3GPP TSG-T (Terminals) Meeting #11
Palm Springs, USA, 14 - 16 March, 2001

Tdoc TP-010044

3G TS 31.113 V1.0.0 (2001-03)

Technical Specification

3rd Generation Partnership Project; Technical Specification Group Terminals; USAT Interpreter Byte Codes

Keywords

USIM, UICC, Interpreter

3GPP

Postal address

3GPP support office address

650 Route des Lucioles - Sophia Antipolis

Valbonne - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Internet

http://www.3gpp.org

Contents

Foreword

This Technical Specification (TS) has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the document.

1 Scope

The present document specifies the byte codes that are recognised by an USAT Interpreter. The byte codes primary purpose is to provide efficient programmatic access to the SIM Application Toolkit commands.

The design objectives of the byte code set are:

- Compact representation for efficient transmission over the air interface.
- Minimisation of USAT Interpreter complexity to minimise SIM footprint and ease compliance testing.
- Easily configured and extended.
- Source language independent although XML-style mark-up languages are explicitly envisioned.
- Transport bearer independent (e.g. SMS, GPRS...)
- Transport protocol independent.
- Independent from design of external entities.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.

For a non-specific reference, the latest version applies.

[1] SCP TS 102 220: Numbering system for telecommunication IC card applications

[2] 3GPP TS 31.111: "3rd Generation Partnership Project (3GPP); USIM Application Toolkit (USAT)"

[3] 3GPP TS 23.038: "3rd Generation Partnership Project (3GPP); Alphabets and language-specific information"

[4] GSM 03.48: "Digital cellular telecommunications system (Phase 2+); Security Mechanisms for the SIM Application Toolkit; Stage 2"

[5] ISO/IEC 7816-6 (1995): "Identification cards - Integrated circuit(s) cards with contacts, Part 6: Inter-industry data elements"

[6] ISO/IEC 9797-1 (1999): "Information technology – Security techniques – Message Authentication Codes (MACs) – Part 1: Mechanisms using a block cipher"

[7] ISO 8731-1:1987 "Banking -- Approved algorithms for message authentication -- Part 1: DEA".

[8] ISO/IEC 10116:1997 "Information technology -- Security techniques -- Modes of operation for an n-bit block cipher".

[9] Schneier, Bruce: "Applied Cryptography Second Edition: Protocols, Algorithms and Source code in C", John Wiley & Sons, 1996, ISBN 0-471-12845-7.

[10] SCP TS 102 221: Smart cards; UICC-Terminal interface; Physical and logical characteristics

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

anchor: A named location on a page to which references can be made and at which rendering by the USAT Interpreter initiated. Anchors can be referenced by anchor reference TLVs.

attribute: A property assigned to a TLV. The attribute can consist of a single bit or of a sequence of consecutive bits within the attribute bytes of a TLV.

attribute byte(s): A sequence of consecutive bytes in the value part of a TLV containing the attributes of that TLV.

current page: The page which is currently rendered by the USAT Interpreter.

default page: A locally stored page that can be branched to in exceptional conditions to keep the proactive session (as defined in 3GPP TS 31.111 [2]) alive and to allow further navigation. The default page can be modified by administrative purposes. The default page has a reserved page identifier.

external system entity: Any entity outside the USAT interpreter, able to communicate with the USAT interpreter (e.g. USAT Gateway, content/application system).

navigation unit: A block of a service description that can be referenced (by its anchor) and hence independently activated.

page: The context of a USAT Interpreter rendering, the scope of USAT Interpreter variables and the unit of transmission between an external system and an USIM containing the USAT Interpreter.

protected variable: A shared variable, which is protected by an one time password.

service: A collection of pages that define a unitary capability of the mobile equipment from the point of view of the user. Examples include remote database access, electronic mail, and alerts.

service ID: Unique ID to identify a service on the content side.

shared variable: A variable to be shared with the following page. Shared variables can be provided to the next page in a protected or non protected manner.

string pool: A list of predefined variables with provided by the current page within the page TLV. The string pool is mainly used for optimisation purposes.

variable ID: Identifier to reference a variable within a variable usage area.

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply

AID	Application IDentifier
DCS	Data Coding Scheme
ID	IDentifier
MAC	Message Authentication Code
MSB	Most Significant Bit
NU	Navigation Unit
OTP	One Time Password
PIX	Proprietary application Identifier eXtension
SW1/SW2	Status Word 1 / Status Word 2
TLV	Tag Length Value

UCS2	Universal two byte coded Character Set
UE	User Equipment
UICC	Universal Integrated Circuit Card
URL	Universal Resource Locator
USAT	USIM Application Toolkit
USIM	Universal Subscriber Identity Module
XML	eXtensible Markup Language

3.3 Symbols

'0' to '9' and 'A' to 'F' The sixteen hexadecimal digits.

Single bits are identified by b1 to b8, where b1 is the LSB and b8 is the MSB of the byte containing the bit.

RFU bits and bytes are to be set to '0'.

4 Model of computation

A *service* is mobile phone functionality as seen by the user, for example e-mail, information access or order entry.

A service is composed of one or more *pages*. Pages describe information presented to the subscriber and retrieve input from the subscriber. The unit of transmission to the mobile equipment as well as the unit of USAT Interpreter interpretation is the page. The set of all pages describing a service is called the *service description*.

Pages are composed of navigation units. Anchors reference the beginning of navigation units. Therefore anchors are points in a service description that can be branched to from other points in the service description. Each page has an implicit anchor at the beginning of the page.

In some mark-up languages pages are known as decks and anchors are known as cards.

The USAT Interpreter renders pages and provides a way to navigate from within pages to anchors belonging to the same page or other pages. The requirements of the USAT Interpreter include a way to automatically go back to previously visited anchors.

The USAT Interpreter manages a stack of N last anchors visited. Each anchor visited is added to this history list, except if an appropriate flag is set in the anchor. The back operation is interpreted relative to this history list and means go to the preceding anchor in the list.

When reaching the last byte code of a page, the USAT Interpreter shall behave like ending a Navigation Unit.

4.1 Navigation

A page expressed as compiled byte code instructions is stored as a unit in the USAT Interpreter. The page is the smallest unit that the external system entity can provide to the USAT Interpreter. A page is partitioned into one or more navigation units each of which can be referenced using anchors. In other words, navigation units and anchors are included in pages.

The anchor is defined as being the elementary navigation target. The USAT Interpreter can skip from one anchor to another, backwards and forwards based either on control flow constructs or user interaction. If a navigation unit contains no instructions to branch to an anchor within the current page or another page, the USAT Interpreter shall branch to a locally installed *default page*. This keeps the proactive session alive and allows further navigation.

Pages are stored in the USAT Interpreter. The structure is described later in this document. These pages are stored either permanently in the USIM or received and interpreted on the fly.

Pages and navigation units are referenced using anchor references as described below.

To be able to create multiple-page services, page references are used to fetch new pages or to link pages together.

The behaviour of the USAT Interpreter in response on user interaction (e.g. backward move, exit, help button pressed) is defined by the navigation event handler within the page or navigation unit context.

If no navigation event handler is defined in the page context or in the navigation unit context, the behaviour of the USAT Interpreter is as follows:

When the USAT Interpreter receives "Proactive SIM session terminated by the user" or "No response from user" from the UE, the default page shall be executed. If there is no default page or the current page is the default page, the proactive session shall be terminated by the USAT Interpreter.

When the USAT Interpreter receives "Backward move in the proactive SIM session requested by the user" from the UE, the USAT Interpreter shall go to the preceding anchor in the history list. If there are no more preceding entries, the default page shall be executed. If there is no default page or the current page is the default page, the proactive session shall be terminated by the USAT Interpreter.

All other received responses from the UE shall be ignored by the USAT Interpreter.

4.2 Activation

The USAT Interpreter can be activated in four ways:

- locally from the UE using menu selection,
- locally from the UE as the result of an event,
- by an incoming page as a result of a previous page request from the USAT Interpreter or
- by an incoming page initiated by an external system entity ("push").

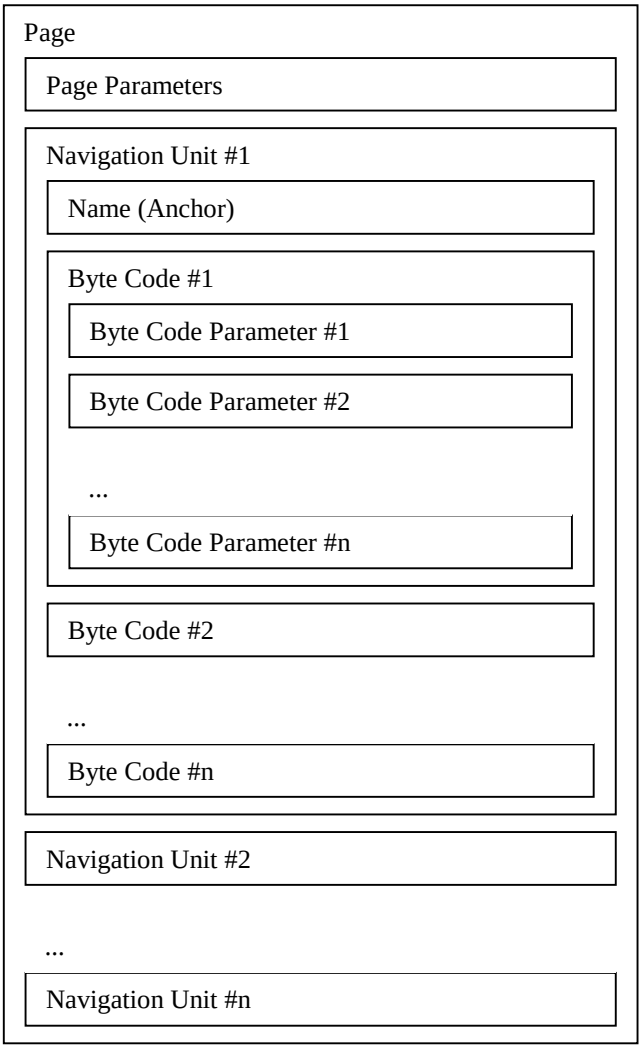
The rendering of a page shall be independent of the means of activation.

With respect to activation locally from the UE using menu selection, the SETUP MENU command as described in 3GPP TS 31.111 [2] can contain one or more references to local or remotely stored page identifiers. When one of these identifiers is selected, the USAT Interpreter is activated and renders the referenced page, local or remote. Registering of pages to the main menu is up to administrative means.

An event (as described in 3GPP TS 31.111 [2] or proprietary events defined by the card issuer) is linked to a local or remotely stored page identifier. When the UE sends an ENVELOPE command containing an event, the USAT Interpreter is activated and renders the referenced page, local or remote. If an event is received not referencing to a page, the event shall be ignored by the USAT Interpreter. For security reasons, setting up events is up to administrative means.

4.3 Page format overview

The following picture gives an overview of the construction and elements of a page to be rendered by the USAT Interpreter:



A transmission initiated by the USAT Interpreter to the external system entity is performed when the USAT Interpreter executes a Byte Code containing a Page Reference TLV containing a Page Parameters TLV (see chapter 7.9.2 Page Parameters) referring to a page which is not locally stored on the USIM.

Page Reference TLVs are used in the following byte code commands:

- Assign and Branch
- Branch on Variable Value

5 TLV Format

The Tag Length Value (TLV) is the basic data structure element. If the value part of a TLV contains other TLV elements it is called a BER-TLV or a template TLV. If not, it is called a simple TLV. See ISO/IEC 7816-6 [5] for more information on data objects.

The tag byte contains a seven-bit tag value and an attribute byte-present bit in the MSB. If the attribute byte-present bit is set then the leading byte(s) in the value field contain attribute information for the element identified by the tag.

Length	Value	Description	M/O
1	T	Tag	M
1-3	L	Length of following data, a length value of '00' is allowed	M
L	V	The data value associated with the tag	O

The length is BER coded onto 1, 2 or 3 bytes according to ISO/IEC 7816-6 [5].

The value of a TLV is the content of its value field and therefore *evaluation* of a TLV yields its value.

TLVs shall appear in the order given in the present document. Additional TLVs may be appended to the TLVs given in the present document. Simple TLVs not understood by the USAT Interpreter shall be ignored by the USAT Interpreter. Template TLVs not known by the USAT Interpreter shall generate an error message to the user.

5.1 Coding of the tag byte

The tag byte of all TLVs described in the present document is as follows:

b8	b7	b6	b5	b4	b3	b2	b1
Attribute byte present bit	Tag value coded on 7 bits						

Attribute byte present bit	Value
Attribute byte present as first byte of V	1
Attribute byte not present as first byte of V	0

5.2 Attributes in TLVs

Every TLV can have one or more attributes bytes if indicated by the attribute byte present bit of the tag byte. The coding of an attribute byte is shown below. Attributes provided in the attribute byte shall be related to the belonging TLV. The meaning of the attributes of a TLV is TLV specific and specified in the TLV descriptions.

An attribute given in an attribute byte can consist of a single bit or a combination of consecutive bits forming an attribute value.

The default value of an attribute value or an attribute bit within an attribute byte is always '0'. The '0' value of an attribute shall be used by the USAT Interpreter, if the attribute is not available in the TLV.

Whenever the attributes for a tag require more than 7 bits within an attribute byte, the number of attribute bytes will be extended. The extension of the attribute byte shall be indicated by the MSB of the attribute byte, which is called the follow bit.

The order of attribute bytes in the bit stream is from left to the right (attribute byte 1 to 2…) from the MSB to the LSB bits in the structure.

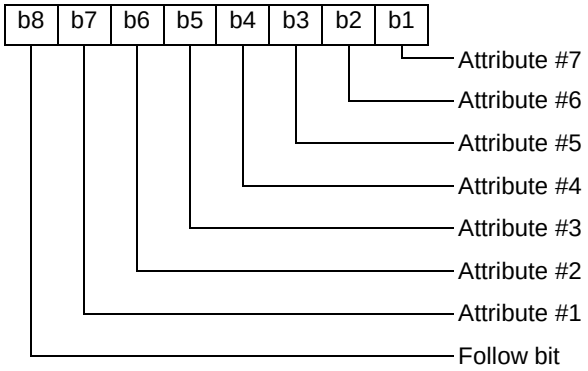
Attributes or attribute bytes not expected or not known by the USAT Interpreter shall be ignored by the USAT Interpreter.

5.3 Coding of attribute bytes

The MSB of each attribute byte indicates if another attribute byte follows or not. The MSB is called Follow Bit. The remaining seven bits of an attribute byte contain TLV specific attributes, either coded as a single bit or as a combination of consecutive bits.

The context, namely the tag, completely determines the order, span and semantics of the bit-packed attribute values. An attribute consisting of more than 1 bit may span two attribute bytes but this situation should be avoided if possible.

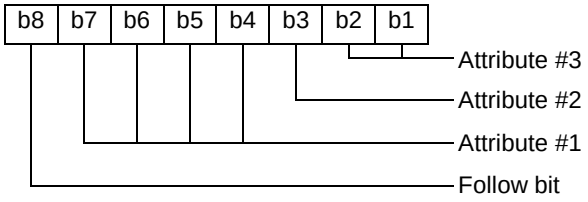
General coding:



Follow bit coding:

Follow bit	**Value**
Another attribute byte available as next byte of V	1
No more attribute bytes available	0

Other coding example where attribute #1 consists of a single bit, attribute #2 consists of a 4 bit value and attribute #3 consists of a 2 bit value.



6 Variables

Variables are name-value pairs. The name is called the variable identifier (ID) and the value is called the variable value. Operations are provided to refer to a variable value by using its variable ID and for setting and resetting the value associated with a variable.

Variables can be stored in the following usage areas:

- Environment variable area
- Permanent variable area
- Temporary variable area
- Page string element

Variables have one of the following variable types:

- SMS default 7-bit coded alphabet as defined in 3GPP TS 23.038 [3] with bit 8 set to 0
- SMS default 7-bit coded alphabet as defined in 3GPP TS 23.038 [3] packed
- Binary
- UCS2 coded string

The list can be extended.

6.1 Usage areas

Variables are referred by using a unified one byte notation. The one byte variable reference is called the variable ID. b8 and b7 of the variable ID are used to indicate the belonging of a variable to a certain usage area. The remaining 6 bits are used to reference a certain variable within the usage area.

Due to the used coding, the number of variables per area is restricted to 64.

The coding of the variable ID is as follows:

b8	b7	b6	b5	b4	b3	b2	b1	
0	0							belongs to Environment usage area
0	1							belongs to Permanent usage area
1	0							belongs to Temporary usage area
1	1							belongs to Page String Element usage area
		x	x	x	x	x	x	identifier of the variable within the usage area

The size of the different usage areas is to be defined by the card issuer and configured during the personalisation process of the USIM.

6.1.1 Environment variable usage area

This usage area consists of 3 different partitions:

- USAT interpreter system information partition
- USIM issuer information partition
- End user information partition

6.1.1.1 USAT Interpreter system information partition

The USAT Interpreter partition is preloaded during the manufacturing process of the USIM or during the runtime of the USAT Interpreter.

At least the following information shall be stored:

Variable ID	Description	Coding
'00'	ICCID of USIM	Binary coding as for EF_{ICCID} specified in SCP TS 102 221 [10]
'01'	USAT Interpreter version	Binary coded in 3 bytes: '01 xx xx": '01' for this version of the byte code specification 'xx xx' card manufacturer specific
'02'	USAT Interpreter profile (features)	tbd. according to administrative configuration coding
'03'	USIM issuer	SMS default 7-bit coded alphabet as defined in 3GPP TS 23.038 [3] with bit 8 set to 0
'04'	Terminal Profile as got at runtime	Binary coded as defined in 3GPP TS 31.111 [2] for TERMINAL PROFILE
'05'	Error Code as generated by the last byte code command executed	Binary coded as specified in chapter "11 Error coding "
'06'	Maximum page size for temporary storage of one page	Binary coded, number of bytes available for page storage, coded in 2 bytes indicating a block of memory consisting of 128 bytes, MSB first
'07'... '13'	RFU	

6.1.1.1.1 Write access

This partition shall not be updated by administrative means after the personalisation process. The variables in this partition may be changed by the USAT Interpreter itself, if e.g. the configuration of the USAT Interpreter changes (e.g. addition of a new plug-in).

6.1.1.1.2 Read access

The information stored in this partition can be freely accessed by any page executed by the USAT Interpreter.

6.1.1.2 USIM issuer information partition

The information stored in this partition is under the control of the USIM issuer. The USIM issuer is responsible to allocate variable IDs for his own purposes in the range from '14' to '28'. The used variable IDs shall be published to content providers.

6.1.1.2.1 Write access

This partition can be updated by the USIM issuer by administrative means.

6.1.1.2.2 Read access

The information stored in this partition can be freely accessed by any page executed by the USAT Interpreter.

6.1.1.3 End user information partition

The information stored in this partition is under the control of the end user. If the user decides to store information in this partition, the following variable IDs shall be used:

Variable ID	Description	Coding
'29'	User name	SMS default 7-bit coded alphabet as defined in 3GPP TS 23.038 [3] with bit 8 set to 0 or UCS2 coded
'2A'	User e-mail address	SMS default 7-bit coded alphabet as defined in 3GPP TS 23.038 [3] with bit 8 set to 0
'2B' ... '3F'	RFU	

6.1.1.3.1 Write access

This area can be updated locally by the end user by an user interface provided by the USAT Interpreter. Most likely, the user interface will be realised by a locally stored application of the USAT Interpreter accessible by the end user.

6.1.1.3.2 Read access

The information stored in this partition can be freely accessed by any page executed by the USAT interpreter.

6.1.2 Permanent variable area

This area is used to store permanently variables which can be accessed even after the USIM was reset. This area is organised as a cyclic variable buffer. If the buffer is full, a new entry shall delete the most oldest entries until enough space is made available to store the new entry.

Each entry consists of the service ID of the page storing the variable in this area, the variable ID and the content of the variable. For pages using this variable area, it is mandatory to provide the service ID in the page TLV. The assignment of service IDs is up to an external system entity.

6.1.2.1 Write access

Any page which provides a service ID may store permanent variables.

6.1.2.2 Read access

The information in this area can be freely accessed by pages providing a service ID within the page TLV which is contained in the list of permanently stored variables. A page shall have access to those variables only, which have the same service ID as stored in the page TLV.

6.1.3 Temporary variable area

Temporary variables are used during the execution of the current page. They may be shared with the following page. Temporary variables are used for 2 purposes:

- as variables defined and used within the current page,
- as variables to be shared between the current page and the following page.

The current page shall define, which variables are to be kept for access of the following page. To ensure, that only a dedicated following page can access the variables defined to be sharable, the current page may protect them with a One Time Password (OTP), which has to be presented by the following page in order to get access to the shared variables. The OTP to be presented by the following page may be ciphered.

If this mechanism is used to protect shared variable, it might happen that a page is not able to access the protected shared variables, if the sequence of pages provided to the USAT Interpreter is disturbed (e.g. by using backward navigation between pages...).

6.1.3.1 Write access

Only the current page can allocate temporary variables. The current page can allocate temporary variables as many as it is space available in this area.

To indicate how to provide variables to the next page, the KeepAll flag in the attribute of the current page and the OTP TLV and the KeepAlive TLV within the current page TLV is used according to the following table:

KeepAll flag	OTP TLV	KeepAlive TLV	Actions
set	present	present	not valid, if occurs, the KeepAll attribute shall be ignored, variables listed in KeepAlive shall be kept for the following page and shall be protected by OTP
set	present	not present	all temporary variables shall be kept for the following page and shall be protected by OTP
set	not present	present	not valid, if occurs, the variables listed in KeepAlive shall be kept for the following page and shall not be protected by OTP
set	not present	not present	all temporary variables shall be kept for the following page and shall not be protected by OTP
not set	present	present	variables listed in KeepAlive shall be kept for the following page and shall be protected by OTP
not set	present	not present	not valid, no variables to be kept for the following page
not set	not present	present	variables listed in KeepAlive shall be kept for the following page and shall not be protected by OTP
not set	not present	not present	no variables to be kept for the following page

6.1.3.2 Read access

A current page can freely access temporary variables stored by this current page. Protected variables of a previous page shall only be accessible after a successful verification of the One Time Password set by the previous page to protect temporary variables to be shared between these two pages.

In order to unlock the shared protected variables the Page Unlock TLV has to be present within the page TLV. The Page Unlock TLV shall contain the OTP (ciphered or in clear as indicated by the KIc of the TLV) of the previous page. If the OTP in the Page Unlock TLV matches the OTP stored with the protected variables, the protected variables are made available to the current page as regular temporary variables.

6.1.2.3 Lifetime of temporary variables

By default, all variables which are not kept explicitly to be shared by the following page are deleted, after the page is processed.

If there are protected variables, but the current page does not contain a matching OTP the protected variables are deleted before processing the current page.

6.1.4 Page string element

This area is provided optionally by the current page. It can be used to store e.g. strings that are used several times in the current page.

The first string element in the String Pool TLV shall be identified by the variable reference 'C8', the next with 'C9' and so on.

6.1.4.1 Write access

The information contained in this area is read only.

6.1.4.2 Read access

The information can be accessed by the current page.

6.2 Variable values

The value associated with a variable identifier is a length-byte string pair. The type of a variable value is determined by the usage context. The USAT Interpreter shall keep track of the type of a variable. How the type of the variable is stored internally within the USAT Interpreter is up to the implementation of the USAT Interpreter.

The length of the variable value is restricted to 65535 ('FFFF') bytes. Each variable has one of the following types:

Type of variable	coding (3 bits)
SMS default 7-bit coded alphabet as defined in 3GPP TS 23.038 [3] with bit 8 set to 0	'000'
SMS default 7-bit coded alphabet as defined in 3GPP TS 23.038 [3] packed	'001'
Binary format	'010'
UCS2 coded string	'011'
Other values	RFU

The coding specified, shall be used to indicate the type of variable, when variable substitution is used.

6.3 Variable substitution

Variable IDs may appear in fields explicitly labelled as containing a variable identification. Variable substitution can take place in the following TLVs:

- Simple TLV Indicator
- Inline Value TLV
- Inline Value 2 TLV
- Page Parameters TLV

In the case of running text, the escape values 'C0'...'C7' are used to signal that the next byte is a variable ID.

Variable IDs shall never use the values 'C0' to 'C7' as these values are used as escape values.

The least significant 3 bits of the escape value shall be used to indicate the type of the variable coded according to the table above.

Coding of escape values:

Coding of escape value	Type of variable referenced to
'C0'	SMS default 7-bit coded alphabet as defined in 3GPP TS 23.038 [3] with bit 8 set to 0
'C1'	SMS default 7-bit coded alphabet as defined in 3GPP TS 23.038 [3] packed
'C2'	Binary format
'C3'	UCS2 coded string
'C4' ... 'C7'	RFU

If the text contains an escape value as a regular character, that character has to be doubled in the text. If for instance an UCS2 string contains a character code in the range 'C0' to 'C7' without it being a variable reference, the value is byte-stuffed by doubling the value, i.e. {'C0'} becomes {'C0', 'C0'}, {'C1'}becomes {'C1', 'C1'}.

Whenever a variable reference is encountered one of the following mechanisms are used to replace the variable reference depending on the context:

Method 1:

- the escape character is removed from the running text
- the following variable reference is simply replaced by
  - the current content of the variable (that means inserting the variable content into the running text)

Method 2 (used for USAT Interpreter Response Parameters):

- the escape character is not removed from the running text, the type information of the escape character shall be set according to the type of the variable
- the following variable reference is replaced by

- the length of the content of the variable. The length is coded onto 1, 2 or 3 bytes according to ISO/IEC 7816-6 [5].
- the current content of the variable (inserting the variable content into the text)

A variable value shall not contain a variable substitution, i.e. an inserted variable value is not rescanned for variable IDs.

7 Used TLVs

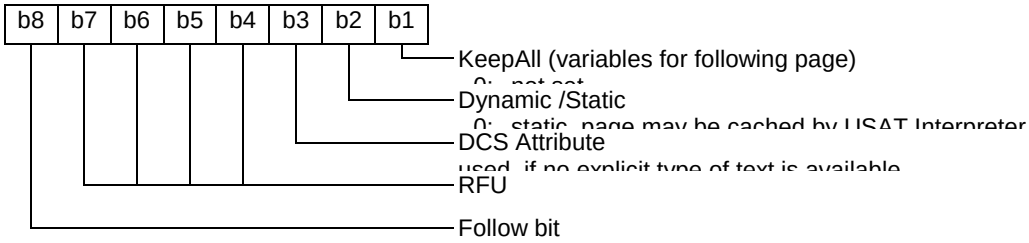
7.1 Page

A page is the unit of USAT Interpreter rendering and the name scope of the temporary variables defined in its anchors.

Length	Value	Description	M/O
1	'01' / '81'	Page Tag	M
1-3	A+B+C+D+ E+F+G+H+I	Length	M
A	Data	Attributes	O
B	TLV	Page Identification	M
C	TLV	Page Unlock Code	O
D	TLV	One Time Password	O
E	TLV	Keep Alive List	O
F	TLV	Service ID	O
G	TLV	String Pool	O
H	TLVs	Navigation Event Handler - one or more TLVs	O
I	TLVs	Navigation Units - one or more TLVs	M

The following chapters specify the attributes and simple TLVs used in the page template TLV.

7.1.1 Attributes



7.1.2 Page Identification

The content of this TLV is a sequence of bytes to uniquely identify the page. This identification shall not contain a #-character (coded '23'). This reference can later on be used by the interpreter to reference the page (e.g. for caching mechanisms or accessing the page by the end-user from the menu structure).

Coding:

Length	Value	Description	M/O
1	'02'	Page Identification Tag	M
1-3	L	Length	M
L	Data	Unique identification of the page	M

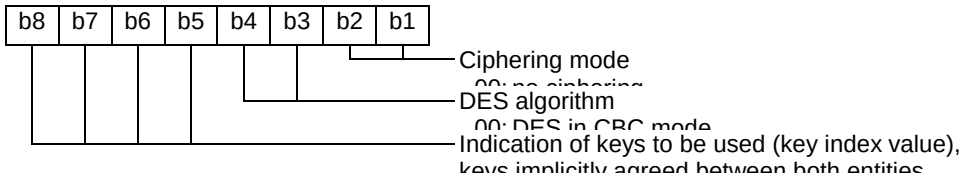
7.1.3 Page Unlock Code

The content of this TLV is a key index and algorithm identifier KIc for ciphering and a sequence of bytes encrypted according to the information provided by the KIc. This sequence of bytes shall be decrypted by the USAT Interpreter and verified against an OTP provided by a previous page.

Coding:

Length	Value	Description	M/O
1	'03'	Page Unlock Code Tag	M
1	L+1	Length (up to 1+8 bytes)	M
1	KIc	Key index and algorithm identifier for ciphering	M
L	Data	Page unlock code (one time password of the previous page)	M

The KIc is coded as follows:



DES is the algorithm specified as DEA in ISO 8731-1 [7]. DES in CBC mode is described in ISO/IEC 10116 [8]. Triple DES in outer-CBC mode is described in section 15.2 of [9]. DES in ECB mode is described in ISO/IEC 10116 [8].

The initial chaining value for CBC modes shall be zero.

7.1.4 One Time Password

The content of this TLV is a sequence of bytes generated by random to protect the temporary variables of the current page against unauthorised access.

Coding:

Length	Value	Description	M/O
1	'04'	One Time Password Tag	M
1	L	Length (up to 8 bytes)	M
L	Data	One time password (random value generated by an external system entity)	M

7.1.5 Keep Alive List

The content of this TLV is a list of variable IDs to indicate which variables of the current page may be shared with the following page. The list shall not contain other variable IDs than variable IDs referring to temporary variables.

Coding:

Length	Value	Description	M/O
1	'05'	Keep Alive List Tag	M
1	L	Length (number of temporary variable IDs, up to 64 variables)	M
L	Data	Variable IDs	M

7.1.6 Service ID

The content of this TLV is a sequence of bytes to indicate that the current page shall belong to a certain service. The assignment and coding of service IDs is up to an external system entity. The length of a service ID shall not exceed 8 bytes.

Coding:

Length	Value	Description	M/O
1	'06'	Service ID Tag	M
1	L	Length (number of bytes of the service ID, <= 8 bytes)	M
L	Data	Service ID, unique identification of a service	M

7.1.7 String Pool

The content of this TLV is a list of strings coded in with the alphabet indicated in the DCS attribute used within the page. Within the page the strings are referenced by using their variable references within the page string element area.

Coding:

Length	Value	Description	M/O
1	'07'	String Pool Tag	M
1 - 3	L	Length	M
L	Data	LV values of each string element in the string pool	M

7.1.8 Navigation Event Handler

This TLV specifies the behaviour of the USAT Interpreter for the case of a user interaction (e.g. behaviour on backward move, on exit, etc.). This page level template TLV may be overridden by Navigation Event Handlers in the navigation units contained in this page. The content describes the action which shall be performed after the USAT Interpreter has received a matching General result byte of the Terminal Response within a proactive session. The USAT Interpreter shall either branch to another anchor or execute a native command.

Coding:

Length	Value	Description	M/O
1	'08'	Navigation event handler tag	M
1 – 3	L+1	Length	M
1	Data	General Result byte of Terminal Response (see 3GPP TS 31.111 [2])	M
L	TLV	either - Anchor Reference TLV or - Page Reference TLV or - Execute Native Command TLV	M

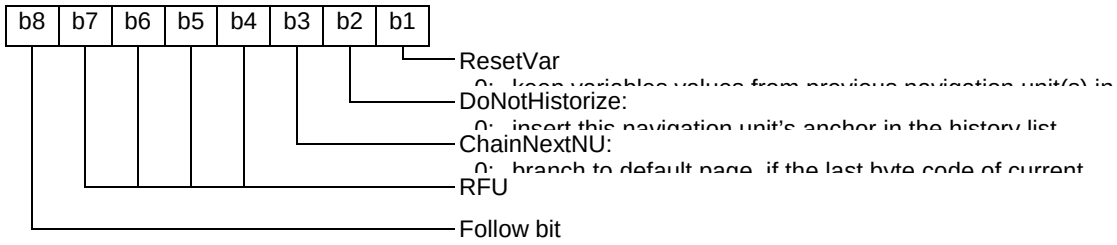
7.2 Navigation Unit

A navigation unit is a component of a page. It is named using an anchor. A navigation unit is referenced using an anchor reference.

Length	Value	Description	M/O
1	‘09’ / '89'	Navigation Unit Tag	M
1-3	A+B+C+D	Length	M
A	Data	Attributes	O
B	TLV	Anchor (name of a navigation unit)	O
C	TLVs	Navigation Event Handler - one or more TLVs	O
D	TLVs	USAT Interpreter Byte Codes - one or more TLVs	O

The following chapters specify the attributes and simple TLVs used in the navigation unit template TLV.

7.2.1 Attributes



7.2.2 Anchor

The content of this TLV is a sequence of bytes identifying the current navigation unit. It is mandatory to provide this TLV, if a navigation unit of the current page or another page wants to branch to this navigation unit.

Coding:

Length	Value	Description	M/O
1	'0A'	Anchor Tag	M
1 - 3	L	Length	M
L	Data	Unique identification of navigation unit within the page. A sequence of bytes to uniquely identify the Anchor. This identification may not contain a #-character (coded '23') and is coded by the external system entity.	M

7.2.3 Navigation Event Handler

These TLVs shall specify the behaviour of the USAT Interpreter for the case of a user interaction (e.g. behaviour on backward move, on exit, etc.). This TLV overrides the Navigation Event Handler TLV of the current page, if present. The content describes the action which shall be performed after the USAT Interpreter has received a matching General result byte of the Terminal Response within a proactive session. The USAT Interpreter shall either branch to another anchor or execute a native command.

Coding:

See chapter "7.1.8 Navigation Event Handler".

7.2.4 USAT Interpreter Byte Codes

These TLVs contain the executable part of the page.

Coding:

See chapter "8 USAT Interpreter byte codes".

7.3 Anchor Reference

This TLV is used to refer to a navigation unit in the current page or in another page.

Length	Value	Description	M/O
1	'0B'	Anchor Reference Tag	M
1-3	L	Length	M
L	Data	Anchor Reference Name	M

An anchor reference name is a page identification (see "7.1.2 Page Identification") followed by a '23' ("#") and the anchor (name of a navigation unit, see "7.2.2 Anchor") within the page. Either the page identification or the anchor (but

not both) can be omitted. If the page identification is omitted the reference is to an anchor on the current page. If the anchor name is omitted the reference is to the beginning of the referenced page.

7.4 Variable Identifier List

This TLV is used to list a sequence of variables.

Length	Value	Description	M/O
1	'0C'	Variable Identifier List Tag	M
1	L	Length	M
L	Data	Variable IDs (up to 64 Variable IDs)	M

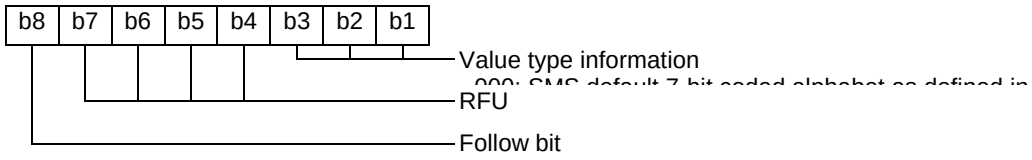
7.5 Inline Value

This TLV inserts a byte array, which often is simply running text, at the point of its appearance. The TLV is thus simply a way to encapsulate an immediate value.

The Inline value content may contain escape characters to indicate variable references. Therefore the Inline Value content has to be byte- stuffed. The possibly available variable references have to be expanded according to chapter "6.3 Variable substitution" Method 1 during processing of this TLV by the USAT Interpreter.

Length	Value	Description	M/O
1	'0D' / '8D'	Inline Value Tag	M
1-3	A+B	Length	M
A	Data	Attributes	O
B	Data	Inline value content	O

Coding of the attributes:



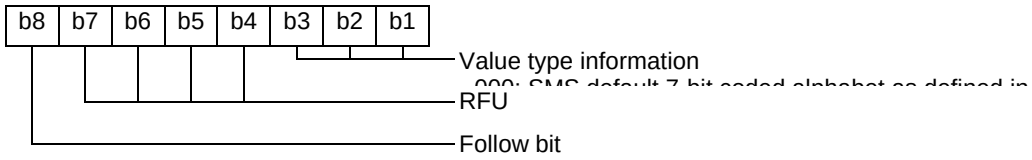
7.6 Inline Value 2

This TLV inserts a byte array, which often is simply running text, at the point of its appearance. The template is thus simply a way to encapsulate an immediate value. Usage and syntax and behaviour of this TLV is similar to the Inline Value TLV, but another tag value is used.

The Inline Value 2 content may contain escape characters to indicate variable references. Therefore the Inline Value 2 content has to be byte-stuffed. The possibly available variable references have to be expanded according to chapter "6.3 Variable substitution" Method 1 during processing of this TLV by the USAT Interpreter.

Length	Value	Description	M/O
1	'0E' / '8E'	Inline Value 2 Tag	M
1-3	A+B	Length	M
A	Data	Attributes	O
B	Data	Inline Value 2 content	O

Coding of the attributes:



7.7 Input List

This TLV contains a list of Variable ID List TLVs and Inline Value TLVs.

Length	Value	Description	M/O
1	'0F'	Input List Tag	M
1-3	L	Length	M
L	TLVs	Any sequence of - Variable Identifier List TLVs and / or - Inline Value TLVs	M

7.8 Ordered TLV List

This TLV is used to associate a list of other TLVs. The order and the possible types of contained TLVs within an ordered TLV list is specified. The number of actual contained TLVs is implicitly given by the length indication of the Ordered TLV List. It is allowed, that the ordered TLV list does not contain any TLV.

Depending on the context each optional TLV within the Ordered List of TLVs shall have a different tag value.

Length	Value	Description	M/O
1	'10'	Ordered TLV List Tag	M
1-3	A+...+Z	Length	M
A	TLV	First TLV	O/M
...	...	...	
Z	TLV	Last TLV	O/M

7.9 Page Reference

This TLV can represent a page, an anchor within the current page, or an anchor within another page.

If the Anchor Reference TLV or the Variable ID List TLV is available, then the USAT Interpreter shall start rendering the requested locally stored Anchor. If the Anchor is not found locally, an error is generated.

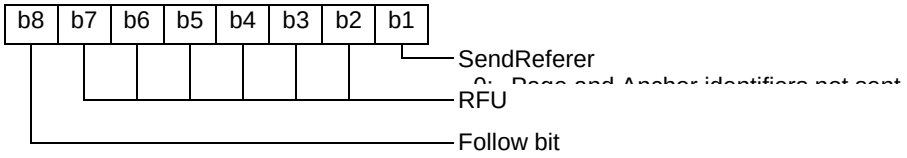
If the Page Parameters TLV is available (that indicates that the page is not locally stored on the USIM, i.e. e.g. stored at an external system entity), then the USAT Interpreter shall substitute all found variable references in the Page Parameters TLV according to chapter "6.3Variable substitution" method 2, and transmit the resulting data formatted as a Page Parameters TLV to the external system entity. If the SendReferer attribute is set, the Page Identification TLV of the current page is appended to the generated Page Parameters TLV prior to the transmission.

If the transmission to the external system entity fails, an USAT Interpreter transmission error shall be generated by the USAT Interpreter and the execution shall stop.

If the transmission to the external system entity was successful, the USAT Interpreter shall continue to render the current page.

Length	Value	Description	M/O
1	'11' / '91'	Page Reference Tag	M
1-3	A+B+C	Length	M
A	Data	Attributes	O
B	TLV	either - Anchor Reference TLV or - Variable ID List TLV (referring to a variable containing the Anchor Reference, only the first variable ID shall be considered by the USAT Interpreter, remaining variable IDs shall be ignored) or - Page Parameters TLV	M

7.9.1 Attributes



7.9.2 Page Parameters

This TLV contains the Page Parameters. The Page Parameters are a sequence of bytes possibly containing variable references to be substituted according to chapter "6.3 Variable substitution" method 2. The sequence of bytes is to be byte-stuffed to ensure, that variable references can be detected. The content of the Page Parameters are coded by the external system entity and possibly contains a request for the next page to be transmitted to the USAT Interpreter by an external system entity.

Length	Value	Description	M/O
1	'12'	Page Parameters Tag	M
1	L	Length	M
L	Data	Parameters (text possibly containing variable references)	M

7.10 Secure Message

This is for further study.

8 USAT Interpreter byte codes

Each USAT Interpreter byte code is a TLV. Each byte code has its own byte code tag value, optional attributes and a list of arguments. Arguments, if present, shall appear in the order given.

The byte codes make use of the USAT Interpreter TLVs as follows:

	Attribute Bytes	Variable Reference	Variable List	Inline Value	Inline Value 2	Page Reference	Ordered TLV List		Input List
Set Variable		✓		✓					
Assign and Branch		✓		✓	✓	✓	✓		
Extract		✓							
Go Back	1								
Branch on Variable Value		✓	✓			✓	✓		
Exit	1								
Execute SAT Command	1	✓							
Execute Native Command	1		✓						✓
Get Length		✓	✓						
Get TLV Value		✓	✓						
DISPLAY TEXT				✓					
GET INPUT				✓	✓				

8.1 Set Variable

This byte code sets one or more variables to a value contained in the corresponding Inline Value TLV. This byte code can be used to e.g. copy the content of one variable to another variable or to concatenate a list of variables and/or constant text into another variable.

Length	Value	Description	M/O
1	'14'	Set Variable Tag	M
1-3	1+A+...+1+X	Length	M
1	Data	Variable ID	M
A	TLV	Inline Value TLV or Variable Reference List TLV	M
...	...	...	
1	Data	Variable ID	O
X	TLV	Inline Value TLV	O

Possible errors:

Error Code	Description	Action
No error	OK	Continue
Syntax error	Syntax error	Stop
Reference to undefined	Reference to undefined variable	Stop
Problem in memory management	Memory allocation problem	Stop

At least one pair of Variable ID and Inline Value TLV or Variable Reference List TLV shall be present in the Set Variable byte code.

8.2 Assign and Branch

This byte code displays a menu on the UE and assigns a selected value to a variable according to the selection of the user.

Length	Value	Description	M/O
1	'15'	Assign and Branch Tag	M
1-3	1+A+...+1+X	Length	M
1	Data	destination Variable ID, identifier of the variable to be set	M
A	TLV	Inline Value TLV: Contains the select item alpha-identifier (according to 3GPP TS 31.111 [2])	O
B	TLV	Ordered TLV List TLV (see description below) containing possibly: - Inline Value 2 TLV - Inline Value TLV - Page Reference TLV	M
...	...	...	
X	TLV	Ordered TLV List TLV (see description below)	O

Possible errors:

Error Code	Description	Action
No error	OK	Continue
Reference to undefined	Reference to undefined variable	Stop
Problem in memory management	Memory allocation problem	Stop
Syntax error	Syntax error (for example: try to initialise a text element)	Stop
USAT command failed	USAT command failed.(SELECT ITEM could not be built)	Stop

Explanation of used arguments:

8.2.1 Destination Variable Identifier

The content of this value identifies the destination variable. The value contained in the selected Inline value TLV within the Ordered TLV List TLV will be assigned to this destination variable by the USAT Interpreter.

8.2.2 Inline TLV containing Select Item Title

The content of this TLV is running text which specifies the alpha identifier to be used by the USAT Interpreter when generating a SELECT ITEM command from the "Assign and Branch" byte code according to 3GPP TS 31.111 [2].

8.2.3 Ordered TLV List

One or more of this TLVs shall be contained in the "Assign and Branch" byte code.

This TLV encapsulates the

- "Inline Value 2"
- "Inline Value" and
- "Page Reference"

TLVs in the given order, which determine the action to be performed.

General variable assignments and navigation operations may be performed by the "Assign and Branch" byte code dependent on the data provided in the Ordered TLV List TLV. When optional TLVs are omitted, special cases can be encoded according to the following table:

Inline Value 2	Inline value (to be assigned to destination variable)	Page Reference	
present	present	present	**"Display, Assign and Branch"** Generation of a SELECT ITEM command by the USAT Interpreter. When the user has selected this item from the list, the USAT Interpreter shall assign the value of the Inline value TLV to the destination variable and branch to the navigation unit specified within the Page Reference TLV.
present	present	not present	**"Set Variable Selected"** Generation of a SELECT ITEM command by the USAT Interpreter. When the user has selected this item from the list, the USAT Interpreter shall assign the value of the 'Inline value' TLV to the destination variable.
present	not present	present	**"Go Selected"** Generation of a SELECT ITEM command by the USAT Interpreter. When the user has selected this item from the list, the USAT Interpreter shall branch to the navigation unit specified within the Page reference TLV. A destination variable identifier shall be ignored for this case.
present	not present	not present	**"Select Item"** Generation of a SELECT ITEM command by the USAT Interpreter. When the user has selected this item from the list, the USAT Interpreter shall process the next byte code. A destination variable identifier shall be ignored for this case.
not present	present	present	**"Assign and Branch"** No generation of a SELECT ITEM command by the USAT Interpreter. The USAT Interpreter shall assign the value of the 'Inline value' TLV to the destination variable and branch to the navigation unit specified within the Page reference TLV.
not present	present	not present	**"Set Variable"** No generation of a SELECT ITEM command by the USAT Interpreter. The USAT Interpreter shall assign the value of the Inline value TLV to the destination variable.
not present	not present	present	**"Direct Go"** No generation of a SELECT ITEM command by the USAT Interpreter. The USAT Interpreter shall directly branch to the navigation unit specified within the Page reference TLV. The destination variable identifier shall be ignored for this case.
not present	not present	not present	not valid, if occurs an error shall be issued.

8.3 Extract

This byte code extracts a byte array from a value and stores the result in a variable.

Length	Value	Description	M/O
1	'16'	Extract Tag	M
1	4	Length	M
1	Data	Variable ID, which will contain the result	M
1	Data	Variable ID, containing the source data	M
1	I	Zero based start index in the byte array	M
1	N	Maximum number of bytes to extract, '00' indicates to retrieve all remaining bytes	M

Possible errors:

Error Code	Description	Action
No error	OK	Continue
Syntax error	Syntax error	Stop
Problem in memory management	Memory allocation problem	Stop
Reference to undefined	Reference to undefined variable	Stop
Out of range	Index out of range.	Stop

8.4 Encrypt

This is for further study.

8.5 Decrypt

This is for further study.

8.6 Go Back

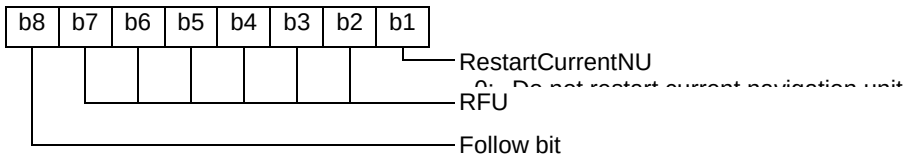
This byte code forces branching to the last anchor pushed on the history list. It has no impact on the history list itself.

Length	Value	Description	M/O
1	'19' / '99'	Go Back Tag	M
1	A	Length	M
A	Data	Attributes	O

Possible errors:

Error Code	Description	Action
No error	OK	Continue
Jump to undefined	Reference to undefined (case of history empty)	Stop

Attributes:



8.7 Branch On Variable Value

This byte code compares a variable to a list of values that have an associated Page Reference and when a match is found, the referenced page is executed. If no match is found, the first Page Reference after the Ordered TLV List is used to branch. If this last Page Reference TLV is not contained in the byte code, no branch is executed and the USAT Interpreter continues to render the next byte code after the Branch on Variable Value byte code.

Length	Value	Description	M/O
1	'1A'	Branch on Variable Tag	M
1	1+A+...+X+Y	Length	M
1	Data	Variable ID (containing the match value)	M
A	TLV	Ordered TLV List TLV (see description below) containing: - Variable Identifier List TLV (referring to one variable containing the value to be compared with the match value, additional Variable IDs to be ignored) - Page Reference TLV, to branch to, if values match	M
...	...	...	
X	TLV	Ordered TLV List TLV	O
Y	TLV	Page Reference TLV, if no match is found, go to this reference	O

Possible errors:

Error Code	Description	Action
No error	OK	Continue
Reference to undefined	Reference to undefined variable	Stop
Jump to undefined	Page Reference not found.	Stop

Explanation of used arguments:

8.7.1 Variable ID

This variable shall contain the value to be compared.

8.7.2 Ordered TLV List

In each of these TLVs the following TLVs are encapsulated:

- Variable Identifier List TLV (referring to one variable containing the value to be compared with the match value; additional Variable IDs to be ignored)
- Page Reference TLV.

The Page Reference TLV contains the location to be branched to, if the comparison is successful.

8.7.3 Page Reference

If no match was found, the reference contained in here is used to branch. If this TLV is not available, no branch is executed and the USAT Interpreter continues to render the next byte code after the Branch on Variable Value byte code.

8.8 Exit

If the TerminateSession Attribute is not set, the USAT Interpreter shall behave as defined by the current Navigation Event Handler for the case of " Proactive SIM session terminated by the user".

If the TerminateSession Attribute is set, the proactive session is terminated immediately by the USAT Interpreter. The USAT Interpreter shall respond to the UE with SW1/SW2='9000' in this case.

If the USAT Interpreter had been USIM internally (by an proprietary internal interface) called, the Variable Identifier List TLV can be used to provide return values to the calling function. Handling of these internal return values are out of the scope of this document.

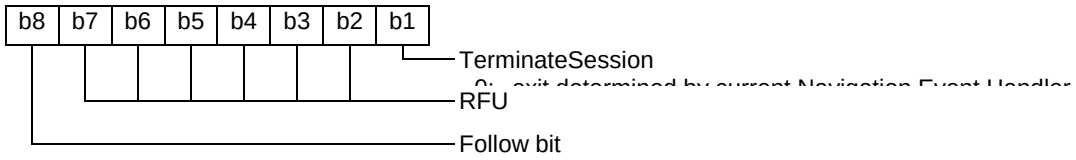
If the USAT Interpreter does not support the mechanism of providing return values, it shall ignore the possibly available Variable Identifier List TLV.

Length	Value	Description	M/O
1	'1B' / '9B'	Exit Tag	M
1	1+A	Length	
L	Data	Attributes	M
A	TLV	Variable Identifier List TLV (containing return values)	O

Possible errors:

Error Code	Description	Action
No error	OK	Stop
Reference to undefined		Stop

Attributes:



8.9 Execute USAT Command

This byte code executes an USAT command using the provided arguments.

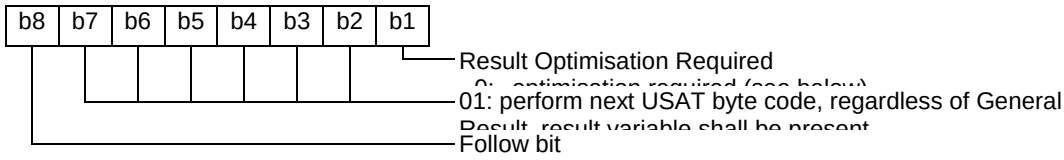
Length	Value	Description	M/O
1	'1C' / '9C'	Execute USAT Command Tag	M
1	A+4+B	Length	M
A	Data	Attributes	O
1	Variable ID	General Result code from Terminal Response	O
1	Variable ID	Variable to hold the output of the USAT command	O
1	Cmd type	Command type value according to 3GPP TS 31.111 [2]	M
1	Cmd qual.	Command qualifier value according to 3GPP TS 31.111 [2]	M
1	Dest dev.	Destination device according to 3GPP TS 31.111 [2]	M
B	TLVs and Simple TLV Indicators	Sequence of - simple TLVs of the proactive command as defined in 3GPP TS 31.111 [2] - and / or Simple TLV Indicators	O

Possible errors:

Error Code	Description	Action
No error	OK	Continue
Reference to undefined		Stop
Problem in memory management	Memory problem in the preparation of the SAT command	Stop
Syntax error	Try to initialise a text element	Stop
USAT command failed	USAT Command could not be delivered to UE	Stop
USAT command not allowed		Stop

Explanation of used arguments:

8.9.1 Attributes



8.9.2 Simple TLV

This TLV shall be a simple TLV coded as described in 3GPP TS 31.111 [2] for the USAT proactive command to be executed.

8.9.3 Simple TLV Indicator

A Simple TLV Indicator is a placeholder for a Simple TLV. A Simple TLV Indicator is coded as follows:

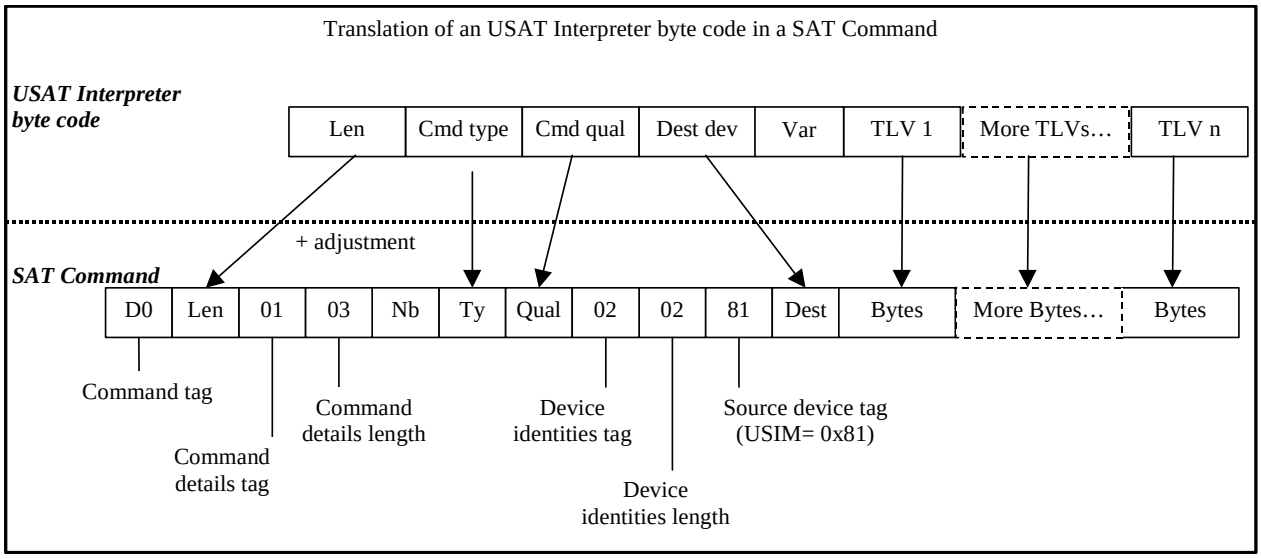
Coding	Description
'00'	This value indicates the Simple TLV Indicator
Length	This value indicates the length of the following data belonging to the Simple TLV Indicator
Result Tag	This value represents the Tag value of the resulting Simple TLV
Simple TLV Indicator content	The Simple TLV Indicator content may contain escape characters to indicate variable references. Therefore the Simple TLV Indicator content has to be byte-stuffed. The possibly available variable references have to be expanded according to chapter "6.3 Variable substitution" Method 1 during processing of this indicator by the USAT Interpreter.

The result of processing the Simple TLV Indicator shall be a Simple TLV. When the USAT Interpreter processes a Simple TLV Indicator the Result Tag shall be the Tag of the resulting Simple TLV. The value part shall be formed of the Simple TLV Indicator content and the length of the resulting Simple TLV is the length of the Simple TLV Indicator content after possible variable substitution.

8.9.4 Sequence of Simple TLVs and Simple TLV Indicators

The sequence of these Simple TLVs and Simple TLV Indicators is translated by the USAT Interpreter to form the sequence of Simple TLVs of an USAT command (3GPP TS 31.111 [2]). When expanding Simple TLV Indicators to Simple TLVs the length of the BER-TLV of the resulting USAT command shall be adjusted by the USAT Interpreter before issuing the command to the UE.

When executing a Execute SAT command byte code, the USAT Interpreter issues a normal SAT command to the UE using the USAT protocol. The translation procedure from the Execute SAT Command TLV to an USAT command can be visualised in principle as follows:



8.9.5 Result of an Execute USAT Command

The result is of executing an USAT command is a Terminal Response containing a list of Simple TLVs as defined in 3GPP TS 31.111 [2]. The list of TLVs is processed by the USAT Interpreter as specified in the following 2 chapters.

8.9.5.1 Optimisation not Required

The complete proactive command specific response data (Terminal Response according to 3GPP TS 31.111 [2]) is stored in the result variable

8.9.5.2 Optimisation Required

Only the first TLVs after the Result Simple TLV within a Terminal Response (see 3GPP TS 31.111 [2]) will be processed by the USAT Interpreter as follows:

- If the first TLV after the Result Simple TLV is a Text String TLV according to 3GPP TS 31.111 [2], the value part without the DCS byte is assigned to the result variable. The DCS is removed from the V field, but used for variable management internally the USAT Interpreter.
- In all other cases, the value part of the first TLV after the Result Simple TLV is assigned to the result variable.

8.10 Execute Native Command

This byte code is used to execute an operating system call, "plug-in" or application external to the USAT Interpreter.

The attribute indicates if the execution returns to the USAT Interpreter or not. Arguments are passed for input and output. The output is stored in a list of variables.

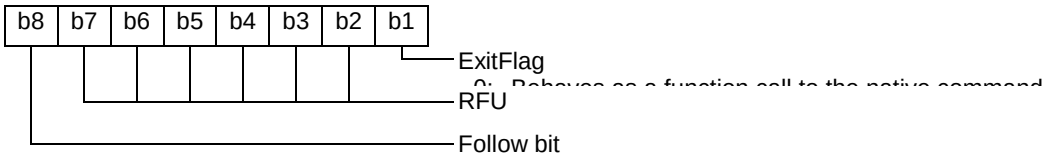
Length	Value	Description	M/O
1	'1D' / '9D'	Execute Native Command Tag	M
1	A+4+B	Length	M
A	Data	Attributes	O
1	B	Length of following AID	M
B	Data	AID of application or plug-in	M
C	TLV	Input List TLV containing arguments	O
D	TLV	Variable ID List TLV for output of application or plug-in	O

Possible errors:

Error Code	Description	Action
No error	OK	Continue
Reference to undefined	Reference to undefined	Stop
Jump to undefined	Execute element does not exist	Stop
Problem in memory management	Memory problem in the preparation of the structure	Stop
User Abort	Execute was aborted by user	Stop
Syntax Error	Incorrect number of arguments passed to the execute element.	Stop
Execution Error	Execute element generated an internal error.	Stop

Explanations:

8.10.1 Attributes



8.10.2 Result of a Native Function Call

If the native function call returns, the values produced by the call are the values associated with the variable references in the output list.

May be more precise.

8.11 Get Length

This byte code instructs the USAT Interpreter to calculate the length of all variable contents of the variables in the Variable List and to assign the result to the output variable.

Length	Value	Description	M/O
1	'1E'	Get Length Tag	M
1-3	1+A	Length	M
1	Data	Variable ID (output, containing the BER encoded result length in binary format according to ISO/IEC 7816-6 [5])	M
A	TLV	Variable List TLV, Variable List for length calculation	M

Possible errors:

Error Code	Description	Action
No error	OK	Continue
Reference to undefined	Reference to undefined variable	Stop

8.12 Get TLV Value

This byte code instructs the USAT Interpreter to extract the value part of a TLV from a sequence of TLVs and to assign the resulting value to the output variable.

If the requested tag value is not found in the sequence of TLVs, the output variable is generated with no content (i.e. the length of content of the variable is 0).

Length	Value	Description	M/O
1	'1F'	Get TLV Value Tag	M
1-3	1+1+A	Length	M
1	Data	Variable ID (output, containing the value of the requested TLV)	M
1	Data	Tag value to search for	M
A	TLV	Variable List TLV, each referenced variable shall contain a list of TLVs (e.g. generated Terminal Response of Execute USAT Command)	M

Possible errors:

Error Code	Description	Action
No error	OK	Continue
Reference to undefined	Reference to undefined variable	Stop

8.13 DISPLAY TEXT

This byte code instructs the USAT Interpreter to issue a DISPLAY TEXT command according to 3GPP TS 31.111 [2].

This command is used to display text of informational nature without require any input from user. The USAT Interpreter shall use the DCS value according to the indication given in the attributes of the Inline Value TLV. If no attributes are given in the Inline Value TLV, the coding scheme indication of the current page shall be used.

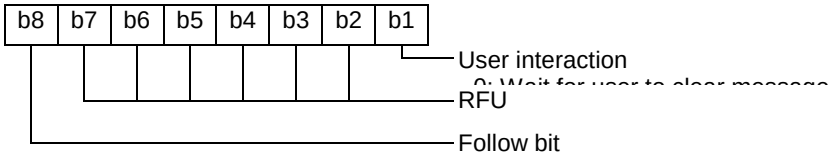
Support for packed text: check impact of type coding Inline.

Length	Value	Description	M/O
1	'20'	DISPLAY TEXT Tag	M
1-3	1+A	Length	M
1	Data	Attributes	O
A	TLV	Inline Value TLV, containing text to be displayed	M

The following parameters shall be used for the generated DISPLAY TEXT command:

Field	Comment
Command Details according to 3GPP TS 31.111 [2]	High Priority shall always be used. Other command parameters shall be used according to the information provided in the attribute byte.

Coding of the attributes:



Possible errors:

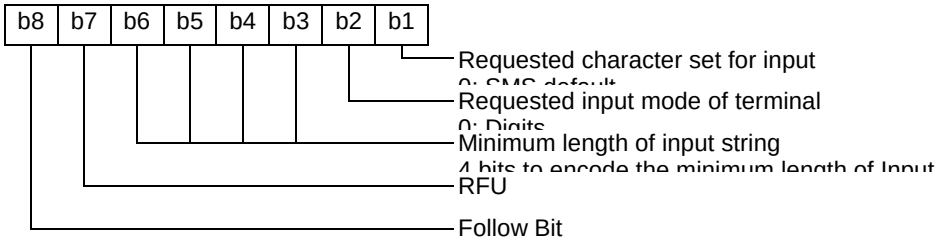
Error Code	Description	Action
No error	OK	Continue
Reference to undefined	Reference to undefined variable	Stop

8.14 GET INPUT

This command is used to request multiple character input from user.

Length	Value	Description	M/O
1	'21'	GET INPUT Tag	M
1-3	1+A+B	Length	M
1	Data	Attributes	O
1	Data	Variable ID (for storing the entered characters, the variable type information of the variable is set according to the DCS indication received from the UE. The DCS received from the UE is not stored in the variable value.)	M
A	TLV	Inline Value TLV, containing text to be displayed (e.g. the question, to be used in the text string TLV of the GET INPUT SAT command)	M
B	TLV	Inline Value 2 TLV, containing the default text for the default text TLV of the GET INPUT SAT command	O

Coding of the attributes:



The following parameters shall be used for the generated GET INPUT command:

Field	Comment
Response length	Minimum length: the value supplied by the attribute byte is to be used; Maximum length: 'FF' shall to be used. UE
Command Qualifier	UE may echo user input on the display; User input to be in unpacked format; No help information available;

If more parameters are necessary for the GET INPUT command, for security reasons, the generic USAT interface shall be used.

Possible errors:

Error Code	Description	Action
No error	OK	Continue
Reference to undefined	Reference to undefined variable	Stop

11 Error coding

For the indication of errors occurring during byte code processing error codes listed in the following table are defined. This information can be accessed using the Error Code variable.

Type of error	Coding
No error	‘0000’
Syntax error	‘6F01’
Jump to undefined	‘6F02’
Problem in memory management	‘6F03’
Security problem	‘6F04’
Reference to undefined	‘6F05’
Out of range	‘6F06’
User abort	‘6F07’
Execution error	‘6F08’
USAT command failed	'6F09'
USAT command not allowed	'6F0A'
USAT Interpreter transmission error	'6F0B'
General unspecific error	‘6FFF’

Annex A (Informative): Native Commands

Each native command or plug-in is identified by an AID. Where a plug-in is common across application providers, the 3GPP RID ('A000000087') shall be used together with a PIX from Appendix A of 31.111. The following is an example of a commonly used plug-in.

A.1 ConvertTextPhoneNumberToGSMPhoneNumber

Description: Allows converting a text string containing a phone number to a GSM 11.11 dialling number string, before using it in a Setup Call command (for example).

Attribute byte: Exit Attribute not set (call)

Input arguments:

Variable containing a text phone number (result of a user input for example)

Output arguments:

Variable to contain the new phone number format: Length + TON/NPI + swapped nibbles coded according EF_{ADN} of GSM 11.11

Errors:

EXECUTE tag will return "Execution Failure" in case of conversion error.

Remarks:

- The input variable must contain only digit characters and optionally "+",."*", "#", and "," characters in SMS alphabet
- The "+" sign or a double zero ("00") as first part of the input variable indicates an international number (TON/NPI will be set to '91'), in the other case the TON/NPI will be set to national ('81').
- The resulting value contains the length in its first byte (the total number of following bytes, including the TON/NPI byte) and is composed of digit characters and "A", "B", "C" signs, as specified in GSM 11.11 (EF_{ADN} coding and extension 1).

Examples:

- The execution of the plug-in with an input variable containing the string "+33442365000" will produce in the resulting variable the bytes '07 91 33 44 32 56 00 F0' and no error will be produced.
- The execution of the plug-in with an input variable containing the string "0442365000" will produce in the resulting variable the bytes '06 81 40 24 63 05 00' and no error will be produced.
- The execution of the plug-in with an input variable containing the string "04Z42365000" will not produce anything in the resulting variable and the error "Execution failure" will be produced.

History

Document history		
V0.0.9	January 2001	First full ETSI style version
V0.0.10	February 2001	Prepared version for ad-hoc #25 05-07.02.2001 Stockholm
V0.0.11	February 2001	New version after ad-hoc #25 and #29
V1.0.0	March 2001	Presented for information to TSG-T #11