S@T 01.50 v2.0.0 (Release 2004)

S@T Browser Behavior Guidelines

Published by simalliance now Trusted Connectivity Alliance

TABLE OF CONTENTS

1 TERMINOLOGY

1.1 Abbreviations

SBC	S@T Byte Code
SSP	S@T Session Protocol
STLS	S@T Transport Layer Security
S@TML	S@T Markup Language
HTTP	Hyper Text Transfer Protocol
S@T	SIM Alliance Toolkit
STK	SIM Application Toolkit
TLV	Tag Length Value encoding
URL	Unified Resource Location

2 LIST OF DOCUMENTS

/GSM 03.38/	GSM 03.38: "Digital cellular telecommunications system (Phase 2+); Alphabets and language-specific information".
/GSM 11.11/	GSM 11.11. "Digital cellular telecommunications system (Phase 2+); Specification of the Subscriber Identity Module – Mobile Equipment (SIM – ME) interface".
/GSM 11.14/	GSM 11.14. "Digital cellular telecommunications system (Phase 2+); Specification of the SIM Application Toolkit for the Subscriber Identity Module – Mobile Equipment (SIM – ME) interface".
/GSM 03.48/	GSM 03.48. "Digital cellular telecommunications system (Phase 2+); Security Mechanisms for the SIM Application Toolkit; Stage 2".
/SBC/	SBC, S@T Byte Code (Technical Specification S@T 01.00)
/SSP/	SSP, S@T Session Protocol (Technical Specification S@T 01.20)
/Admin/	S@T Administration Commands (Technical Specification S@T 01.21)
/Operational/	S@T Operational Commands (Technical Specification S@T 01.22)
/Push/	S@T Push Commands (Technical Specification S@T 01.23)

This document is part of a specification set, please refer to "S@T Release Note" for a comprehensive document list, including document versions.

3 OVERVIEW

This document describes the S@T browser behavior and gives guidelines for its implementation.

The objective is to define all aspects of the browser and its implementation that have impacts on the interoperability between different suppliers of the S@T browsing system.

4 MEMORY MANAGEMENT

4.1 Buffer Management

4.1.1 Resident Decks Buffer

The resident decks buffer is located in non-volatile memory.

There is no limitation concerning the minimum and the maximum size of the resident decks buffer.

If installation of a new resident deck fails, there is no access to this deck or parts of it. There is no impact on previously installed resident decks.

Note: Some URLs of resident decks are related to specific internal functionality of the browser (see annex: Reserved URLs).

4.1.2 Temporary Decks Buffer

The temporary decks buffer (reception buffer for the temporary decks) may be located either in volatile or non-volatile memory.

The minimum size of the temporary decks buffer is 1 Kbytes.

4.2 SPS Management

SPS support in the browser is indicated in the browser profile sent by the browser during the connection phase.

If allocated, the minimum size of the SPS is 64 bytes.

4.3 Variables Memory Location and Validity Time

4.3.1 Temporary Variables

Temporary variables may be stored in volatile or non-volatile memory.

The minimum memory size for the allocation of temporary variables is 512 bytes for values and variable management information (type, DCS, etc.).

The validity time of temporary variables is equivalent to the duration processing one or several decks. If a temporary variable is not explicitly deleted using a byte code macro or an attribute in the deck, it may still be available in the next deck.

Since sharing of temporary variables between resident and temporary decks is not defined yet, it is highly recommended to clean up all temporary variables when leaving a resident deck.

Because of the limited size of the temporary variables buffer, the browser may delete existing variables. If such a variable is referenced later in a deck (read access), a NULL string will be returned.

4.3.2 Permanent Variables

Permanent variables shall be stored in non-volatile memory.

There is no validity time for permanent variables.

4.3.3 Environment Variables

System variables and user preferences shall be stored in non-volatile memory.

Runtime variables don’t have to be stored in non-volatile memory: their value is assigned every time they are accessed.

5 VARIABLES MANAGEMENT

There are two basic types of variables (see /SBC/):

- Application variables (temporary variables, text elements and permanent variables) and
- Environment variables.

5.1 Internal Variable Structure

There is no additional information about the variable type stored in the value part. The DCS attribute to identify a variable type shall be managed internally.

5.2 Variables DCS handling during Byte Code processing

The browser profile indicates the support of UCS2 entry and UCS2 display of the mobile phone and the browser itself. If a deck with a DCS indicating UCS2 is received and the browser does neither support the UCS2 entry nor the UCS2 display, the browser shall generate an error.

For the different byte code macros (see /SBC/) the following rules apply:

5.2.1 INIT VARIABLES and INIT VARIABLES SELECTED

If a variable reference is used to initialise an existing variable, the memorised DCS of the referenced variable shall be mapped to the initialised variable.

If an inline value is used, the DCS to be taken is defined by the corresponding attribute of the inline value structure (see /SBC/).

5.2.2 SWITCH CASE ON VARIABLE

If the variables to be compared have a different DCS or a different length, the result of the comparison is ‘not match’ and the byte code macro execution is continued according to this result.

5.2.3 STK GENERIC

5.2.3.1 STK Proactive Command String

In any proactive command string where a DCS is needed (e.g. in a text string object), the browser inserts a DCS byte according to the following rules:

- In case of variable substitution the deck DCS attribute is inserted if a text element content is used as part of the proactive command.
- An individual, variable related DCS is inserted, if a temporary or permanent variable content is used as part of the proactive command string.

5.2.3.2 STK Command Output Value

If LV encapsulation of the returned string is not required (default) and the returned data part of the terminal response is a simple TLV object, and this is a text string object including a DCS byte, the DCS is removed before storing the value in the destination variable. The DCS byte must be memorised by the browser for later type checking (e.g. within the byte code macro ‘CONCATENATE’).

If LV encapsulation of the returned string is not required but the returned data part of the Terminal Response consists of several TLV objects, the LV part of the first TLV object is to be stored in the destination variable and all further TLV objects are discarded.

5.2.4 CONCATENATE

The browser must check the DCS of variables and inline values before concatenation. On any type mismatch the browser generates an error. If one element to concatenate is a text element, the DCS is inherited from the deck level. If the DCS of the elements matches, it is memorised for the destination variable.

If one element to be concatenated contains a NULL string, this element is not evaluated.

5.2.5 EXTRACT

The memorised DCS of the referenced variable shall be mapped to the destination variable.

5.3 Error handling

The following error handling applies during variables processing:

- In the case a variable is empty, the browser shall use a NULL string as value.
- In the case an application variable does not exist (is not accessible), the browser shall stop processing the current deck and indicate an error ‘Reference to undefined’. If the browser does not support SPS, variables in the range of variable identifiers reserved for permanent variables (see /SBC/) are not accessible. This only refers to permanent variables. For temporary variables in this case a NULL string will be returned as value.
- In case a temporary variable is not initialised, the browser shall return a NULL string as value.
- In the case an environment variable does not exist (is not accessible), the browser shall return a NULL string as value.

6 PROCESSING DECKS AND CARDS

6.1 Navigation

6.1.1 Starting a Card in a Deck

If a deck is referenced in another deck without specifying a card name, the first card of the referenced deck will be executed. Nevertheless a precise card in a deck can also be called by defining the specific card name.

6.1.2 Navigation through Decks

If a deck is unsuccessfully requested from the gateway, the error handling is performed by the gateway, no error related action is to be performed by the browser.

If a resident deck cannot be accessed for any reason, the browser shall indicate an error of type ‘Jump to undefined’ and call an appropriate error handler.

6.1.3 Navigation through Cards

If a card cannot be accessed for any reason, the browser shall indicate an error of type ‘Jump to undefined’ and call an appropriate error handler.

6.2 Handling of unknown Tags and Structures

To consider a Tag or a Structure as “unknown”, the browser shall **not** take into account the position of the Tag or the Structure within the byte code.

Regardless whether the browser checks the order (position) of the presented byte code elements or not, it cannot be assumed, that a received byte code structure will be processed correctly, if it contains elements not presented in the same order as in the S@T specifications.

If a browser checks the order of the presented byte code elements and a valid element is found at a wrong position, the error case “unknown Tag” shall **not** apply. This case forces the browser to stop processing and a **syntax error** shall be indicated.

6.2.1 Unknown SSP Commands

Any unknown SSP command shall be ignored by the SSP layer (see /SSP/) in the browser and all following data in the current message (one or several SMS) shall be discarded.

6.2.2 Unknown operational Commands

In operational mode any unknown command is skipped and the browser continues processing the next command.

6.2.3 Unknown administrative Commands

In administrative mode any unknown command is skipped and the browser continues processing the next command.

6.2.4 Unknown Byte Code Structures

Any unknown byte code structure encapsulated in another byte structure (i.e. the Tag value of the encapsulated byte code structure is not valid within the encapsulating structure) must be skipped and the browser continues processing the next byte code structure. The status word is not affected.

6.2.5 Unknown Attributes

If an unknown attribute within a byte code structure is found, the browser continues processing, and the status word is not affected.

6.3 Byte Code Macros containing incorrect STK Command Parameters

The browser only assures that no error in the proactive communication (STK protocol level (see /GSM 11.14)) can occur, but it is up to the implementation, if the texts are truncated or an exception is issued by the browser; the application level is not affected.

6.4 Browser Idle Mode

After the execution of the last byte code in a card and if the attribute 'ChainNextCard' is not set, the browser turns to idle mode, i.e. a 'Display Text' command is issued to keep the proactive session alive and allow the user to call a contextual menu as defined in /SBC/, chapter 5.2.2 for further navigation.

After the execution of the last byte code of the last card of a deck (even if the attribute 'ChainNextCard' is set) the browser turns to idle mode, i.e. a 'Display Text' command is issued to keep the proactive session alive and allow the user to call a contextual menu as defined in /SBC/, chapter 5.2.2 for further navigation.

6.5 Pause Browser

If the user requests to pause the browser, the browser stops issuing proactive commands. However, the proactive session may be still alive, if the browser has been called by another application.

All application and user specific (temporary) data have to be kept by the browser when it is in the pause state. In addition, all status information needed to resume the service shall be stored by the browser.

6.6 Resume Browser

If the browser is in a pause state and the browser is recalled, the browser offers the user to restore the old session or to initiate a new one.

If the user chooses to resume the old session, the browser continues processing the byte code macro, being executed when the user paused the browser. The browser shall present again the previous screen to the user.

All application and user specific (temporary) data are unchanged when the browser resumes.

6.7 Browser Waiting Mode

When the user requests a distant URL, the browser issues a 'Send SMS' command. After a successful terminal response - while waiting for the response from the gateway - the browser shall do a loop on a 'Display Text' command to keep the proactive session alive.

The browser must provide a method to the user to abort the current request. If the current request is aborted, the browser shall not process the expected deck.

If the current request is aborted before the first gateway 'Connect Response' in the session has been received by the browser, the connection may not be terminated.

If the current request is aborted before the first gateway 'Connect Response' in the session has been received by the browser and if another request is performed by the user before receiving the 'Connect Response', then a new 'Connect Request' and a new 'Express Data Request' shall be formatted.

6.8 Contextual Menu Management

On exit of the contextual menu with no specific action to be performed (quit the contextual menu) browsing goes on with the macro which has triggered the contextual menu.

If a system contextual menu item is addressed in the byte code macro 'MANAGE CONTEXTUAL MENUITEM' and the specified item identifier is not defined, then the error code 'Reference to undefined' shall be returned by the byte code macro.

The same error condition applies for the macro 'ADMIN MANAGE CONTEXTUAL MENUITEM'.

The macro 'ADMIN MANAGE CONTEXTUAL MENUITEM' results in a persistent change of the status of the addressed menu item, i.e. the default visibility for system items is changed or application items with the operator attribute set can be added/updated or removed. If an application item is addressed and the operator attribute is not set, then the error code 'Type mismatch' will be returned.

The following tables show the different options of the macro 'MANAGE CONTEXTUAL MENUITEM' for the operational and the macro 'ADMIN MANAGE CONTEXTUAL MENUITEM' for the administrative mode:

Operational Mode	System	Application
Card specific item	*Temporary change of visibility*	*Temporary change (item added/updated or removed)*
Operator specific item	*Error: Type mismatch*	*Error: Type mismatch*

Administrative Mode	System	Application
Card specific item	*Permanent change of visibility*	*Error: Type mismatch*
Operator specific item	*Permanent change of visibility*	*Permanent change (item added/updated or removed)*

If by an operational or administrative macro '[ADMIN] MANAGE CONTEXTUAL MENUITEM' the system menu item 'Stop browser' shall be set to 'not visible', this macro is ignored by the browser, the return code 'No error' is provided and the browser processes the next byte code macro.

6.9 Execute Elements Management

An execute element must check itself if all needed input parameters are available, i.e. the browser is not responsible for any error management needed while calling the executable element.

6.10 URL Validity Time

Any URL reference (coded name) is only valid within one browsing session (related to a certain Session ID).

7 BROWSER START UP PROCEDURE

7.1 Memory Management and Data Initialisation

All temporary variables shall be deleted in the start up phase of the browser.

Memory parts of the browser maintained during runtime shall be set to its initial values, e.g. the history for browser navigation shall be cleared.

7.2 Browser Interaction with the Gateway

It is not mandatory for the browser to interact with the gateway when it starts, i.e. the connection phase is done either when the browser starts or at the first request.

7.3 Start Up Mechanism

The browser may be started by several means, e.g. by a menu selection of the mobile user or any resident STK application. This allows calling the browser with a specific URL as parameter.

7.4 Starting Page

By default (i.e. whenever no specific URL is provided), the home page is displayed when starting the browser. The home page can be a resident deck identified using a specific deck name: /s/home (see annex: Reserved URLs). This deck name is reserved for the homepage defined by the operator.

8 BROWSER SHUT DOWN PROCEDURE

8.1 Memory Management and Data Saving

During the shutdown procedure, the browser saves all relevant data in non-volatile memory, e.g. the latest values of the runtime variables, if necessary.

It is in the responsibility of the browser, that after the shutdown procedure has been performed, no data fragments are present in the different buffers, that may cause any errors.

Memory parts of the browser maintained during runtime shall be set to its initial values, e.g. the history for browser navigation shall be cleared.

8.2 Browser Interaction with the Gateway

When the browser stops, any interaction with the gateway before performing the shut down procedure is optional.

8.3 Information given to the User

It is not mandatory to give any information to the user when the browser shuts down.

9 Fields of the 03.40

To ensure interoperability on low level transport protocol, the usual fields of the 03.40 header should be set as specified below for mobile originated short message.

Abbr.	Value	Meaning
TP-MTI	0b01	Submit SMS
TP-RD	0b0	(No interoperability impact)
TP-VPF	0b10	(No interoperability impact)
TP-SRR	0b0	(No interoperability impact)
TP-UDHI	0b1	User data header present
TP-RP	0b0	(No interoperability impact)
TP-MR	XX	Managed as specified in 03.40.
TP-DA	Address of the server	
TP-PID	0x00	Implicit protocol (No interoperability impact)
TP-DCS	0xf6	Class 2: SIM specific,(8 bit)
TP-VP	XX	(No interoperability impact)

9.1 Server Address Management

The browser should be able to accept messages from any known server. A list of server addresses is available on the card and the browser checks the TP-OA of incoming SMS. The responses send (if any) must use the previous TP-OA. During the same session the browser can communicate only with one server. The reception of a SMS from an unauthorised (not listed on the card) server is discarded.

10 ERROR MANAGEMENT

10.1 Setting the Runtime Variable ‘Status Word’

After the execution of each byte code macro, the execution result shall be stored in the runtime variable ‘status word’. In each success case, the value 0x0000 shall be stored in the status word. If the execution fails, a specific error code is to be stored. The list of available error codes is provided in /SBC/.

10.2 Error handling

If the status word variable contains any error code (not 0x0000) and the error related action indicated in /SBC/ is ‘Stop browser’, a user notification shall be given by the browser. All further actions are implementation specific (‘Error Handler’).

10.3 Handling of specific Errors

10.3.1 Mandatory Parameter not found

If a mandatory parameter within a byte code macro is not found, the error code ‘Syntax error’ shall be written to the status word.

10.3.2 Incorrect Length Coding

If any length within a TL[A]V element is not coded in BER-TLV format, the error code ‘Syntax error’ shall be written to the status word.

10.4 03.40 Error handling

If a 03.40 error appears, the message should not be treated and no user notification is used. The 03.40 errors include bad concatenation header.

List of the possible error of the 03.40 header:

- UDHI set to 0
- Bad DCS (others than specified in chapter 9 ‘Fields of the 03.40’)
- Unauthorised Server address
- Bad header tag (others than concatenated header or 03.48 header)
- Bad header format (others than concatenated header or 03.48 header)

Error in the concatenation header:

- SMS number higher than the SMS maximum number
- SMS number set to 0

10.5 SSP Error handling

If a SSP error appears, the further processing of the message is stopped. An unknown SSP tag is considered as an error.

11 EXTENSION MANAGEMENT

For the future new byte codes and extended functionality will be added to the S@T browser.

The implementation of the browser shall be open to handle additional TLV-objects / byte code macros in the deck and card structure.

Backward compatibility can be achieved by evaluation of the version number within the browser profile. Current version (*S@T Version Number* value) is 0x1X, related to the documents mentioned in chapter 2 ‘List of documents’.

12 MULTIPLE SSP SESSIONS MANAGEMENT

12.1 Multiple Sessions Support

Multiple SSP session support is not mandatory for the browser.

A S@T browser not supporting multiple sessions will refuse the connection messages coming from a S@T gateway.

12.2 Blocking Cases Management

The browser has always the priority on the gateway.

To avoid blocking cases, the following rules apply:

For browsers which do not support multiple sessions:

- If the browser requests a new session (‘Connect Request’), then it will ignore or refuse (by sending back a ‘Connect Response’ with an error status) all gateway ‘Connect Request’ until it receives the ‘Connect Response’ of its pending ‘Connect Request’.
- If the gateway requests a new session (‘Connect Request’), then it will accept a browser ‘Connect Request’ (and will be ready to receive a ‘Connect Response’ with an error status from the browser for its pending ‘Connect Request’).
- If a session is established, then the browser/gateway will ignore or refuse (by sending back a ‘Connect Response’ with an error status) all the gateway/browser ‘Connect Request’.
- Is a session is established, the browser/gateway will not request a new session.

13 ANNEX

13.1 Minimum Requirements

13.1.1 TL[A]V Attributes

The browser shall process a minimum of 3 attribute bytes correctly. More attribute bytes may lead to an error or may be skipped correctly (implementation specific).

13.1.2 Resident Deck Buffer Size

There is no limitation concerning the minimum and the maximum size of the resident deck buffer.

13.1.3 Temporary Deck Buffer Size

The minimum size of the temporary deck buffer is 1 K. There is no maximum limitation of the size.

13.1.4 Temporary Variables

The minimum memory size for allocation of temporary variables is 512 bytes. There is no maximum limitation for the memory size.

13.1.5 Service Permanent Store

The minimum memory size for allocation of a SPS is 64 bytes. There is no maximum limitation.

13.2 Reserved URLs

Following URLs are reserved for future functionality. Service developers shall never use such URLs.

- "/s/home"
- "/s/bookmarks"
- "/s/resident"
- "/s/push"
- "/s/prev"
- "/s/next"
- "/s/history"

14 HISTORY

Document history		
Release	Approved by	Comment
1.0.0	SIM Alliance TDG	First internal release
1.0.1	SIM Alliance TDG	TDG Meeting #M30 Editorial changes for publication
2.0.0	SIM Alliance TDG	Section 7.4: CR 2004-014 Section 4.3.1: CR 2004-031 Section 5.3: CR 2004-032 Section 6.4: CR 2004-034 Section 4.3.1: CR 2004-036 Some minor editorial changes

14.1 ANNEX: LIST OF CHANGE REQUESTS [informative]

CR Number	CR Identifier	Subject	Document Reference	Status / Meeting No.
2004-014	GEMPLUS-FEBRUARY-2004#3	Home page identification	S@T 1.50 V2.0.0	Accepted (email vote 9th March 2004)
2004-031	G&D 02.2004 #9	4.3.1 error on deleted temp variable	S@T 1.50 V2.0.0	Accepted (email vote 9th March 2004)
2004-032	G&D 02.2004 #10	5.3 error handling temp variable	S@T 1.50 V2.0.0	Accepted (email vote 9th March 2004)
2004-034	G&D 02.2004 #12	6.4 specify which contextual menu	S@T 1.50 V2.0.0	Accepted (email vote 9th March 2004)
2004-036	G&D 02.2004 #14	4.3.1 variable space incl. admin data	S@T 1.50 V2.0.0	Accepted (email vote 9th March 2004)