

Services accessible by legacy phones do not need any modifications within the handsets already on the market. Thus only a TTML server is needed in the network to start services. The TTML server is capable of acquiring information from the Internet. This makes service setup a short process, as most service providers are already well versed in setting up Internet based services.

Browser capable cellular phones provide the user with links to services, selection lists, text and numeric entry fields and the like. A combination of these plus free-form text give service providers a good ground to base their services on.

< 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. Backus-Naur Form (BNF) notation is used to describe the format of the messages. The BNF format and 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 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 < and > characters, and the symbols are written in *italics*.
- Textual definitions for non-terminal characters are enclosed between apostrophes (**'**).
- Operator "|" is used as a delimiter between multiple choices.
- Grouping of items is expressed by enclosing them in meta symbols { and }.
- Optional parts are enclosed in meta symbols [and].
- Kleene's "+" operator is supported, i.e., <A>$^{+}$ means repetition of non-terminal <A> from **1** to ∞ times.
- Kleene's "*" operator is supported, i.e., <A>* means repetition of non-terminal <A> from **0** to ∞ times.
- 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 used bearer service (for example GSM Short Message Service). Note that the bearer's native character set may be the only character set which 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 which 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.'

Dialing Definitions

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

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

'Character "**W**" / "**w**" commands the handset to wait for a dial-tone before dialing the rest of the sequence. Character "**P**" / "**p**" commands the handset to pause for a predefined interval prior to dialing 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 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*>

Addressing Definitions

<*common-nbs-port*> ::= "**;**" | 'value in range [0..65535] decimal' ; ' "**;**", i.e., No port is given, handle as text only'

<*common-source-nbs-port*> ::= <*common-nbs-port*> ; 'the NBS port from which the message originated'
<*common-destination-nbs-port*> ::= <*common-nbs-port*> ; 'the NBS port to which the message will be sent'
<*common-ms-isdn*> ::= "**;**" | <*common-phone-number*> ; ' "**;**" means that no source address is given'
<*common-destination-ms-isdn*> ::= <*common-ms-isdn*> ; 'the MS to send to'
<*common-source-ms-isdn*> ::= <*common-ms-isdn*> ; 'the MS from which the message originated'
<*common-smsc-ms-isdn*> ::= "**;**" | <*common-phone-number*>
; ' "**;**" means that no SMSC is given, use default if available'
<*common-source-smsc-ms-isdn*> ::= <*common-smsc-ms-isdn*>
; 'the SMSC through which message was received'
<*common-destination-smsc-ms-isdn*> ::= <*common-smsc-ms-isdn*> ; 'the SMSC to send through'

Miscellaneous Definitions

<*common-language*> ::=
<*default-char-not-ld*>$^{+}$ ["**-**" <*default-char-not-ld*>$^{+}$] | ; 'For TTML'
<*default-char-not-lf*>$^{+}$ ["**-**" <*default-char-not-lf*>$^{+}$] ; 'For all others'
; '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 TRANSFERRING CONFIGURATION MESSAGES

Smart Messaging has been originally designed to operate over GSM Short Message Service. GSM Short Messages are alphanumeric paging messages. When using keyword headers (that is, NBS or WDP is not used), all messages are transferred using GSM Class 1 Short Messages. If NBS or WDP are used, the bearer will be selected and used according to the respective protocol specifications.

Some Smart Messaging-capable devices also support alternative methods of transport. It is possible to transfer configuration messages and ringing tones as MIME objects over a suitable transfer protocol, such as HTTP (Hypertext Transfer Protocol) or email.

In order to transfer configuration messages or ringing tones over a MIME transport, the message is saved in a file having exactly the same syntax as a normal configuration message or a ringing tone. If the message contains textual information, the character set used must be the same character set which is used when transferring the Smart Messages in GSM Short Messages. This text file is then sent to the client using one of the following MIME types:

application/vnd.nokia.configuration-message

application/vnd.nokia.ringing-tone

It is also highly recommended that the files have the suffix

.ncm

.rng

respectively. These identify the file types as Nokia configuration messages or ringing tones in a file system which uses suffixes for file typing. The MIME transport should take into account that some Smart Messages may require a content encoding in order to be transferred over a 7-bit transmission channel.

For more information on MIME types in Nokia vendor tree (vnd.nokia), please contact Nokia Third Party Developer Support.

3.3 COMPACT BUSINESS CARD

A business card containing typical contact information can be transmitted over the air interface using compact business card format. Implementation of the service depends on the handset's capabilities. Where a legacy handset only shows the business card as a text message, high-end handset can extract the name and the number present in the business card and store them into the phone memory. With a smart phone the whole business card can be stored in the contact directory.

In order to ensure compatibility with existing products on the market, the compact business card should not be sent over NBS or WDP. However, receiving compact business cards should be enabled over NBS or WDP as well.

The compact business card reader is listening to NBS port 5501decimal (157D hexadecimal).

The default usage of this format is on top of 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.

3.3.1 Syntax

The syntax of the compact business card is based on <*line-feed*> delimited presentation. The content is formatted as follows:

<*compact-card-message*> ::= <*compact-card-keyword*> <*compact-card-body*>

<*compact-card-keyword*> ::= "**Business Card**" <*line-feed*> ; 'case-sensitive'

<*compact-card-body*> ::=

[<*name-field*>] <*line-feed*> ; 'name'

[<*company-field*>] <*line-feed*> ; 'company'

[<*title-field*>] <*line-feed*> ; 'title'

[<*phone-item*>] <*line-feed*> ; 'phone number'

[<*fax-item*>] <*line-feed*> ; 'fax number'

[<*email-address*>] <*line-feed*> ; 'email address'

[<*postal-address*>] <*line-feed*> ; 'street address'

<*name-field*> ::= <*default-char-not-lf*>* ; 'recommended contents: last-name first-name'

<*company-field*> ::= <*default-char-not-lf*>* ; 'company name'

<*title-field*> ::= <*default-char-not-lf*>* ; 'work title'

<*phone-item*> ::= ; 'one or more phone numbers'

"**tel**" <*space*> [<*phone-number-item*>] |

"**tel**" <*space*> [<*phone-number-item*>] <*line-feed*>

<*phone-item*>

<*phone-number-item*> ::= [<*phone-number-item-identifier*>] <*common-phone-number*>

<*phone-number-item-identifier*> ::=
"**(**" <*default-char-not-lf*>$^{+}$ "**)**" <*space*> ; 'identifier for the number, like "(**GSM)**"'

<*fax-item*> ::= ; 'one or more fax numbers'
"**fax**" <*space*> [<*phone-number-item*>] |
"**fax**" <*space*> [<*phone-number-item*>] <*line-feed*>
<*fax-item*>

<*email-address*> ::= <*default-char-not-lf*>* ; 'RFC822 based name@domain format '
<*postal-address*> ::=
<*default-char-not-lf*>* | ; 'one or more lines of postal address information'
<*default-char-not-lf*>* <*line-feed*>
<*postal-address*>

Any of the supplementary information fields (<*company-field*> , <*title-field*>, <*phone-item*>, <*fax-item*>, <*email-address*>, <*postal-address*>) may be absent, but as the syntax shows, the field separators (<*line-feed*>) must exist.

The telephone numbers and fax numbers may have optional field labels (*phone-number-item-identifier)* in parenthesis to identify what kind of number is displayed. More than one telephone or fax number can be presented with the compact business card format.

The postal address (<*postal-address*>) is always the last field in the format and can be more than one line.

3.3.2 Informative Examples

For example following messages are valid compact business cards:

```
Business Card
John Smith
Nokia Mobile Phones Ltd.
Design Engineer
tel +358 55 12345
fax +358 55 1234678
john.smith@nokia.com
P.O. Box 12
FIN-12345 TAMPERE

Business Card
John Miller
Some Company Ltd.

tel +44 1 2345678
tel (GSM) +44 123 123456
fax +44 1 1234567
fax (private) +44 123 456789
```


3.4 GENERIC BUSINESS CARD

Generic business card information transfer is based on 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 NBS port 226 decimal (E2 hexadecimal). When using a WAP stack, the vCard reader is listening to 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 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 Nokia Third Party Developer Support for more information.

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

Please note: If interoperability with old NBS-only implementations is not critical, vCards should be sent to the WDP ports.

3.4.1 Informative Examples

The following message is an example of a valid generic business card:

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

3.5 SERVICE CARD

Service cards enable handset users to access easily services that normally require them to remember and to enter touch tone sequences. Businesses can provide their clients with the service cards that ease the use of touch tone based services. In markets where touch tone based services are popular, like the US, service cards make greatly facilitate access to services as well as save people time and money. For example, rather than entering the whole bank account number every time an individual calls the bank to hear balance information, a service card entry can be selected to do this automatically. Other examples of service card possibilities include access to a help desk or customer service, or template that a business might offer containing user-defined personal information like a bank account number.

In order to ensure compatibility with existing products on the market, service cards, like compact business cards, should not be sent over NBS or WDP. However, as with compact business cards, it should be possible to receive them over NBS or WDP. It is important to implement the service card reader in order to enable message reception without keyword checking (i.e. NBS only solutions).

The service card reader is listening to NBS port 5502 decimal (157E hexadecimal).

The primary use of this format is on top of 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.

3.5.1 Syntax

The service card syntax is the following:

<*service-card-message*> ::= < *service-card-keyword*> <*service-card-body*>
<*service-card-keyword*> ::= "**Service Card**" <*line-feed*> ; 'case-sensitive'
<*service-card-body*> ::=
    <*provider-name-field*> <*line-feed*> ; 'name of the service provider'
    <*common-phone-number*> <*line-feed*> ; 'phone number'
    <*service-field*> <*line-feed*> ; 'service definition'

<*provider-name-field*> ::= <*default-char-not-lf*>* ; 'name of the service provider'
<*service-field*> ::=
    <*service-definition*> | ; 'one or more service definitions'
    <*service-definition*> <*line-feed*>
    < *service-field*>
<*service-definition*> ::= <*service-name*> "**:**"<*common-tel-char*>* ; 'the touch tone sequence to activate service'
<*service-name*> ::= <*default-char-not-lf*>*

The name (<*name-field*>) is the name of the service provider, and the phone number (<*common-phone-number*>) is its phone number. In service definition (<*service-field*>) the service name tells what service is

activated using this sequence. The DTMF-sequence will be sent to network when the service is selected. The number of service-fields in a card is not limited.

In the DTMF-sequence character “w” commands the handset to wait dial-tone before dialing the rest of the sequence. Character “p” commands the handset to pause a predefined interval (typically 0,5 seconds) prior to dialing the rest of the sequence.

3.5.2 Informative Examples

An example that shows how access to timetables and booking at Airline-number-one can be defined:

```
Service Card
Airline-number-one
+358 0 8823111
Timetable: 123w2323
Booking: 32p777
```

This example shows how to access National Bank phone bank account information:

```
Service Card
National Bank
+15559886262
Money Market Savings: 2p2p1p1234567890#1234#
Checking: 2p2p1p1234567890#1234#
```

3.6 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 cover also bookmarks, scripts, FTP, Telnet, terminal and WWW settings. Also GSM Short Message Centre numbers and Voice Mail numbers can be configured 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 over NBS or WDP, <*iap-compatibility-header*> should be used, and vice versa. When compatibility with older implementations is critical, the message should not be sent over NBS or WDP. Recipients should always be prepared to receive either variant of this message.

The Internet access point configuration data reader is listening to 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 which 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 remotely configured.

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 capabilities of a specific product, please contact Nokia Third Party Developer Support.

Please note: Service providers are recommended to take field length limits into consideration when designing configuration and setup of their devices. If a field identifier is present, but 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.

3.6.1 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>*

3.6.1.1 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 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 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'

3.6.1.2 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'

3.6.2 Informative Examples

An example with minimum information to setup:

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

Typical situation:

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

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:Prov
Iuid:Us3
```

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

Typical mail service information:

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

3.6.3 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> |
<ttml-info> |
<ftp-info> |
<internet-info> |
<telephone-info> |
<www-autofetch-info>

3.6.3.1 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 tunneling proxy IP address'
<iap-parameter-secure-port> ::= "**Ispo:**" *<common-digit>* ; 'SSL tunneling 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.'

3.6.3.2 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**".'

3.6.3.3 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'

3.6.3.4 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'

Sample script message:

```
Pname:script.dat
Ptype:PPPS
Pdata:47:This is my script line1
and this is the line2
```

3.6.3.5 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'

3.6.3.6 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})'

3.6.3.7 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 initialization 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'

3.6.3.8 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'

3.6.3.9 TTML Settings

TTML settings control the features of the device's TTML browser.

<*ttml-info*> ::= {<*ttml-access-point*> <*line-feed*> }$^{+}$

<*ttml-access-point*> ::=

"**Bsap:**" <*default-char-not-lf*>* <*line-feed*> ;'maximum title length 10 characters'

<*common-destination-ms-isdn*> "**/**" <*common-destination-smsc-ms-isdn*>

3.6.3.10 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-If*>* ; 'FTP server user name'

3.6.3.11 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-If*> ; ' for "Custom"

3.6.3.12 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'

3.6.3.13 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-If*>* ; 'location (URL) to fetch'
<*www-autofetch-body*> ::= "**Abdy:**" <*default-char-not-If*>* ; 'send this data as HTTP POST body part'
<*www-autofetch-headers*> ::= "**Ahdr:**" <*default-char-not-If*>* ; 'additional HTTP headers'

<*www-autofetch-iap*> ::= "**Aiap:**" <*default-char-not-If*>* ; ‘IAP to use’

<*www-autofetch-method*> ::= "**Amth:**" <*www-autofetch-method-parametres*> ; ‘which HTTP method to use’

<*www-autofetch-name*> ::= "**Aname:**" <*default-char-not-If*>* ; ‘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’

3.7 CALENDAR

Calendar information transfer is based on 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 NBS port 228 decimal (E4 hexadecimal). When using a WAP stack the vCalendar reader is listening to 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 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 Nokia Third Party Developer Support for more information.

Please note: If interoperability with old NBS-only products is not critical, the new WDP port numbers should be used.

3.7.1 Informative Examples

This section shows some examples of how calendar data is presented.

A meeting in Portal on 6th of September 1996 from 10 am to 12 PM.

```
BEGIN:VCALENDAR
VERSION:1.0
BEGIN:VEVENT
DESCRIPTION:Steering Group meeting in Portal
DTSTART:19960906T100000
DTEND:19960906T120000
END:VEVENT
END:VCALENDAR
```

A project meeting in Portal on 3rd of October 1997 from 9 am to 12 PM with a reminder 15 minutes before.

```
BEGIN:VCALENDAR
VERSION:1.0
BEGIN:VEVENT
DESCRIPTION:Project meeting in Portal
DTSTART:19971003T090000
DTEND:19971003T120000
AALARM:19971003T084500;;;
END:VEVENT
END:VCALENDAR
```

3.8 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 NBS port 5505 decimal (1581 hexadecimal).

Ringing tones require 8-bit communication channel configuration. Please refer to NBS specifications [NBS_SPEC] on how to setup 8-bit channel if you are using NBS.

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

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).

3.8.1 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 ensure that *<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 the following: *<ringing-tone-programming>*, [*<unicode>* ,] *<sound>*.

Table 3.8-1. 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> |

<digitized-song-type> <digitized-song>

Table 3.8-2. Song Type Encoding

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

<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 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 3.8-3. <*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. Value zero means no repeat. Value binary 1111 means infinite.’

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

<*already-defined-pattern*> ::= ‘binary [00000000]; This indicates that a 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. 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.

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

Table 3.8-4. 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 3.8-5. 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 3.8-6. 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 3.8-7. 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 3.8-8. 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 3.8-9. 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 3.8-10. 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 3.8-11. 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 3.8-12. 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*

3.9 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 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 is specified here.

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

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

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

Operator Logo Reader is listening to NBS port 5506 decimal (1582 hexadecimal). Both CLI Icon and Operator Logo require 8-bit communication channel configuration. If you are using NBS, please refer to NBS specifications [NBS_SPEC] on how to setup 8-bit channel.

The default character set is ISO 8859-1.

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 utilized using parsing to determine the end of the OTA bitmap.

3.9.1 OTA Bitmap Syntax

<*ota-bitmap*> ::= <*header*> <*image- data*> [<*palette*>]

<*header*>::=<*infofield*> [<*extfield*>]* <*width*> <*height*> <*depth*>

<*infofield*>::= 'Octet which is defined in table 3.8.1-2'

<*extfield*>::= 'Octet which is defined in tables 3.8.1-3 and 3.8.1-4'

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

Table 3.88-1-1. Length of Header Parts

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

Table 3.88-1-2. 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 3.88-1-3. 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 3.88-1-4. 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*

3.9.1.1 Coding Rules

The format described here is a pure bitmap format, no data stream type handling is supported; 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 an access to error free data.

In this version of the specification definition of 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 following decoding and encoding recommendations are given:

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

3.9.1.2 Image Data Structure

An image comprises <Depth> planes where first plane is black and white mask and other planes gives additional color 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 utilize the available bandwidth no filler bits are used in the end of the row, fillers are put end of the whole bitmap if the size of the bitmap is not divisible with eight. Filler bits assume value zero.

3.9.2 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’

3.9.2.1 Example of a CLI Icon Message

This logo is default logo for the “family” caller group. The recipient associates the given logo with one of the caller groups in the phone.

```
00                  Protocol Identifier
F5                  Data Coding Scheme
00                  Validity Period
8B                  User Data length
User Data (in hex):
06                  User Data Header Length
05                  IEI
04                  IEDL
1583                Destination Port
0000                Originator Port
00480E01            Ota Icon Header
0000000000000000    Ota Icon Data
00038F71E0000000
000004718E300000
0000000921241800
000000000A014018
00000000000A0140
1800000000000801
0018000000000004
0180300000000000
0203406000000000
00010620C0000000
0000008C11800000
00000000580B0000
0000000000300600
0000000000000000
000000000000
```

3.9.3 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*> <*validity-period*>

<*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}’

<*validity-period*> *::=* <*common-date*> ; ‘Expiration date’

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

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

The default usage of this format is on top of 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.

Extended Email notification chapter specifies the format of smart message which can be used to build mail headline view without establishing connection to the remote mailbox or just to notify the user of new mail, if 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 smart message contains 8-bit characters). This is achieved by cutting subject field so that the size limit is not exceeded and/or leaving unimportant fields out of mail notification. In the non-restricted mode, no size limit is defined. There are no obligatory fields in either mode.

3.10.1 Simple Email Notification

3.10.1.1 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.'

3.10.1.2 Informative Examples

Indication of 5 new email messages:

```
//MLAP11
5
```

Indication of 42 new email messages:

```
//SCKL1588
42
```

3.10.2 Extended Email notification

3.10.2.1 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 after colons (:) are allowed as well as folding of header fields (see RFC822). It is recommended that fields which contain information significant to user (I.e. from, subject, size and so on) are located as beginning of message as possible. Service provider can freely define its own fields, as long as they match to field format defined in this specification (see <extension_field> in chapter 3). If different field format is used, the different NBS port number MUST be used.

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

3.10.2.2 Description of the Fields

3.10.2.2.1 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 here for backward compatibility with older notification format which told only the amount of new mail messages and nothing more. There can be an optional comment after the number of new messages.

3.10.2.2.2 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 user, so it is highly recommended to include this field to the Mail Notification Smart Message.

3.10.2.2.3 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 in a single Short Message. If the subject is truncated, it SHOULD be

indicated by three dots (…) at the end of subject line. This field is important to user, so it is highly recommended to include this field to the Mail Notification Smart Message.

3.10.2.2.4 Size

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

3.10.2.2.5 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.

3.10.2.2.6 Server ID

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

Before hash value can be calculated, the characters must be converted to ISO 8859-1 [ISO_8859] charset. 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)*

3.10.2.2.7 Attachments

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

3.10.2.2.8 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 to the Mail Notification Smart Message. This should be indicated by inserting comment in parenthesis as last address of To: field, for example '(...)'.

3.10.2.2.9 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 to the Mail Notification Smart Message. This should be indicated by inserting a comment in parenthesis as last address of Cc: field, for example '(...)'.

3.10.2.2.10 Date

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

3.10.2.2.11 Folder

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

3.10.2.2.12 Sender

This field determines the sender of the mail message.

3.10.2.2.13 Reply-To

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

3.10.2.2.14 UID Validity

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

3.10.2.3 Informative Examples

3.10.2.3.1 Restricted mode version for legacy phones

//SCKL1588

1 new mail message

from: smart.messaging@nmp.nokia.com

subject: test only

size: 42 lines

3.10.2.3.2 Restricted Mode Version for Smart Phones

//SCKL1588

1

subject:test only

from: smart.messaging@nmp.nokia.com

size:42 lines

date:10 Oct 1997 11:42 +0000

iuid:1234567891

sid: 007

3.10.2.3.3 Non-restricted Mode Version for a Legacy Phone

//SCKL1588

1 new mail message

From: Messaging Smart <smart.messaging@nmp.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: smart.messaging@nmp.nokia.com

reply-to: smart.messaging@nmp.nokia.com

fldr: user.smartme.this.folder.is.for.junk.mail

3.10.2.3.4 Non-restricted Mode Example for Smart Phones

//SCKL1588

1 new mail message received

from: smart.messaging@nmp.nokia.com

subject: test only, please ignore

size: 4242 KB

date:10 Oct 1997 11:42 +0000

att: 42

to: Messaging Smart <smart.messaging@nmp.nokia.com>, Another User <another.user@big.corporation.com>

cc: tester@mail.org

fldr: user.tester.this.folder.is.for.junk.mail

sender: smart.messaging@nmp.nokia.com

reply-to: smart.messaging@nmp.nokia.com

sid:007

iuid:1234567892

uidv:1020304050

< This page intentionally left blank >

4. PROTOCOL SPECIFICATIONS

4.1 NOTATION

The notation used in content specifications uses the common elements presented in this section.

4.1.1 Common Elements

Refer to the Message Formats section, 3.1 Notation for a list of common elements that also apply to the Protocol Specifications section.

4.2 DYNAMIC MENU CONTROL PROTOCOL (DMCP)

The Dynamic menu control protocol enables over-the-air updates of the handset menu structure. Depending on the handset implementation, applicable parts of DMCP are supported. Typical to all implementations is the menu structure, which is divided to sub-menus that can grant control to authorized parties.

As users become familiar with the concept of dynamic menus, this capability is expected to be present in all handsets. The dynamic menus enable the customization of the menu structure, based on the needs of the user or based on the set of services that the operator provides.

When a menu item is selected, it may cause a number of different actions to take place. For example, a TTML based service can be started, a phone call can be initiated, or a message can be sent. Access to supplementary services can be made automatic, as the supplementary service string required to initiate a network service is hidden from the user.

The DMCP protocol operates at NBS port 5508 decimal (1584 hexadecimal).

DMCP enables handsets to have any number of menus. The menus are divided into the basic categories: home operator, roaming operator, local, user's favorite and manufacturer-specific menus. The usage of these categories is explained next.

The Operator Menus

The home and the roaming operator menus are two examples of the same concept. The operator menus are described here in terms of the home operator menu.

The home operator menu provides the operator with a powerful tool to personalize the handsets their subscribers use. This concept is an operator-specific menu structure in the handset that may vary from subscriber to subscriber, and can be updated over the air without any user assistance.

Access to the home operator menu is limited to the home operator. This enables the operator to provide service sets to their subscribers as a part of the basic service. Access to the roaming operator menu requires the home operator to authorize the roaming operator's server to send menu items to the handset.

The following menu example shows how the home operator services can be provided to users. The name of the home operator is also provided over the air to the handset.

```
MENU:
      Phone Settings
      Tele Services
            Call Tele Customer Service
            Download Ringing Tones
            New Offerings
            Enable Call Waiting
            Disable Call Waiting
      Local Services
      Personal Services
      Memory Settings
```

The following menu example shows how roaming operator services can be provided to the user. In this example both the home operator menu and the roaming operator menu exist in the handset. The roaming user has arrived in a new network, and menu items have been sent over the air to the phone:

```
MENU:
      Phone Settings
      Cellnet Services
      Radiolinja Services
            Roamer Services
            Local phonebook
      Local Services
      Personal Services
      Memory Settings
```

The Local Services Menu

The local services menu enables local service providers who are not necessarily operators to advertise their services. The list of local services depends on the location of the user and/or on the demographically selected group of users to which he belongs.

This facility provides an easy way to direct information or advertisements to handset users traveling through or within a geographical area. Technically, cell broadcasts can be used to send local service information to all users in a certain area. An important point is that as such information flow increases, handsets will be required to support functions that enable the end user to block this flow when it is not desired.

The local services menu is not persistent in the handset. In practice, the persistence rules may differ from handset to handset. One handset might clear the volatile service list as a handover takes place, whereas another does so only when the phone is turned off.

The following menu example shows how local services can be provided to users. As shown, a user is provided with ‘pointers’ to more information. By selecting local news, a news server sends the news headers to the handset, and based on the headers, main topics of the news article are then sent to the handset.

```
MENU:
      Phone Settings
      Radiolinja Services
      Local Services
            Join Wal Mart Direct-AD
            Traffic at Dallas I-75
            Weather Update
            Local News (courtesy of Nokia)
      Personal Services
      Memory Settings
```

The Personal Services Menu

The personal services menu is equivalent to the bookmark list of a web browser. The handset user has the ability to add and remove items from the menu. Menu items can be moved from other menus to the personal services menu. If a menu item addition is received by a handset, the result is handset implementation-specific. The recommended action is for the handset to query the user as to whether the item should be added or not.

The following example shows how the personal services menu can be provided to the user.

```
MENU:
      Phone Settings
      Radiolinja Services
      Local Services
      Personal Services
            Stock NOK A
            Send Email
            Find Restaurant
            Holiday Travel Inc.
      Memory Settings
```

4.2.1 The DMCP Protocol Description

DMCP defines how the menu structure of the cellular handsets can be dynamically changed. The protocol is provided as generic, being able to support any number of menus. However, in practice it depends on the cellular handset model and/or firmware version as to which menus are supported.

DMCP enables adding, removing, or querying existing menu items in a handset. The protocol enables implementation of menus with restricted access. In a handset, menu items are uniquely identified by each (<*menu-group-name*>, <*menu-item-name*>) pair. This means that all menu group names in a handset are unique, and each menu entry in a menu has a unique name.

4.2.1.1 Menu Group Names

A number of menu names have been reserved, in order to indicate which dynamic menu a command is intended for. The menu names and the menu descriptions are shown in Table 4.2-1.

Table 4.2-1. Reserved menu group names

Menu group name	*Description*
OPER <operator name>	*Home-operator accessible menu. Access is restricted in handsets, i.e., authorization list is enabled.*
ROAM <operator name>	*Roaming-operator accessible menu. Access is restricted in handsets, i.e., authorization list is enabled.*
LOCAL	*Local area services list. Services that any services provider may offer in the current geographical/ demographically divided area. Access to local services menu is open, but menu item tokens are used to implement weak authentication.*
USER	*User's favorite services list. Access is dependent on handset capabilities.*
MANUF	*Manufacturer-specific menu. Access is restricted in handsets, i.e., authorization list is enabled.*

4.2.1.2 Authorization Schemes

Access control to menus is based on two schemes. These schemes are called strong and weak authentication.

Strong authentication is based on checking the originator address of the received menu command. In this scheme only commands from authorized servers are accepted. In the case of GSM, the address pair

(<common-smsc-ms-isdn>, <common-ms-isdn>) is used to identify authorized servers. This system is as secure as the network is.

The list of authorized servers is stored in the handset, and it can be updated from an already authorized server. The first authorized server is entered by the user, or it is stored on SIM later on. After one authorized server is stored in the handset, no user intervention is required.

Weak authentication is available for all menus, regardless of the existence of an authorization list. Weak authentication is based on checking a specific <menu-item-token>, when a menu command is received that would affect an already existing menu item. The token can always be omitted from the menu command when this security feature is not needed. Depending on the handset implementation, the token is set dynamically for a menu item (recommended practice), or it can be preset and stored in the handset.

The weak authentication enables preventing hostile changing of menu items, for example, in the local services menu. Only a party with the token can change an existing menu item.

4.2.1.3 Protocol Primitives

The DMCP protocol primitives consist of request and response primitives (See Figure 4.2-1). Depending on handset implementation, only a subset of these primitives may be supported. The primitives are described in the following sections.

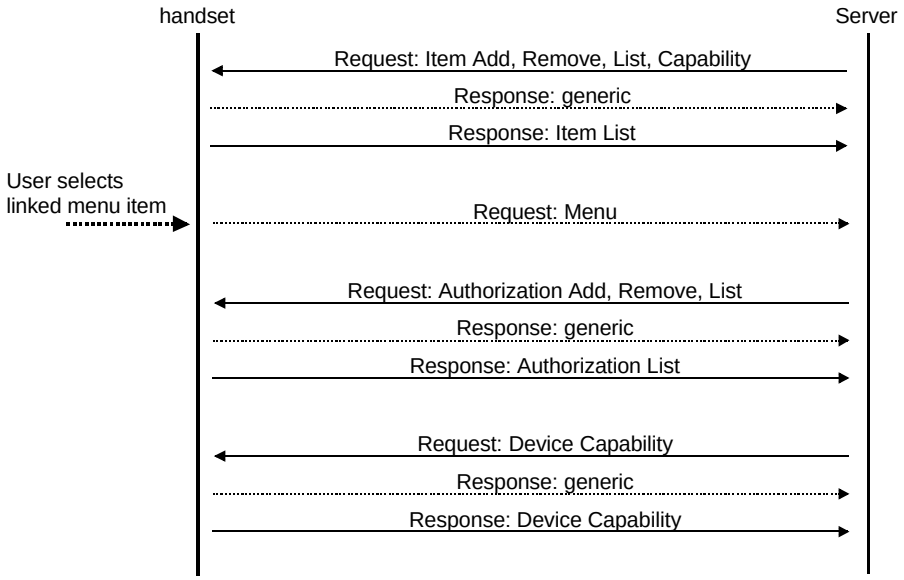

Figure 4.2-1. The DMCP protocol primitives

4.2.2 Item Primitive Definitions

There are four item primitives in the protocol: add, remove, list, and item capability.

4.2.2.1 Item Add

The item add command adds a new menu item to a menu in the handset.

Parameters: MENU-ITEM-TOKEN

MENU-GROUP-NAME

MENU-ITEM-NAME

MENU-ITEM-TYPE

MENU-ITEM-PROVIDER-INFORMATION

MENU-ITEM-HELP

MENU-ACTION-LIST

MENU-ITEM-TOKEN can be used by the server for authorization purposes. Once a menu item has been sent to a handset with a MENU-ITEM-TOKEN set, no command can be given to change or remove the existing MENU-ITEM without the same MENU-ITEM-TOKEN.

MENU-GROUP-NAME specifies the menu group into which the menu item with name MENU-ITEM-NAME is put. If a menu item with the same (MENU-GROUP-NAME, MENU-ITEM-NAME) pair exists and the server is authorized to update the menu item, then the existing menu item is replaced with the new one.

MENU-GROUP-NAME reserves the character ":" which has a special line-feed function inside the menu group name. For example, the menu group name "phone:settings" will displayed on a handset as 'phone' on the first line, and 'settings' on the second line.

MENU-GROUP-NAME reserves the character ";" which has a special function of indicating a change of hierarchy level in multiple hierarchy menus. This feature is implementation-specific and may be limited to manufacturer-specific menus in the handsets.

MENU-ITEM-NAME reserves the character ":" which has a special line-feed function inside the menu item name. For example, the menu item name "phone:volume" will displayed on a handset as 'phone' on the first line, and 'volume' on the second line.

MENU-ITEM-TYPE indicates the type of menu item. A menu item can be: normal, volatile, selected (volatile), or link. Depending on the handset implementation, only a subset of these may be supported, e.g. only the normal menu items.

NORMAL menu items are stored in the handset normally.

VOLATILE menu items are for multiple choice menu items which may vary from time to time. This type also indicates to the handset that a menu item may be discarded if the storage space limit in the handset is reached. Volatile type is intended to be used for menu items that are returned as a response to a link request; using linked menu items and volatile menu items enables the creation of multiple choice menus with a set of items that varies from time to time. The handset will always request the menu that currently is in effect.

SELECTED (volatile) menu item indicates "the currently active" volatile menu item. This can be used to indicate the current state of a multiple choice menu item.

LINK menu items indicate to the handset that it must request the volatile selection items from the server. This type should be used for any menu items that have a varying set of sub-items, because link type implies that a request is always made for sub-items. A link menu item causes an unsolicited menu response to be sent to the server requesting the volatile menu items.

MENU-ITEM-PROVIDER-INFORMATION is a text string which contents is decided by the provider. This parameter can be used to store in the handset information concerning the version of the menu item stored etc. This is unstructured field, and the accepted length of the field is handset implementation dependent.

MENU-ITEM-HELP is a text string which is shown when the user needs help regarding the current menu item. It is handset-specific, as to whether the handset automatically shows the help text when, for example, an idle timer expires. Menu item help reserves the character ":" which has a special line-feed function inside the help text.

MENU-ACTION-LIST is a list of actions. The actions can be of the following type: ACTION-SEND-MESSAGE, ACTION-CALL-CONTROL, or ACTION-PHONE-CONTROL. Any unknown type identifiers should be ignored and the content skipped.

ACTION-SEND-MESSAGE causes an embedded message to be sent in one of several methods. The method is dependent on the message type, which can be SMS, USSD, UU, URL, or NBS. It is handset implementation-specific as to which of the message types are supported. At minimum, the SMS and NBS message types should be supported.

Message type SMS causes the embedded message to be sent through the given SMSC (SMSC-MS-ISDN) to the given destination address (DESTINATION-MS-ISDN). The destination NBS port may be specified, in which case the handset will generate the NBS header in front of the embedded message to be sent.

If the SMSC address is replaced with ";", then the handset should use its default SMSC number to send the message.

If the destination MS address is replaced with ";", then the handset should prompt for an address when the action is executed.

If the NBS port is missing or it is replaced with ";", then no NBS header is generated, and the message is handled as text.

Message type USSD causes the embedded message to be sent to a given destination at a given NBS port.

Message type UU causes the phone to make a voice call to the given destination. When the circuit is opened, the handset will send the embedded information as user-to-user data to the destination. This can be used to automatically provide subscriber identity as a business card to the called party.

Message type URL causes the phone to process the given URL with its default browser. It is handset implementation-specific in regards to whether it has a browser that will be able to process the URL.

Message type NBS causes the phone to redirect the embedded message to the given local (*nbs-port*) NBS port. No action external to the phone takes place. If the destination NBS port is missing or is replaced with ";", then no NBS header is generated, and the message is handled as text. It is possible to redefine the source address for the NBS message, in order to enable a reply message to be sent over the air.

ACTION-CALL-CONTROL causes a call to be made in one of many methods. The method is dependent on CALL-CONTROL-TYPE, which can be VOICE, DATA, FAX, SS (supplementary services), AVD (alternate voice/data), AVF (alternate voice/fax), or SVD (sequential voice/data). Any unrecognized types must be ignored and skipped.

Call control type VOICE causes the handset to initiate a voice call to the given destination address.

Call control type DATA causes the handset to initiate a data call to the given destination address.

Call control type FAX causes the handset to initiate a fax call to the given destination address.

Call control type SS causes a supplementary service string to be sent to the network. Note: The GSM specifications require that if the SS string is not recognized, then it is sent as USSD.

Call control type AVD causes the handset to initiate alternate VOICE/DATA call (GSM: BS61). Additional parameter indicates does the call start with VOICE or with DATA.

Call control type AVF causes the handset to initiate alternate VOICE/FAX call (GSM: TS61). Additional parameter indicates does the call start with VOICE or with DATA.

Call control type SVD causes the handset to initiate sequential VOICE + DATA call (GSM: BS81).

ACTION-PHONE-CONTROL enables special control of phone functionality. The control is specified using the CONTROL-TYPE-IDENTIFIER, and parameters are provided using PHONE-CONTROL-TYPE-PARAMETER field.

Phone control type VAR causes the handset to place an internal value, such as IMEI code, or the language currently in use into a variable. The name of this variable can then be used in actions sent in sequence, to place the IMEI code or the preferred language into the body of the message to be sent. It is also possible to request network related information such as the network code (NC), location area code (LAC), or cell identity (CI) to be placed into a variable.

A vendor-specific extension of the variable definition is supported. It can be done by prefixing the vendor-specific internal variable name with "X-". It is recommended that prior to the actual variable name, the name of the vendor is placed between the prefix and the internal variable name, i.e., "X-Nokia-status1".

Phone control type MODE causes the handset to change state of a given internal function. If additional state is provided, the internal function is set to ON or OFF depending on this state information. If the state information is missing, then the MODE command toggles the internal state, advances it by one, triggers an event, and so forth, depending on the nature of the internal function.

4.2.2.2 Item Remove

The item remove command removes an item from a menu in the handset.

Parameters: MENU-ITEM-TOKEN

MENU-GROUP-NAME

MENU-ITEM-NAME

The item remove command causes a menu item identified by the (MENU-GROUP-NAME, MENU-ITEM-NAME) pair to be removed from the handset provided that authorization (based on MENU-ITEM-TOKEN and/or on authorization list) does not fail.

4.2.2.3 Item List

The item list command requests a handset to list the contents of a menu.

Parameters: MENU-ITEM-TOKEN

MENU-GROUP-NAME

When successful, this command will cause the handset to send a list of menu item names to the server from which this command was originally sent. The server can specify the menu for which the list of items is requested.

The item list command is a privileged command, and the availability depends on handset implementation. It is recommended that the item list command be available for manufacturer and operator menus.

When menu-item-token is given, only those items with a matching token will be returned.

The handset will use MENU-ITEM-LIST-RESPONSE format and send a message back to the server from which this command was originally sent. The MENU-ITEM-LIST-RESPONSE will contain the requested MENU-GROUP-NAME, and depending on the handset implementation, a line-feed delimited list of

- (MENU-ITEM-PROVIDER-INFORMATION, MENU-ITEM-NAME) pairs, or
- MENU-ITEM-NAMEs

Refer to section 4.2.5.5 Responses for further information.

4.2.2.4 Item Capability

The item capability command changes the capability of a menu item in the handset.

Parameters: MENU-ITEM-TOKEN

MENU-GROUP-NAME

MENU-ITEM-NAME

MENU-ITEM-CAPABILITY

The item capability command can be used to add or remove extra capabilities from a menu item. This command can be used to add an icon to a menu item. This command is currently under study, and should be ignored or skipped.

4.2.2.5 Unsolicited Menu Response

The unsolicited menu response can be used by the handset to request a linked sub-menu associated with the menu item indicated in the unsolicited menu response.

Parameters: MENU-GROUP-NAME

MENU-ITEM-NAME

This primitive can be used to request from the network the state of a function provided by the network. As the state of the network service is not known to the handset, it is explicitly requested from the network. Based on the sub-menu items received from the network, end user can request a change in the state of the service.

4.2.2.6 Unsolicited Menu Update Response

The unsolicited menu update response can be used by the handset to request a whole menu.

Parameters: MENU-GROUP-NAME

MENU-ITEM-PROVIDER-INFORMATION MENU-ITEM-NAME, …

This primitive enables end user to explicitly request update for a menu from the menu provider. Depending on the handset implementation, it may use in the unsolicited menu update response only MENU-GROUP-NAME, or the MENU-GROUP-NAME together with menu item information. The menu item information may consist of

- MENU-ITEM-PROVIDER-INFORMATION only,
- (MENU-ITEM-PROVIDER-INFORMATION, MENU-ITEM-NAME) pairs, or
- MENU-ITEM-NAME only

This enables a variety of implementations for the menu providers. For example,

(i) when the menu provider only provides one version of a menu named MENU-GROUP-NAME, then only that name is required in the unsolicited menu update response.

(ii) when the menu contains menu items of which only one version exists, then only the MENU-ITEM-NAMEs are required to be sent along with the MENU-GROUP-NAME.

(iii) when the menu provider has several versions of menu items in a menu, i.e., the menu items in the menu named MENU-GROUP-NAME have been updated separately, then the (MENU-ITEM-PROVIDER-INFORMATION, MENU-ITEM-NAME) pairs are required to be sent in order to be able to update the menu.

(iv) when the situation is as in case (iii), but the menu provider has assigned a unique ID in MENU-ITEM-PROVIDER-INFORMATION for each MENU-ITEM-NAME, it is possible to omit the MENU-ITEM-NAMEs, and only send the MENU-PROVIDER-INFORMATION.

4.2.3 Authorization Primitive Definitions

There are three authorization primitives in the protocol. They are add, remove, and list. In some cases the authorization primitives are not implemented in the handset. In such cases some other means are provided to enter server information into the handset. It is recommended that authorization lists be enabled for operator and manufacturer menus. Authorization primitives are implementation-specific as to how many entries the phone is capable of storing in the authorization list of each menu.

4.2.3.1 Authorization Add

The authorization add command adds new authorized server(s) for a menu in the handset.

Parameters: MENU-GROUP-NAME

MENU-AUTHORIZATION-SERVER-LIST

This command can be used to add one or more new authorized servers to the list of authorized servers for the given menu (MENU-GROUP-NAME). This command is privileged in the sense that it will be accepted only for a menu with an authorization list active, and when sent from an already authorized server.

The MENU-AUTHORIZATION-SERVER-LIST consist of one or more lines of information, each defining one authorized server.

For GSM/SMS, an authorized server is defined by the pair (MS-ISDN of a SMSC, MS-ISDN of the server).

4.2.3.2 Authorization Remove

The authorization remove command removes authorized server(s) from a menu in the handset.

Parameters: MENU-GROUP-NAME

MENU-AUTHORIZATION-SERVER-LIST

This command can be used to remove one or more authorized servers from the list of authorized servers within the given menu (MENU-GROUP-NAME). This command is privileged in the sense that it will be accepted only for a menu with an authorization list active, and when sent from an already authorized server.

4.2.3.3 Authorization List

The authorization list command requests the authorization list for a menu from the handset.

Parameters: MENU-GROUP-NAME

This command can be used to request the handset to send the list of privileged servers for a requested menu (MENU-GROUP-NAME). The availability of this command depends on the handset implementation. This command is privileged in the sense that it will be accepted only for a menu with an authorization list active, and when sent from an already authorized server.

The handset will use MENU-AUTHORIZATION-LIST-RESPONSE format and send a message back to the server from which this command was originally sent. The MENU-AUTHORIZATION-LIST-RESPONSE will contain the requested MENU-GROUP-NAME, and a line-feed delimited list of authorized servers.

4.2.4 Device Capability Primitive Definitions

4.2.4.1 Device Capability

The device capability command requests information from the handset.

Parameters: DEVICE-CAPABILITY-IDENTIFIER, …

This command can be used by a server to request device capability related information. This command is a privileged command.

DEVICE-CAPABILITY-IDENTIFIER is used to specify what kind of information is returned.

Field DEVICE-CAPABILITY-IDENTIFIER with the value LAN causes the handset to return the ID of the currently active (i.e., preferred) language. The format is based on Internet RFC 1766 [RFC_1766].

Field DEVICE-CAPABILITY-IDENTIFIER with the value INFO causes the handset to return the manufacturer name, and handset model. The rest of the information is manufacturer/model specific, for example, firmware version and date of the last firmware update could be indicated.

4.2.5 Syntax

All lines that start with an unknown identifier should be skipped. All identifiers, i.e. reserved words or commands, are written using uppercase characters. Menu and item names may contain any characters that are available in the current character set.

4.2.5.1 Requests

Menu control requests contain the optional menu meta information part and the control command list:

<menu-control-request> ::=

[*<menu-meta-header>*] "**BODY:**" *<line-feed>*

<menu-command-list>

Menu meta information can contain specific information on the character set encoding used in the control command list. The version number of the menu control protocol supported may also be present:

<menu-meta-header> ::= { *<menu-meta-information>* *<line-feed>* }$^{+}$

<menu-meta-information> ::=

"**CH:**" *<menu-charset-field>* | ; 'character set in use'

"**VER:**" *<menu-version-field>* ; 'version of the menu control protocol'

<menu-charset-field> ::= *<common-charset-field>*

; 'If not present, defaults to network default character set (in GSM, the GSM character set from GSM 3.38/3.40 specifications).'

<menu-version-field> ::= *<common-version-field>*

; 'Defaults to version 1.0 set of primitives, if version is higher, the version meta information must be present. If the version is 1.0, the meta information must NOT be present.'

<menu-command-list> ::=

<menu-control-command> *<line-feed>* |

<menu-control-command> *<line-feed>*

"**--**" *<line-feed>*

<menu-command-list>

<menu-control-command> ::=

"**IA:**" <*menu-item-token*> <*menu-item-add-field*> | ; 'add menu item'
"**IR:**" <*menu-item-token*> <*menu-item-remove-field*> | ; 'remove menu item'
"**IL:**" <*menu-item-token*> <*menu-item-list-field*> | ; 'list menu items'
"**IC:**" <*menu-item-token*> <*menu-item-capability-field*> | ; 'change item capability'
"**AA:**" <*menu-authorization-add-field*> | ; 'add trusted server'
"**AR:**" <*menu-authorization-remove-field*> | ; 'remove trusted server'
"**AL:**" <*menu-authorization-list-field*> | ; 'list trusted servers'
"**DC:**" <*device-capability-request-field*> ; 'request capability'

4.2.5.2 Item Commands

<*menu-item-token*> ::= <*default-char-not-lf*>* <*line-feed*>; 'empty token or unique token to change menu item.'

<*menu-item-add-field*> ::=

<*menu-group-name*> <*line-feed*>
<*menu-item-name*> <*line-feed*>
<*menu-item-type*> <*space*> <*item-provider-information*> <*line-feed*>
<*menu-item-help*> <*line-feed*>
<*menu-action-list*>

<*item-provider-information*> ::= <*default-char-not-lf-or-space*>*

; ' The <*item-provider-information*> may contain version, language, etc. information that the menu provider wants to associate with this instance of the menu item. The information is also received by the provider with the item list command.'

<*menu-group-name*> ::=

"**OPER**" [<*space*> <*operator-name*>] |
"**ROAM**" [<*space*> <*operator-name*>] |
"**LOCAL**" |
"**USER**" |
"**MANUF**" |
<*default-char-not-lf*>+

<*operator-name*> ::= <*default-char-not-lf*>* ; 'name of the operator'

<*menu-item-name*> ::= <*default-char-not-lf*>*

<*menu-item-type*> ::=

"**N**" | ; 'normal (default)'
"**L**" | ; 'linked item - selection items will be requested'
"**V**" | ; 'volatile selection item'
"**S**" ; 'selected (volatile) item'

<*menu-item-help*> ::= <*default-char-not-lf*>* ; 'help text associated with the menu item'

<*menu-action-list*> ::=
<*menu-action*> |
<*menu-action*> <*line-feed*>
<*menu-action-list*>

<*menu-action*> ::=
<*action-send-message*> |
<*action-call-control*> |
<*action-phone-control*> |
<*default-char-not-lf*>* ; 'should ignore all other types'

<*menu-item-remove-field*> ::=
<*menu-group-name*> <*line-feed*>
<*menu-item-name*>

<*menu-item-list-field*> ::= <*menu-group-name*> ; 'list menu items in menu'

<*menu-item-capability-field*> ::=
<*menu-group-name*> <*line-feed*>
<*menu-item-name*> <*line-feed*>
<*menu-item-capability*>

SEND MESSAGE ACTION

<*action-send-message*> ::=
"**M**" <*space*> <*message-type-identifier*> <*line-feed*>
<*message-address-field*> <*line-feed*>
<*common-information-content*>

<*message-type-identifier*> ::= 'depends on the network'
for GSM: <*message-type-identifier*> ::=
"**SMS**" | "**USSD**" | "**UU**" | "**URL**" | "**NBS**" | <*default-char-not-lf*>*

<*message-address-field*> ::= 'depends on the message-type-identifier'
for GSM/ "**SMS**": <*message-address-field*> ::=
<*common-smsc-ms-isdn*> **"/"** <*common-destination-ms-isdn*>
[**"/"** <*common-nbs-port*>]
for GSM/ "**USSD**":<*message-address-field*> ::= 'RESERVED'
for GSM/ "**UU**": <*message-address-field*> ::=
<*common-destination-ms-isdn*> [**"/"** <*common-nbs-port*>]
for GSM/ "**URL**": <*message-address-field*> ::= 'full URL as specified in RFC 1738 [RFC_1738]'
for GSM/ "**NBS**": <*message-address-field*> ::=
<*common-destination-nbs-port*> **"/"**
<*common-source-smsc-ms-isdn*> **"/"**

<*common-source-ms-isdn*> [**"/"** <*common-source-nbs-port*>]

CALL CONTROL ACTION

<*action-call-control*> ::=

"**C**" <*space*> <*call-control-type*> <*line-feed*>
<*call-control-type-modifier*>

<*call-control-type*> ::= "**VOICE**" | "**DATA**" | "**FAX**" | "**SS**" | "**AVD**" | "**AVF**" | "**SVD**" | <*default-char-not-lf*>+

<*call-control-type-modifier*> ::= 'depends on the <*call-control-type*>'

for GSM/ "**VOICE**": <*call-control-type-modifier*> ::= <*common-destination-ms-isdn*>
for GSM/ "**DATA**": <*call-control-type-modifier*> ::= <*common-destination-ms-isdn*>
for GSM/ "**FAX**": <*call-control-type-modifier*> ::= <*common-destination-ms-isdn*>
for GSM/ "**SS**": <*call-control-type-modifier*> ::= <*supplementary-service-string*>

for GSM/ "**AVD**": <*call-control-type-modifier*> ::= ;'Alternate: voice or data'
<*call-control-type-alternate-start-mode*>
<*common-destination-ms-isdn*>

for GSM/ "**AVF**": <*call-control-type-modifier*> ::= ;'Alternate: voice or fax'
<*call-control-type-alternate-start-mode*>
<*common-destination-ms-isdn*>

for GSM/ "**SVD**": <*call-control-type-modifier*> ::= ;'Sequential: voice followed by data'
<*common-destination-ms-isdn*>

<*supplementary-service-string*> ::= <*common-tel-char*>*

<*call-control-type-alternate-start-mode*> ::=

"**V**" | ;'Voice, available in "**AVD**", "**AVF**", and "**SVD**".'
"**D**" | ;'Data, available in "**AVD**", and "**SVD**".'
"**F**" ;'Fax, Available in "**AVF**".'

PHONE CONTROL ACTION

<*action-phone-control*> ::=

"**P**" <*space*> <*phone-control-type-identifier*> <*line-feed*>
<*phone-control-type-parameter*>

<*phone-control-type-identifier*> ::=

"**VAR**" | ; 'set variable to be used in other actions'
"**MODE**" | ; 'change phone internal state'
<*default-char-not-lf*>*

<*phone-control-type-parameter*> ::= ‘depends on the <*phone-control-type-identifier*>’

for GSM/ "**VAR**": <*phone-control-type-parameter*> ::=
<*phone-control-variable-name*> <*space*>
<*phone-control-internal-data-name*> <*space*>
<*phone-control-internal-data-value*>

for GSM/ "**MODE**":<*phone-control-type-parameter*> ::=
<*default-char-not-lf-or-space*> [<*space*> <*phone-control-mode-state*>]

<*phone-control-variable-name*> ::= "**%**" <*default-char-not-lf-or-percent*> "**%**"

<*default-char-not-lf-or-percent*> ::= ‘Any character in the default character set (or in character set explicitly indicated) except <*line-feed*> or "**%**". In GSM, the GSM character set from 3.38/3.40.’

<*phone-control-internal-data-name*> ::=
"**CI**" | ; ‘Cell identity’
"**IMEI**" | ; ‘IMEI of the handset’
"**LAC**" | : ‘Location area code’
"**LAN**" | : ‘The preferred language of the user’
"**NC**" | : ‘Network code’
<*default-char-not-lf*>*

<*phone-control-internal-data-value*> ::= ‘depends on the <*phone-control-internal-data-name*>’

for "**CI**": <*phone-control-internal-data-value*> ::= <*common-hex-digit*>$^{+}$
; ‘Up to 8 hexadecimal digits of CI information’

for "**IMEI**": <*phone-control-internal-data-value*> ::= <*common-digit*>$^{+}$
; ‘Up to 20 digits of IMEI information’

for "**LAC**": <*phone-control-internal-data-value*> ::= <*common-hex-digit*>$^{+}$
; ‘Up to 4 hexadecimal digits of LAC information’

for "**LAN**": <*phone-control-internal-data-value*> ::= <*common-language*>
; ‘Preferred language’

for "**NC**": <*phone-control-internal-data-value*> ::= <*common-hex-digit*>$^{+}$
; ‘Up to 6 hexadecimal digits of NC information’

<*phone-control-mode-state*> ::= “**ON**” | “**OFF**” ; ‘set mode ON/OFF. IF this is missing, then TOGGLE’

<*menu-item-capability*> ::= ‘RESERVED’

4.2.5.3 Authorization Commands

<*menu-authorization-add-field*> ::=
<*menu-group-name*> <*line-feed*> ; 'add authorization'
<*menu-authorization-server-list*>

<*menu-authorization-server-list*> ::=
<*menu-server-address*> | <*menu-server-address*> <*menu-authorization-server-list*>

<*menu-server-address*> ::= 'Depends on the addressing scheme of the system'
for GSM/ "**SMS**": <*menu-server-address*> ::=
<*common-smsc-ms-isdn*> **"/"** <*common-ms-isdn*> <*line-feed*>

<*menu-authorization-remove-field*> ::= ; 'remove authorization'
<*menu-group-name*> <*line-feed*>
<*menu-authorization-server-list*>

<*menu-authorization-list-field*> ::= <*menu-group-name*> ; 'name of the group to check authorizations for'

4.2.5.4 Capability Commands

<*device-capability-request-field*> ::=
<*device-capability-identifier*> |
<*device-capability-identifier*> <*line-feed*>
<*device-capability-request-field*>

<*device-capability-identifier*> ::=
"**LAN**" | ; 'request preferred language'
"**INFO**" | ; 'request device information; manufacturer, model'
<*default-char-not-lf*>$^{+}$

4.2.5.5 Responses

<*menu-control-response*> ::=
[<*menu-meta-information*>] "**BODY:**" <*line-feed*>
<*menu-control-command-response*> <*line-feed*>

<*menu-control-command-response*> ::=
"**RG:**" <*menu-generic-response*> |
<*menu-item-list-response*> |
<*menu-authorization-list-response*> |
<*unsolicited-menu-response*> |
<*unsolicited-menu-update-response*> |
<*device-capability-response*>

<*menu-generic-response*> ::= "**OK**" | "**ERROR**" | "**BAD-VERSION**" | "**BAD-TOKEN**" | "**BAD-SERVER**"

<*menu-item-list-response*> ::=
"**RIL:**" <*menu-group-name*> <*line-feed*>
<*menu-items*>

<*menu-items*> ::=
<*item-provider-information*> <*space*> <*menu-item-name*> |
<*item-provider-information*> <*space*> <*menu-item-name*> <*line-feed*>
<*menu-items*>

<*menu-authorization-list-response*> ::=
"**RAL:**" <*menu-group-name*> <*line-feed*>
<*menu-authorization-server-list*>

<*unsolicited-menu-response*> ::= ; 'handset requires linked menu item'
"**RM:**" <*menu-group-name*> <*line-feed*>
<*menu-item-name*> <*line-feed*>

<*unsolicited-menu-update-response*> ::= ; 'handset requires menu update'
"**RU:**" <*menu-group-name*> <*line-feed*>
{ [<*item-provider-information*>] [<*space*> <*menu-item-name*>] <*line-feed*> }*

<*device-capability-response*> ::=
"**RDC:**" <*line-feed*>
<*device-capability-response-parameter-list*>

<*device-capability-response-parameter-list*> ::=
<*device-capability-identifier*> <*space*> <*device-capability-value*> |
<*device-capability-identifier*> <*space*> <*device-capability-value*> <*line-feed*>
<*device-capability-response-parameter-list*>

<*device-capability-value*> *::=* 'Depends on the <*device-capability-identifier*>'
for "**LAN**": <*device-capability-value*> ::= <*common-language*> ; 'Preferred language'
for "**INFO**": <*device-capability-value*> ::=
<*manufacturer-info*> <*line-feed*>
<*model-info*> <*line-feed*>
<*manufacturer-specific-info*>

<*manufacturer-info*> ::= <*default-char-not-lf*>* ; 'Manufacturer name.'
<*model-info*> ::= <*default-char-not-lf*>* ; 'Handset model name.'
<*manufacturer-specific-info*> ::= <*default-char*>* ; 'One or more lines of manufacturer-specific fields.'

4.2.6 Procedures

The dynamic menu item protocol has one state, which is called IDLE. This section provides information on how handsets will handle protocol actions.

4.2.6.1 Notation

∧	Logical AND of predicates
⊂	Subset
⊄	Not Subset
<>	Not Equal
==	Equal
(a,b)	Pair consisting of members ‘a’ and ‘b’
a->b	Lookup of ‘b’ with ‘a’ as a lookup key
a.b	Member of ‘a’ identified by ‘b’
∪	Set Union
∩	Set Intersection

4.2.6.2 Handset FSM State Transition Table: Item primitives

State	Event	Action	Next State	
IDLE (entry state)	*recv IA:token:group-name:item-name:item-type:provider-information:item-help:action-list from smsc:ms:port* ∧ (*smsc, ms*) *Ë AuthorizedServers(group-name)*	Unauthorized (server not accepted)	IDLE	1
	recv IA:token:group-name:item-name:item-type: provider-information:item-help:action-list from smsc:ms:port ∧ (*smsc, ms*) ⊂ *AuthorizedServers(group-name)* ∧ *(group-name, item-name)* ⊂ *StoredItems* ∧ *token == StoredItems.*(*group-name, item-name*)*-> token*	*StoredItems := StoredItems* ∩ *(group-name, item-name)* *remove old* *StoredItems := StoredItems* ∪ (*group-name, item-name*)*->token:group-name:item-name:item-type:item-help:action-list*	IDLE	1
	recv IA:token:group-name:item-name:item-type: provider-information:item-help:action-list from smsc:ms:port ∧ (*smsc, ms*) ⊂ *AuthorizedServers(group-name)* ∧ (*group-name, item-name*) *Ì StoredItems* ∧ *token <> StoredItems.*(*group-name, item-name*)*-> token*	Unauthorized (token not accepted)	IDLE	1
	recv IA:token:group-name:item-name:item-type: provider-information:item-help:action-list from smsc:ms:port ∧ (*smsc, ms*) ⊂ *AuthorizedServers(group-name)* ∧ (*group-name, item-name*) ⊄ *StoredItems*	*StoredItems := StoredItems* ∪ (*group-name, item-name*)*->token:group-name:item-name:item-type:item-help:action-list*	IDLE	1

	recv IR:token:group-name:item-name from smsc:ms:port ∧ (*smsc, ms*) ⊄ *AuthorizedServers(group-name)*	Unauthorized (server not accepted)	IDLE	1
	recv IR:token:group-name:item-name from smsc:ms:port ∧ (*smsc, ms*) ⊂ *AuthorizedServers(group-name)*∧ (*group-name, item-name*) *Ì StoredItems* ∧ *token == StoredItems.*(*group-name, item-name*)-> *token*	*StoredItems := StoredItems* ∩ *(group-name, item-name)*	IDLE	1
	recv IR:token:group-name:item-name from smsc:ms:port ∧ (*smsc, ms*) ⊂ *AuthorizedServers(group-name)* ∧ (*group-name, item-name*) *Ì StoredItems* ∧ *token <> StoredItems.*(*group-name, item-name*)-> *token*	Unauthorized (token not accepted)	IDLE	1
	recv IR:token:group-name:item-name from smsc:ms:port ∧ (*smsc, ms*) ⊂ *AuthorizedServers(group-name)* ∧ (*group-name, item-name*) ⊄ *StoredItems*	Error (Item does not exist)	IDLE	1
	recv IL:token:group-name from smsc:ms:port ∧ (*smsc, ms*) ⊄ *AuthorizedServers(group-name)*	Unauthorized (server not accepted)	IDLE	1
	recv IL:token:group-name from smsc:ms:port ∧ (*smsc, ms*) ⊂ *AuthorizedServers(group-name)* ∧ (*group-name, ANY) Ì StoredItems* ∧ *token == StoredItems.*(*group-name, ANY*)-> *token*	*Generate reply to smsc:ms:port with list of items with matching token.*	IDLE	1
	recv IL:token:group-name from smsc:ms:port ∧ (*smsc, ms*) ⊂ *AuthorizedServers(group-name)* ∧ (*group-name, ANY) Ì StoredItems* ∧ *token <> StoredItems.*(*group-name, ANY*)-> *token*	Unauthorized (token not accepted)	IDLE	1
	recv IL:token:group-name from smsc:ms:port ∧ (*smsc, ms*) ⊂ *AuthorizedServers(group-name)* ∧ (*group-name, ANY) Ë StoredItems*	*Generate reply to smsc:ms:port with empty item list.*	IDLE	1

4.2.6.3 Handset FSM State Transition Table: Authorization primitives

State	Event	Action	Next State	
IDLE (entry state)	*recv AA:group-name:server-list from smsc:ms:port* ∧ (*smsc, ms*) ⊂ *AuthorizedServers(group-name)*	*AuthorizedServers := AuthorizedServers* ∪ (*group-name*)->*server-list*	IDLE	1
	recv AA:group-name:server-list from smsc:ms:port ∧ (*smsc, ms*) ⊄ *AuthorizedServers(group-name)*	Unauthorized (server not accepted)	IDLE	1
	recv AR:group-name:server-list from smsc:ms:port ∧ (*smsc, ms*) ⊂ *AuthorizedServers(group-name)*	*AuthorizedServers* := *AuthorizedServers* ∩ (*group-name*)->*server-list*	IDLE	1
	recv AR:group-name:server-list from smsc:ms:port ∧ (*smsc, ms*) ⊄ *AuthorizedServers(group-name)*	Unauthorized (server not accepted)	IDLE	1
	recv AL:group-name from smsc:ms:port ∧ (*smsc, ms*) ⊂ *AuthorizedServers(group-name)*	*Generate reply to smsc:ms:port with list of AuthorizedServers(group-name)*	IDLE	1
	recv AL:group-name from smsc:ms:port ∧ (*smsc, ms*) ⊄ *AuthorizedServers(group-name)*	Unauthorized (server not accepted)	IDLE	1

4.2.6.4 Handset FSM State Transition Table: Device Capability primitives

State	Event	Action	Next State	
IDLE (entry state)	*recv DC:identifier from smsc:ms:port* ∧ (*smsc, ms*) ⊂ *AuthorizedServers(DC)* ∧ *identifier* == "LAN"	*Generate reply to smsc:ms:port with current language selected in handset*	IDLE	1
	recv DC:identifier from smsc:ms:port ∧ (*smsc, ms*) ⊂ *AuthorizedServers(DC)* ∧ *identifier* == "INFO"	*Generate reply to smsc:ms:port with handset information (manufacturer, model, firmware version, date)*	IDLE	1
	recv DC:identifier from smsc:ms:port ∧ (*smsc, ms*) ⊄ *AuthorizedServers(DC)*	Unauthorized (server not accepted)	IDLE	1

4.2.7 State Definitions

IDLE. The device can receive frames from remote peer systems.

4.2.8 Parameter Definitions

StoredItems. A set of dynamic menu items. This set contains the current dynamic menu structure of the handset. Based on this information the handset builds the menu structure.

AuthorizedServers. A set of authorized servers. This set contains menu names associated with authorized servers. If a menu name exists in the set of authorized servers, then commands concerning that menu will only be accepted from the list of servers associated with that menu name. If a menu name does not exist in the authorized servers set, then any server may send commands concerning that menu.

4.2.9 Event Descriptions

recv IA:token:group-name:item-name:item-type:provider-information:item-help:action-list from smsc:ms:port. Item add request has been received through given smsc (smsc) from ms-isdn address (ms), and from given NBS port (port). The request contains the token, group-name, item-name, item-type, provider-information, item-help, and action-list fields.

recv IR:token:group-name:item-name from smsc:ms:port. Item remove request has been received through given smsc (smsc) from ms-isdn address (ms), and from given NBS port (port). The request contains the token, group-name, and item-name fields.

recv IL:token:group-name from smsc:ms:port. Item list request has been received through given smsc (smsc) from ms-isdn address (ms), and from given NBS port (port). The request contains the token, and the group-name fields.

(smsc, ms) Ë AuthorizedServers(group-name). The pair (smsc, ms) is not associated with the group-name in the AuthorizedServers set, i.e., the group is in the AuthorizedServers set (authorization is required), but the message source address does not match any of the authorized servers. It should be noted that the group for which authorization is not required, should never be put into the AuthorizedServers set. The underlying assumption is that if the group does not exist in the AuthorizedServers list, then no authorization is required.

(smsc, ms) Ì AuthorizedServers(group-name). The pair (smsc, ms) is associated with the group-name in the AuthorizedServers set, or the group-name does not exist in the AuthorizedServers set, i.e., either the server identified with the pair is authorized, or no authorization is required.

(group-name, item-name) Ì StoredItems. A menu item which is uniquely identified by the pair (group-name, item-name), w, exists in the handset.

(group-name, item-name) Ë StoredItems. A menu item which is uniquely identified by the pair (group-name, item-name), does not exist in the handset.

token == StoredItems.(group-name, item-name)-> token. The authorization token received in the request is the same as the one stored for the menu item identified by the pair (group-name, item-name), i.e., authorization based on token matching is successful.

token <> StoredItems.(group-name, item-name)-> token. The authorization token received in the request is not the same as the one stored for the menu item identified by the pair (group-name, item-name), i.e., authorization based on token matching is not successful.

recv AA:group-name:server-list from smsc:ms:port. The authorization add request has been received through the given smsc (smsc) from the ms-isdn address (ms), and from the given NBS port (port). The request contains the group-name and server-list fields.

recv AR:group-name:server-list from smsc:ms:port. The authorization remove request has been received through the given smsc (smsc) from the ms-isdn address (ms), and from the given NBS port (port). The request contains the group-name and server-list fields.

recv AL:group-name from smsc:ms:port. The authorization list request has been received through the given smsc (smsc) from the ms-isdn address (ms), and from the given NBS port (port). The request contains the group-name field.

recv DC:identifier from smsc:ms:port. The device capability request has been received through given smsc (smsc) from ms-isdn address (ms), and from given NBS port (port). The request contains the identifier field.

(smsc, ms) Ì AuthorizedServers(DC). The pair (smsc, ms) is associated with the special group-name for device capabilities in the AuthorizedServers set, or the group-name does not exist in the AuthorizedServers set, i.e., either the server identified with the pair is authorized, or no authorization is required. It is handset implementation-specific as to which servers are authorized to access the device capabilities. The recommendation is to allow servers enabled to access operator and manufacturer menus to access device capabilities.

(smsc, ms) Ë AuthorizedServers(DC). The pair (smsc, ms) is not associated with the special group-name for device capabilities in the AuthorizedServers set, i.e., the server identified with the pair is unauthorized to use device capability request.

identifier == "LAN". The server requests information concerning the current language in effect in the handset.

identifier == "INFO". The server requests information concerning the handset in general, i.e., this information contains manufacturer name, model name, and may contain manufacturer-specific information like firmware version, and firmware update date.

4.2.10 Action Descriptions

Unauthorized (server not accepted). The server is not authorized to use the requested command, because the server from which the command is sent is not found in the authorization list for the menu. It is implementation-specific in that a generic error response is either sent back to the server or not. Because the handset user incurs the cost of sending the error response, this should be limited to special cases.

Unauthorized (token not accepted). The server is not authorized to use the requested command, because the token used does not match the initial token used to add the item. It is implementation-specific in that a generic error response is either sent back to the server or not. Because the handset user incurs the cost of sending the error response, this should be limited to special cases.

StoredItems := StoredItems Ç (group-name, item-name). Remove the menu item uniquely identified by the pair (group-name, item-name) from the set of StoredItems.

StoredItems := StoredItems È (group-name, item-name)->token:group-name:item-name:item-type:item-help:action-list. Add the menu item uniquely identified by the pair (group-name, item-name) to the set of StoredItems. After this addition, the handset will be able to show the menu item in its menu structure.

Error (Item does not exist). Access to an item that does not exist in the StoredItems. No action.

Generate reply to smsc:ms:port with list of items with matching token. Generate response to the Item list command with a list of menu items in the given menu which have a matching token, i.e., scan through the menu items that belong to the given menu, and have a matching token, and generate a reply based on these criteria.

Generate reply to smsc:ms:port with empty item list. No matching items, i.e., generate an empty response to the item list command.

AuthorizedServers := AuthorizedServers È (group-name)->server-list. Add to the set of AuthorizedServers an association of group-name, server-list., i.e., make all servers listed in the server-list authorized to access the menu-group.

AuthorizedServers := AuthorizedServers Ç (group-name)->server-list. Remove from the set of AuthorizedServers all associations between group-name and any server in the server-list, i.e., remove authorization to access the group from all the listed servers.

Generate reply to smsc:ms:port with list of AuthorizedServers(group-name). Generate a response to the authorization list request with all the servers that are associated with the given group.

Generate reply to smsc:ms:port with current language selected in handset. Generate a response to the device capability request concerning language.

Generate reply to smsc:ms:port with handset information (manufacturer, model, firmware version, date). Generate a response to the device capability request concerning generic device information.

4.2.11 Informative Examples

4.2.11.1 Protocol Action Examples

Message example that adds three entries to the home operator menu. The first item enables user to select the "Call Customer Service" menu item, and then a call will automatically be made to the operator's customer help desk. The second item will cause a supplementary service string to be sent to the network. This string will cause all calls to be forwarded to the user's voice mailbox. The third item is a linked item which prompts the handset to send a SMS message that causes the operator's server to send back volatile menu items associated with ("OPER Radiolinja", "Billing:Info") pair.

```
BODY:
IA:
OPER Radiolinja
Call:Customer:Service
N
Automatically call Radiolinja customer service.
C VOICE
+35850123456w123
--
IA:
OPER Radiolinja
All calls:to voice:mail box
N

C SS
**21*555123456789#
--
IA:
OPER Radiolinja
Billing:info
L

M SMS
+35850123/+358501234567/5508
38:BODY:
RM:OPER Radiolinja
Billing:Info
```


4.2.11.2 Phone UI Examples

The first phone UI example (Fig 4.2-2.) shows two menu items in a operator-specific menu. The first item initiates a call to the operator's customer help desk. The second item indicates the state of a service provided in the network. This is implemented by an item linked with a selected (volatile) value. As this volatile value is found in the handset memory, it is shown to the user.

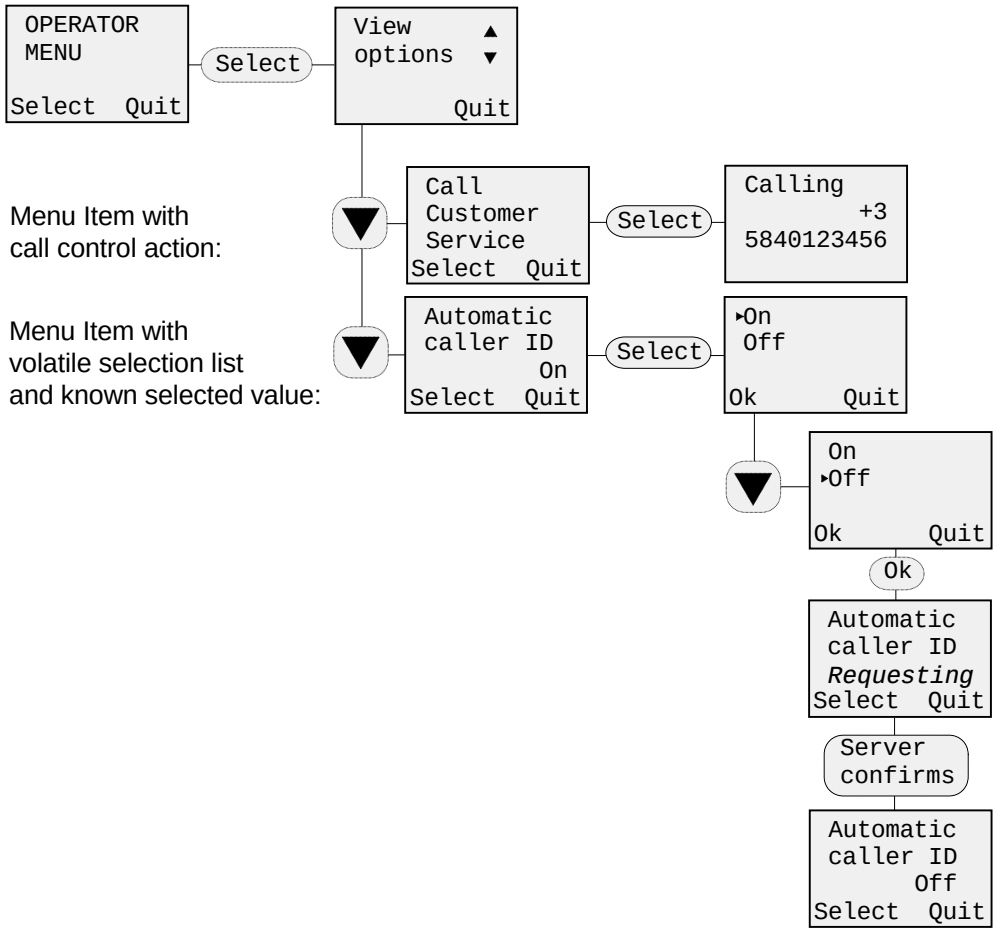

Figure 4.2-2. Phone UI example 1.

The second phone UI example (Fig 4.2-3.) shows a situation in which the state of a service provided in the network is not known. In order to provide to the user information on the selected (active) state of the service, as well as the possible states available, the phone requests linked menu items from network.

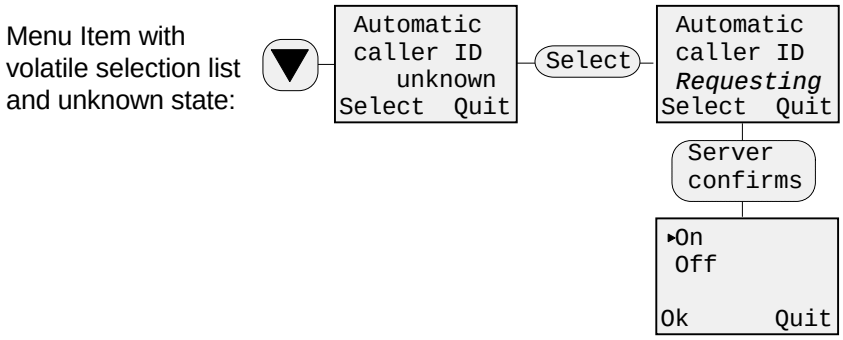

Figure 4.2-3. Phone UI example 2.

4.3 TAGGED TEXT MARKUP LANGUAGE (TTML)

TTML (Tagged Text Markup Language) makes it possible to use a HTML-like markup language in an environment with mixed terminals. It supports clients with limited capabilities, i.e. legacy mobile phones, as well as advanced Value Added Services-enabled phones with browser functionality. The protocol and page layout conventions (conversion rules) support the requirements of this mixed terminal base to guarantee easy development as well as solid implementation.

The TTML language provides means to

- enter text into fields
- make selections from different kinds of menus
- jump to links (using anchors)

At the end of the section there are a number of examples. These examples show the user interface in a legacy phone, i.e. the user will see the pages “as is”. The use of intuitive tags and layout in legacy phones requires only minor learning even for an untrained user. For even more user friendly applications a Browser (i.e. an adaptation of the phone UI) may be provided. The browser automates the editing otherwise done by the user.

The TTML Gateway or Server listens to the NBS port 5580 decimal (15CC hexadecimal).

4.3.1 TTML Protocol Description

The TTML protocol has been optimized for use on SMS and USSD bearers. When implemented in GSM or in systems derived from GSM, the protocol has the following features specific to GSM Short Message Service:

- Concatenation of message fragments is supported.
- Only request forms “one SMS long” (160 characters, including possible headers) are supported in the uplink of TTML 1.0 , i.e. a filled form can reside on the first fragment
- The protocol implements a reverse logic. Selections are made by removing a single character (“*” or “#”). Selection list might have preselections done already in the downlink message.
- As opposite to other markup languages (for example HTML) only a single form can reside on a page. However, the form might contain several form-parts, intended to be interpreted by a single script in the server.

4.3.1.1 Formal TTML Protocol Specification

The protocol is implemented according to the following definitions:

Downlink

; ‘downlink is messages from server to mobile device’

<downlink-TTML-page> ::= [*<header-line>*] *<page-content>**

;'header line is mandatory if the page content contains a form

<*page-identifier*> ::= <*default-char-not-ld-or-space*>+

; 'page-identifier is case-insensitive. Must not be sequence starting with "*#*#*#".'

<*header-line*> ::= <*header-esc*> [<*page-identifier*> | <*default-page-identifier*>] <*header-item*>* <*line-delimiter*>
;'the header-line enables the client and/or the server to exchange operational parameters. It can be used in uplink as well as in downlink.

<*header-item*> ::= <*wait-indicator*> |<*client-protocol-version*> | <*client-buffer-size*> | <*ttml-SAP-information*> |
<*future-header-extension*>

<*wait-indicator*> ::= <*comma*>"**w**"<*default-char-not-ld-or-space*>*

;' the characters after the "**w**" are for future protocol extensions

<*client-protocol-version*> ::= <*comma*> " **v**" <*protocol-version-number*>

;' when no version number is present, version 1 is defined.

<*client-buffer-size*> ::= <*comma*> "**s**" <*byte-size-of-buffer*>

;' the buffer size defines the maximum page size the client can handle

<*ttml-SAP-information*> ::= <*comma*> "**a**" <*common-destination-ms-isdn*> "***l***" <*common-destination-smsc-ms-isdn*>

;' The Service Access Point defines the location (SMSC, destination number) of the available service

<*protocol-version-number*> ::= <*default-char-not-ld-or-space*>+

<*byte-size-of-buffer*> ::= <*default-char-not-ld-or-space*>+

<*future-header-extension*> ::= <*comma*> <*default-char-not-ld-or-space*>*

<*default-page-identifier*> ::=

"*#*#*#" [<*common-language*>] <*line-delimiter*> ; 'Language'

;'the default-page-identifier is a language independent keyword used to request the "**homepage**" of the server/gateway'

<*page-content*> ::= <*default-char*>* | <*form-part*> <*line-delimiter*>

<*default-char-not-ld-or-space*> ::= 'Any character in the default character set (or in character set explicitly indicated) except <*line-delimiter*> or <*space*>. In GSM, the GSM character set from 3.38/3.40.'

<*form-part*> ::=

<*anchor-selection*> |
<*alpha-input-prompt*> |
<*numeric-input-prompt*> |
<*one-of-many-input-selection*> I
<*many-of-many-input-selection*> |
<*extension-escape-sequence*>

<*anchor-selection*> ::= <*anchor-selection-prompt*> <*anchor-selection-option*>+

<*anchor-selection-prompt*> ::= <*jump-list-esc*> <*prompt-text*>

<*anchor-selection-option*> ::=

<*loc-jump-esc*> <*char5*> <*prompt-text*> |
<*glob-jump-esc*> <*char5*> <*prompt-text*>

; 'the anchor-selection form part presents a list of hyperlinks'

<*alpha-input-prompt*> ::= <*alpha-entry-esc*>[<*input-field-length*>]<*prompt-text*> <*prompt-sentinel*>[<space>*<*entry-suggestion*>]<space>*

<*numeric-input-prompt*> ::= <*digit-entry-esc*>[<*input-field-length*>] <*prompt-text*> <*prompt-sentinel*>[<space>*<*entry-suggestion*>]<space>*

; 'the input-prompt form-parts allow for entry of text or numbers'

<*input-field-length*> ::= <*digit*>$^{+}$; 'one or more digits defining the maximum size of the input field'

<*entry-suggestion*> ::= <*default-char-not-ld*>$^{+}$

<*prompt-text*> ::= <*default-char-not-ld*>$^{+}$; 'input prompt text, not reserved char sequence. Cannot start with a digit'

<*prompt-sentinel*> ::= "**:**" | "**?**"

<*one-of-many-input-selection*> ::= <*one-of-many-input-prompt*> <*one-of-many-input-option*>$^{+}$

<*one-of-many-input-prompt*> ::= <*one-list-esc*> <*prompt-text*>

<*one-of-many-input-option*> ::= <*select-esc*> [<*char5*>] <*prompt-text*>

;'form-part similar to radio buttons'

<*many-of-many-input-selection*> ::= <*many-of-many-input-prompt*> <*many-of-many-input-option*>$^{+}$

<*many-of-many-input-prompt*> ::= <*many-list-esc*> <*prompt-text*>

<*many-of-many-input-option*> ::= <*select-esc*>[<*char6*>] <*prompt-text*>

;'form-part similar to checkboxes'

<*extension-escape-sequence*> ::= <*extend-esc*> <*extend-string*> <*line-delimiter*>

;'form-part for advanced and future features, provides protocol extension capability'

<*extend-string*> ::= <*hidden-field*> | <*hidden-variable-field*> | <*request-prompt*> | <*default-char-not-ld*>*

<*hidden-field*> ::= "**H**" <*default-char-not-ld*>$^{+}$ <*line-delimiter*>

<*hidden-variable-field*> ::= "**V**" [<*hidden-presentation*>*":"*] <*default-char-not-ld*>$^{+}$ <*line-delimiter*>

<*hidden-presentation*> ::= <*default-char-not-ld*>$^{+}$; 'name of hidden field'

<*request-prompt*> ::= "**R**"<*default-char-not-ld*>$^{+}$; 'The request-prompt (a user interface button) is used for visual purposes as well as to define the prompt text before sending the request. The request-prompt is optional'.

<*header-esc*> ::= [<*line-delimiter*>] <*char1*> <*char1*> ; 'line-delimiter optional if first char on page, or immediately after NBS header.'

<*alpha-entry-esc*> ::= <*line-delimiter*> <*space*>* <*char3*>

<*digit-entry-esc*> ::= <*line-delimiter*> <*space*>* <*char3*> <*char3*>

<*select-esc*> ::= <*line-delimiter*> <*space*>* <*char1*> <*char2*>

<*loc-jump-esc*> ::= <*line-delimiter*> <*space*>* <*char2*> <*char2*>

<*glob-jump-esc*> ::= <*line-delimiter*> <*space*>* <*char3*> <*char2*>

<*extend-esc*> ::= <*line-delimiter*> <*space*>* <*char2*> <*char3*>

<*one-list-esc*> ::= <*line-delimiter*> <*space*>* <*char2*> <*char1*>

<*many-list-esc*> ::= <*line-delimiter*> <*space*>* <*char3*> <*char1*>

<*jump-list-esc*> ::= <*line-delimiter*> <*space*>* <*char1*> <*char3*>

<*char1*> ::= "**.**"; 'Hex value 2Eh'

<*char2*> ::= “**>**”; ‘Hex value 3Eh’

<*char3*> ::= “**<**”; ‘Hex value 3Ch’

<*char4*> ::= “**=**”; ‘Hex value 3Dh. Manipulation char, special meaning only when escaped by <*line-delimiter*> <*space*>* { <*char1*> | <*char2*> | <*char3*> }.’

<*char5*> ::= “*****”; ‘Hex value 2Ah. Manipulation char, special meaning only when escaped by <*line-delimiter*> <*space*>* { <*char1*> | <*char2*> | <*char3*> }.’

<*char6*> ::= “**#**”; ‘Hex value 23h. Manipulation char, special meaning only when escaped by <*line-delimiter*> <*space*>* { <*char1*> | <*char2*> | <*char3*> }.’

Uplink

<*uplink-TTML-page*> ::=

[<*header-line*> | <*page-identifier*> | <*default-page-identifier*>] <*page-request-content*>*

<*page-request-content*> ::= <*default-char*>* | <*form-request-part*> <*line-delimiter*>

<*form-request-part*> ::=

<*anchor-selection-request*> |

<*alpha-input-reply*> |

<*numeric-input-reply*> |

<*one-of-many-input-selection-request*> I

<*many-of-many-input-selection-request*> |

<*extension-escape-sequence*>

<*anchor-selection-request*> ::= [<*anchor-selection-prompt*>] <*anchor-selection-option-request*>+

<*anchor-selection-prompt*> ::= <*jump-list-esc*> <*prompt-text*>

<*anchor-selection-option-request*> ::=

<*loc-jump-esc*> <*char5*> <*prompt-text*> |

<*loc-jump-esc*> <*prompt-text*> |

<*glob-jump-esc*> <*char5*> <*prompt-text*> |

<*glob-jump-esc*> <*prompt-text*>

<*alpha-input-reply*> ::=

<*alpha-entry-esc*>[<*input-field-length*>] <*prompt-text*> <*prompt-sentinel*><*space*>*

<*default-char-not-ld*>* <*space*>*

<*numeric-input-reply*> ::=

<*digit-entry-esc*>[<*input-field-length*>] <*prompt-text*> <*prompt-sentinel*><*space*>*

<*default-char-not-ld*>*<*space*>*

<*one-of-many-input-selection-request*> ::= <*one-of-many-input-prompt*> <*one-of-many-input-option-list*>+

<*one-of-many-input-option-list*> ::=

<*one-of-many-input-option*> | ; ‘non-selected input options’

<*one-of-many-input-option-request*> ; ‘one selected input option’

<one-of-many-input-option-request> ::= *<select-esc> <prompt-text>*

<many-of-many-input-selection-request> ::= *<many-of-many-input-prompt> <many-of-many-input-option-list>*$^{+}$

<many-of-many-input-option-list> ::=

<many-of-many-input-option> |	; 'non-selected input option'
<many-of-many-input-option-request>	; 'one or more selected options'

<many-of-many-input-option-request> ::= *<select-esc> <prompt-text>*

Strict interpretation of the above protocol description indicates that one line-delimiter immediately following another is needed in some situations. However, in these cases the second line-delimiter can be dropped.

The usage of edited forms causes the response message (to the gateway) to be long. The user (or the browser) is allowed to remove unnecessary (for the interpretation of the request form) text from the original message. Observe that an unidentified extension tag shall not be removed from the uplink message, since it might be of value to the Gateway/Server.

4.3.1.2 TTML Protocol Rules

An element, i.e. an item in the text, can be either a plain text element (ending with a <line-delimiter>, or end of message) or part of a form or a link. The form/link element must start with an escape sequence followed by a name/value, ending with a line-delimiter. The escape sequence might be of variable length since it can contain a number of space characters for indention purposes. The maximum length of

- a plain text element is 254 characters
- a form/link element is 20 characters
- an entry to a text field is 155 characters

These restrictions in element length are defined in order to facilitate browser implementations in terminals with severe memory constraints.

A number of the definitions and variables defined above are explained:

- header-line. The header line contains a list of configuration parameters separated by commas. Parameters not known to the browser (or server) shall be ignored. The header-line is mandatory in the downlink, if a page-identifier is needed.
- hidden-variable. The format, and interpretation, of the hidden variable is the same as for the text entry field (alpha-input-reply). However, the client shall not display the content to the user. The hidden variable is intended to be used as, for example, a state variable.
- hidden-field. The text (up to a line-delimiter) after an hidden-field indicator is not to be displayed to the user.
- ttml-SAP-information (Service Access Point). The Service Access Point is usually the same as the originator of the downlink page (i.e. the address of the server). By explicitly defining another SAP links between servers can be created.
- wait-indicator. The TTML-Server can include a wait-indicator on the downlink page to indicate to the client that the service is interactive. The user interface of the client is thus able to inform the user that it is waiting for a new page from the server.

- page-identifier. The page-identifier is similar to the URL of a HTML page. It is a unique identifier in the context of a TTML server (or gateway). It must be present whenever a TTML form is used. The page-identifier is often part of the header-line, but can also reside separately in the uplink.
- byte-size-of-buffer. If the buffer size is indicated by the client it must be able to handle concatenated messages as well as forms of the given size. The default size of the client buffer is 415 bytes. If the request for a page has been done without a NBS port number then the server shall assume that the client cannot handle forms on any but the first message fragment (160 characters). The plain text part of the downlink message can still be longer.
- client-protocol-version. The default protocol version is “1”, and indicates that the client might not support more than 415 byte forms.
- Word. A word as part of a form must be unique in it’s context. A word representing a selection in a selection-list must be unique in this list, but it can be reused in another selection-list, even on the same page. A word representing a hyperlink can occur several times on a page, but will reference the same target page.
- Local hyperlink. The word expressing a local hyperlink has a unique interpretation on the TTML page (page-identifier) where it is used, but it can be reused on other pages. The “anchor-selection-prompt” is for descriptive purpose only, and is not valid in the uplink message.
- Global hyperlink. A global hyperlink must be unique in the TTML-server, i.e. the same hyperlink text can not be reused to refer to other pages. The “anchor-selection-prompt” is for descriptive purpose only, and is not valid in the uplink message.
- extension-escape-sequence. The extensions enables for future enhancements of the TTML protocol. If the page contains an element not identified by the browser it shall be hidden from the user, but sent back to the server with the request.
- Form. A form is regarded to be equal to the form presentation in HTML (i.e. the items between <FORM> and </FORM>). Thus the form can include items such as selections lists, multi-selections lists and text/digit entry.
- Page. A page is regarded as a TTML page as seen by the client. The page might consist of multiple fragments in the communications media (i.e. SMS).

4.3.1.3 Short Message Concatenation

Concatenation support in the browser is mandatory. The browser knows about long messages only using concatenation headers. In version 1 of the TTML protocol only the downlink supports multi-fragment messages.

- Plain text pages can be concatenated
- Forms can span over page boundaries. However, a few design recommendations should be taken into account:
    - When designing pages for legacy phones it is important to restrict the forms to a single message fragment. Pages with operations (selections, forms) can then only have them on the first fragment of a concatenated message.
    - In order to optimise transfer and response times request pages must fit into one SMS fragment. Selections on pages longer than one short message fragment shall be supported by the phone, as long as the request form fits the first fragment (also when text has been entered)
    - If the browser is used on a concatenated message consisting of at least two message fragments, a response can contain only as much as fits into one SMS (truncated) back to the server.

4.3.1.4 Examples

TTML is characterized by equal support for browser phones and legacy phones. The examples below picture the use of pages in legacy phones, with only basic SMS viewing and editing capabilities. The user interface of phones with browser capabilities might of course be very different.

The page might be a plain menu, with a single selection list. In the uplink direction the page is used by deleting a selection identifier "*". In this and in the following examples, <char1> is replaced with '+' character for readability.

```
++MENU
>+OperatorChoices
+>*Flight-any
+>*Flight-one
+>*Restaurants
+>*News
+>*Bus-service
+>*Stock-Exchange
+>*Currency
+>*Movie
+>*GSM-prices
```

Pages often combine both forms and explanatory text.

```
++INVENTORY
   <Code:
   <<Amount:
   <<Date:
Use the codes
of the inventory
manual! Date in
format ddmmyy.
```

4.3.1.4.1 Hyperlink Usage

A jump list (hyperlinks) is used to access new information. The links are direct references to pages available via the TTML server.

The TTML protocol specified two kinds of hyperlinks, local and global. The local links use keywords that are unique in the context of the specific page, while the global hyperlink is unique in the context of the server.

Next is an example of a typical global hyperlink set-up. A keyword like "News" is prone to be reused as a local hyperlink on many pages, while a global hyperlink can be used only once. Therefore the Reuters news service is called "News-Reuters" rather than only "News".

```
++MENU
Select one of our global
+<services:
<>*Flight-Finnair
<>*Restaurants-Michelin
<>*News-Reuters
<>*Stock-Exchange
<>*Currency-Merita
<>*Movie-by-Rcity
<>*Bus-service
Have a good day!
```

The server will then interpret the above as a global keyword, or as a local (page-specific) keyword.

A link selection, for example “Bus-service” (<>Bus-service) could cause the whole (edited) page to be resent to the server. It is possible to reduce the content of the request in such a way that it contains only one word, i.e. the global keyword.

```
Bus-service
```

The TTML server will respond by sending the page referenced by the specific page-identifier.

Example of a typical local hyperlink service

```
++HA-MENU
Select one of HELSINKI AIRPORT
+<services:
>>*Flight-depart
>>*Flight-arrival
>>*Restaurants
>>*Bus-service
>>*News
or select the service page of
+<Radiolinja
<>*GSM-050
Have a good day!
```

A linked selection, for example “Bus-service” (>>Bus-service), could cause the whole (edited) page to be resent to the server. An alternative is to reduce the content, and send only the page-identifier (mandatory) of the page and the selected link.

```
HA-MENU
>>Bus-service
```

or

```
++HA-MENU
>>Bus-service
```

The hyperlink is not part of a form, but an independent selection.

4.3.1.4.2 Reducing Content

A user interface which automates the editing of the message, i.e. a browser, can optimize the usage of the bearer channel by reducing content for the uplink. A downlink page including free-text can be reduced in size before it is sent to the Server. All free-text can be removed. Only the page-identifier, text entry fields and selections are valid uplink information. The page-identifier is a mandatory field if the uplink page contains a form or a local hyperlinks. If the uplink page contains a selection from a single- or multi-selection list then the “one-of-many-input-prompt” or the “ many-of-many-input prompt” are also mandatory.

The downlink page:

```
++PIZZAORDER
Order the best pizzas in town from
Manolo. Lots of cheese and toppings.
   >+pizza
   +>*Florida
   +>*Margarita
   +>*QuattroStagione
   <address:
1/2 hour for delivery!.
```

might be edited and reduced for uplink as:

```
++PIZZAORDER
>+pizza
   +>Florida
   <address: Village Circle 2 F 34
```

Using this method air time usage can be reduced, thereby reducing the cost involved.

4.3.1.4.3 Header Combinations

Headers can be combined in a number of ways. The following examples explain a number of these.

Downlink

//SCKL15CC CR++ page-identifier CR text

//SCKL15CC//SAP123-456 CR++ page-identifier CR text

//SCKL15CC14AA230201//SAP123-456 ++ page-identifier CR text

++page-identifier text

text

CR++ page-identifier text

Uplink

//SCKL15CC CR++ page-identifier,v2,s450 CR text

page-identifier text

page-identifier

The NBS port address can be presented in text format, as in the examples, but it can also be in binary format. In that case the <space> after the NBS header is equal to the end of the binary header.

```
//SCKL15CC15CC090201
++NEWS
free-text
 +<newsjumps
   >>*SportNews
   >>*FinancialNews
   >>*NationalNews
free-text
```

```
//SCKL15CC15CC090202
   >>*RegionalNews
   >>*WeatherNews
free-text,
…
```

4.3.1.4.4 Service Access Point Presentation

The message can be defined to have a Service Access Point different from the sender. When the user replies to the message/form the request is sent to the address defined as SAP. This makes it possible to create a distributed server architecture, as well as broadcast support.

The downlink page:

```
//SCK15CC
++NEW_LINKS,a9919-35850123456
Here are a number of links from a
new content provider.
+<services:
>>*Movies
>>*Theater
>>*Eating
Enjoy the information! Sponsored by
NOKIA.
```

< This page intentionally left blank >

5. APPENDIX A: RESERVED PORT NUMBERS

5.1 INTRODUCTION

The NBS specification enables port addresses in range of [0..65535] decimal. This range is divided into reserved port address range and dynamic and/or private port address range.

Please note: In the earlier version of the specification, the port number space was not the same as IANA port number space. However, WAP Forum's WDP protocol [WAP_WDP], when run over GSM SMS, is essentially identical to NBS. If UDP is available, that can be used instead of WDP, in which case the UDP datagrams have the same port numbers as the WDP datagrams. Therefore, WDP port number space is the same as IANA's TCP/UDP port number space. See section 2.1 for more information.

Usage of the reserved port address range of [0..49151] decimal is restricted, and assignment of a port in this range requires a port address assignment authority. The assuagement authority is either Nokia or IANA, depending on the protocol used (NBS or WDP). For more information on the registration procedure, contact Nokia Third Party Support. The reserved port address range is further divided into ports for "Well-known protocols" (ranges [0..1023] decimal), and registered ports (range [1024..49151] decimal).

Port addresses in the dynamic/private address range of [49152..65535] decimal may be used freely by any vendor. No registration is required and thus the ports can be used dynamically. These address ranges are not moderated, so conflicts in port addressing may occur.

5.1.1 NBS Protocol

The NBS protocol is described in the Narrow Band Socket Specification [NBS_SPEC], available at Nokia and Intel World Wide Web sites. This section is only intended to describe some of the key features of the NBS protocol.

NBS is intended to use the momentum of the Internet world to create a new set of applications in the extreme narrow band world. The NBS protocol is a datagram protocol, equal to UDP in the Internet world. The paradigm has been adapted to the very narrow channel of SMS and USSD. It provides device addressing, i.e. phone numbers, and port addressing, i.e. application level addressing. The NBS protocol enables devices to have multiple active applications, each identified by a port number.

The protocol allows for text headers as well as binary headers (using the concept of User-Data-Headers). Either one can be used depending on the infrastructure of the network and the devices used, as well as the application.

The NBS protocol is designed to be modular and expandable. New headers can be added as need arise, they are concatenated to the end of the existing headers.

Text headers are positioned in 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 User Data Header as defined in [GSM_03.40]. The information elements 00h for concatenation and 04h and 05h for port addressing are used.

5.1.2 WDP

The WDP (Wireless Datagram Protocol), defined by WAP Forum in [WAP_WDP], is essentially identical to NBS protocol when run over GSM Short Message Service. WDP specifications are available from WAP Forum World Wide Web site.

5.2 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. For the latest list, please contact Nokia Third Party Developer Support.

Table 5-1. 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*	*Generic Business Card exchange (MIME vCard) Card reader*
228	*E4*	*Calendar Items (MIME vCalendar) Calendar reader*
5501	*157D*	*Compact Business Card reader*
5502	*157E*	*Service Card reader*
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*
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)*
5601	*15E1*	*<RESERVED>*
5603	*15E3*	*<RESERVED>*
8500	*2134*	*<RESERVED>*
8501	*2135*	*<RESERVED>*
8502	*2136*	*<RESERVED>*

Table 5-2. 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*

Smart Messaging Specification