3GPP TSG-T (Terminals) Meeting #15
Jeju, Korea, 6 – 8 March 2002

Tdoc TP-020073

Source: T3

Title: Change Request to Java API specifications (TS 03.19 / 43.019 / 11.13)

Document for: Approval

This document contains several change requests as follows:

T3 Doc	Spec	CR	Rel	Cat	Subject
T3z020229	11.13	004	98	F	Testing Framework Update for the 3GPP TS 11.13 Specification
T3-020081	43.019	010	5	F	SET-UP-MENU command issued if all the items supporting help are disabled
T3-020083	43.019	011	5	B	Indication of the handler size to the applet
T3-020084	43.019	012	5	F	Clarification of the framework behavior for PoR using SMS SUBMIT
T3-020085	43.019	013	5	B	Introduction of Concatenated Short Message in SMS Point to Point
T3-020087	43.019	014	5	B	Change in the EnvelopeResponseHandler behaviour
T3-020101	43.019	015	5	C	Handler availability

3GPP ad hoc Meeting #66
Bordeaux, France, 26-28 February, 2002

Tdoc T3z020229

CR-Form-v3

CHANGE REQUEST

⌘ **11.13** **CR** **004** ⌘ rev **-** ⌘ Current version: **7.3.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM [X] ME/UE [ ] Radio Access Network [ ] Core Network [ ]

Title:	⌘	Testing Framework Update for the 3GPP TS 11.13 Specification
Source:	⌘	T3
Work item code:	⌘	SIM API
Date:	⌘	22/01/02
Category:	⌘	**F**
Release:	⌘	R98

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
REL-4	*(Release 4)*
REL-5	*(Release 5)*

Reason for change: ⌘ Add missing text and correction in the framework part, corrections in the API part.

Summary of change: ⌘ API PART: Corrections

FRAMEWORK PART:
Minimum Handler availabilty:
- Adding Test Procedure and Test Coverage for PH, PRH, EH and ERH tests
- Adding a new conformance requirement (CRRN3) for ERH

Handler Integrity:
- Adding Test Procedure and Test Coverage for PH, PRH and EH tests

Applet triggering:
- Adding Test Procedure and Test Coverage for the test of the different events
- _EMSE, _EMSH, _EMCN and _ESTC add new test suite files

Proactive command sending by the STF:
- Modification of test case 1 and 2 within _SPCO test
- Modification of test case 1 within _IGCO test

Framework Security Management
- Modification of test case 1, 2 and 3 within _INDA test
- Modification of test case 2 and 5 within _OUDA test

Envelope Response posting:
- Modification of test case 1 and 2 within _ECCN test
- Modification of test case 1 and 2 within _EMCN test

Toolkit Installation:
- Adding timers allocation, item identifier and item position parts tests.
- Modifications of all the test cases within _MLME
- Modifications of test case 3 within _NBME and deletion of some of the tet suite files
- Modification of test cases from 1 to 6 and adding from 7 to 10 test cases within

		_ACDO test, as consequence is necessary to include more test suite files. ▪ Modification of test cases from 1 to 9 and adding test cases 4, 11 and 12 within _PRLV test, as consequence is necessary to include more test suite files. File System Context: ▪ Test case 4 and 5 to test CRRN3 and CRRN5 are removed because of an inconsistent behavior in 03.19 specification. A note to explain it is included in the test procedure Editorial modifications for all the points Adding Test Area Files for Framework Part in Annex E (Annex_E_SourceCode.zip)
Consequences if not approved:	⌘	The specification would be incomplete.

Clauses affected:	⌘	2, 4.6.2, 6.1.1.2, 6.1.1.3, 6.3.1, 6.3.2, 6.3.3, 6.3.4, 6.3.5, 6.3.6, 6.3.7, 6.3.8, 6.3.9, 6.3.10, Annex C, Annex E, Annex F		
Other specs Affected:	⌘	☐ Other core specifications ☐ Test specifications ☐ O&M Specifications	⌘	
Other comments:	⌘			

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. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] (void)

[2] (void)

[3] GSM 11.11: "Digital cellular telecommunication system (Phase 2+); Specification of the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface".

[4] 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".

[5] GSM 11.17: "Subscriber Identity Module" (SIM) conformance test specification".

[6] (void)

[7] GSM 03.19 ~~version 7.4.0~~Rel-98: "Digital cellular telecommunications system (Phase 2+); Subscriber Identity Module Application Programming Interface (SIM API); SIM API for Java Card™; Stage 2".

[8] GSM 03.48 ~~version 8.4.0~~Rel-99: "Digital cellular telecommunications system (Phase 2+); Security Mechanisms for the SIM application toolkit; Stage 2"

[9] ISO/IEC 7816-3 (1997) " Identification cards - Integrated circuit(s) cards with contacts, Part 3: Electronic signals and transmission protocols"

[10] GSM 02.19 "Digital cellular telecommunications system (Phase 2+, Release 98); Subscriber Identity Module Application Programming Interface (SIM API); Service description; Stage 1"

[11] SUN Java Card Specification "Java Card 2.1 API Specification "

[12] SUN Java Card Specification "Java Card 2.1 Runtime Environment Specification"

[13] SUN Java Card Specification "Java Card 2.1 VM Architecture Specification"

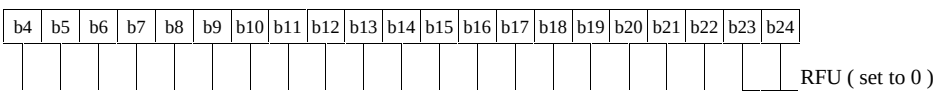
[14] ETSI TS 101 220 v3.0.0 "Numbering System for Telecommunication IC card applications"

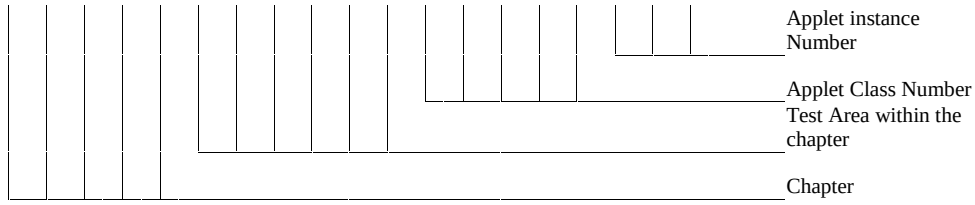
[15] GSM 11.10-1: "Digital cellular telecommunication system (Phase 2+);
Mobile Station (MS) conformance specification; Part 1: Conformance specification".

....

4.6.2 Specific Test Applet Name for Framework

Specific applet test name (bits b4-b24):





for Chapter (5 bits)

00001 Toolkit Installation Parameters

00010 Minimum Handler Availability

00011 Handler Integrity

00100 Applet Triggering

00101 Proactive Command Sending

00110 ~~Envelope Response Posting~~ Framework Security

00111 Envelope Response Posting ~~Framework Security~~

01000 File System Context

01001 Exception Handling

01010 Other parts transferred to framework from API

other are RFU

Test Area within the chapter (6 bits): values are defined in Annex F

Applet Class number (5 bits): linked to Test Area, it shall start with 1 for classes and shall be 0 for package.

Applet Instance number (3 bits) defined in the test procedure it shall start with 01 for applet instance and shall be 00 for package and class.

6.1.1.2 Method select(short fid, byte[] fci, short fciOffset, short fciLength)

Test Area Reference: API_1_SVW_ SLCTS_BSS

6.1.1.2.1 Conformance Requirements

The method with the following header shall be compliant to its definition in the API.

```
public short select(short fid,
                    byte[] fci,
                    short fciOffset,
                    short fciLength)
             throws java.lang.NullPointerException,
                    java.lang.ArrayIndexOutOfBoundsException,
                    SIMViewException
```

Normal execution

CRRN1: If the desired file is selected, the length of the FCI (File Control Information) which has been written to the array fci is returned.

CRRN2: If the length fciLength is greater than or equal to the length of the FCI structure, the whole FCI structure is copied into the array fci and the length of the FCI which has been written to the array fci is returned.

CRRN3: If the length fciLength is less than the length of the FCI structure, the first part of the FCI structure is copied into the array fci and the length of the FCI which has been written to the array fci is returned.

CRRN4: After selecting a DF/MF no EF is selected.

CRRN5: After selecting a linear fixed EF no record is selected.

CRRN6: After selecting a cyclic EF the first record which is the last updated record is selected.

CRRN7: The current files (file context) of any other applets shall not be changed. See GSM 03.19 [7] - §5.2. This will be tested during the testing of the framework.

CRRN8: The information returned by fci shall be formatted as described in GSM 11.11 [3], §9.2.1.

CRRN9: The file with a File-ID that matches fid shall be found according to the following selection rules:

1) An immediate child EF or DF of the current MF/DF can be selected,
2) A sibling DF of the current DF can be selected,
3) The current MF/DF it self can be selected,
4) The parent MF/DF of the current DF can be selected,
5) The MF can always be selected.

Parameter errors

CRRP1: If the array fci is null, an instance of NullPointerException shall be thrown.

CRRP2: If fciOffset is less than 0, an instance of ArrayIndexOutOfBoundsException shall be thrown.

CRRP3: If fciLength is less than 0, an instance of ArrayIndexOutOfBoundsException shall be thrown.

CRRP4: If fciOffset plus fciLength is greater than the length of the array fci.length, or fciOffset equals fci.length, an instance of ArrayIndexOutOfBoundsException shall be thrown.

Context errors

CRRC1: If the file with a File-ID which matches fid could not be found according to the selection rules listed in CRRN9, an instance of SIMViewException shall be thrown. The reason code shall be SIMViewException.FILE_NOT_FOUND.

CRRC2: If the method call causes a memory problem (e.g. memory access error), an instance of SIMViewException shall be thrown. The reason code shall be SIMViewException.MEMORY_PROBLEM.

CRRC3: If the method call causes an error to occur that is not expected and thus not handled, an instance of SIMViewException shall be thrown. The reason code shall be SIMViewException.INTERNAL_ERROR.

6.1.1.2.2 Test Suite Files

Additional requirements for the GSM personalisation: None

Test Script:	API_1_SVW_SLCTS_BSS_1.scr
Test Applet:	API_1_SVW_SLCTS_BSS_1.java
Load Script:	API_1_SVW_SLCTS_BSS_1.ldr
Cleanup Script:	API_1_SVW_SLCTS_BSS_1.clr

6.1.1.2.3 Test Procedure

Id	Description	API Expectation	APDU Expectation
0	**SIM Initialisation**	Responses ignored.	
1	**Select EFICCID in MF (Transparent EF)** `fid = SIMView.FID_EF_ICCID` `byte[] fci = new byte[34]` `fciOffset = 0` `fciLength = 20` `select()`	No exception shall be thrown. Shall return a value not greater than 20. <Description of fci: XX XX XX XX 2F E2 04 >	
2	**Select EF_{ICCID} in MF (Transparent EF)** `fid = SIMView.FID_EF_ICCID` `fciOffset = 0` `fciLength = 13` `select()`	No exception shall be thrown. Shall return 13. fci shall contain the first 13 bytes of the FCI structure.	
3	**Select DF_{GSM} in MF** `fid = SIMView.FID_DF_GSM` `fciOffset = 0` `fciLength = 7` `select()`	No exception shall be thrown. Shall return 7. fci shall contain the first 7 bytes of the FCI~~fci shall contain the entire FCI structure~~. *<Description of fci:* *XX XX* *XX XX* *7F 20* *02* >	
4	**Select EF_{ACM} in DF_{GSM} (CyclicEF)** `fid = SIMView.FID_EF_ACM` `fciOffset = 0` `fciLength = 20` `select()`	No exception shall be thrown. Shall return a value between 15 and 20. (Cyclic EF) fci shall contain the first 15 or more bytes of the FCI structure. fci[14] shall have the value 3 (length of record).	
5	**Select MF** `fid = SIMView.FID_MF` `fciOffset = 0` `fciLength = 34` `select()`	No exception shall be thrown. Shall return a value between 22 and 34. fci shall contain the entire FCI structure.	
6	**Select $DF_{TELECOM}$ in MF** `fid = SIMView.FID_DF_TELECOM` `fci[0] = fci[1] = '05'` `fciOffset = 2` `fciLength = 20` `select()`	No exception shall be thrown. Shall return 20. fci shall contain the first 20 bytes of the FCI structure starting at index 2. The first two bytes shall (still) have the value '05'.	
7	**Select EF_{FDN} in $DF_{TELECOM}$ (Linear FixedEF)** `fid = SIMView.FID_EF_FDN` `fciOffset = 0` `fciLength = 15` `select()`	No exception shall be thrown. Shall return 15. fci shall contain the first 15 bytes of the FCI structure. fci[14] shall have the value 28 (length of record).	
8	**fci is null** `fid = SIMView.FID_EF_FDN` `byte[] nullBuffer = null` `fciOffset = 0` `fciLength = 15` `select()`	Shall throw java.lang.NullPointerException.	
9	**fciOffset < 0** `fid = SIMView.FID_EF_FDN` `fciOffset = -1` `fciLength = 15` `select()`	Shall throw java.lang.ArrayIndexOutOfBoundsException.	
10	**fciLength < 0** `fid = SIMView.FID_EF_FDN` `fciOffset = 0`	Shall throw java.lang.ArrayIndexOutOfBoundsException.	

Id	Description	API Expectation	APDU Expectation
	fciLength = -1 select()		
11	**fciOffset + fciLength > fci.length** fid = SIMView.FID_EF_FDN fciOffset = 20 fciLength = 15 select()	Shall throw java.lang.ArrayIndexOutOfBoundsException.	
12	**fciOffset >= fci.length** fid = SIMView.FID_EF_FDN fciOffset = 34 fciLength = 1 select()	Shall throw java.lang.ArrayIndexOutOfBoundsException	
13	**Selection possibilities** 1 - fid = SIMView.FID_MF fciOffset = 0 fciLength = 15 select() 2 – fid = SIMView.FID_DF_TELECOM select() 3 - fid = SIMView.FID_DF_GRAPHICS select() 4 - fid = SIMView.FID_DF_TELECOM select() 5 - fid = SIMView.FID_DF_GRAPHICS select() 6 - fid = SIMView.FID_MF select() 7 - fid = SIMView.FID_DF_GSM select() 8 - fid = SIMView.FID_DF_TELECOM select() 9 - fid = SIMView.FID_DF_TELECOM select() –	1 – No exception shall be thrown. 2 – No exception shall be thrown. 3 – No exception shall be thrown. 4 – No exception shall be thrown. 5 – No exception shall be thrown. 6 – No exception shall be thrown. 7 – No exception shall be thrown. 8 – No exception shall be thrown. 9 – No exception shall be thrown.	
14	**EF not selected after MF/DF selection** 1 - fid = SIMView.FID_MF select() fid = SIMView.FID_EF_ICCID select() 2 - fid = SIMView.FID_MF select() readBinary()	1 - No exception shall be thrown. 2 - Shall throw sim.access.SIMViewException with reason code NO_EF_SELECTED.	
15	**No selection of non-reachable file** 1 - fid = SIMView.FID_MF select() 2 - fid = SIMView.FID_EF_ACM select()	1 – No exception shall be thrown. 2 – Shall throw sim.access.SIMViewException with reason code FILE_NOT_FOUND.	
16	**No record is selected after selecting linear fixed EF** 1 - fid = SIMView.FID_MF select() 2 - fid = FID_DF_SIMTEST select() 3 - fid = FID_EF_LARU select() 4 – recNumber = 0 mode = REC_ACC_MODE_ABSOLUTE_CURRENT readRecord()	1 – No exception shall be thrown. 2 – No exception shall be thrown. 3 – No exception shall be thrown. 4 – Shall throw sim.access.SIMViewException with reason code RECORD_NUMBER_NOT_AVAILABLE.	
17	**Record pointer in selected cyclic EF** 1 – fid = SIMView.FID_MF select() 2 - fid = FID_DF_SIMTEST select() 3 - fid = FID_EF_CARU select() 4 – byte[] data1 = { 1,2,3 } mode = REC_ACC_MODE_PREVIOUS updateRecord(data1) 5 - fid = ~~SIMView.~~FID_EF_~~ACM~~CARU select() readRecord(data2) compare data1 to data2	1 - No exception shall be thrown. 2 - No exception shall be thrown. 3 - No exception shall be thrown. 4 - No exception shall be thrown. 5 - The contents of data1 and data2 shall be identical.	

6.1.1.2.4 Test Coverage

CRR Number	Test Case Number
N1	1-7
N2	3, 5
N3	1, 2, 4, 6, 7
N4	14
N5	16
N6	17
N8	1, 3
N9	1-7, 13
P1	8
P2	9
P3	10
P4	11, 12
C1	15
C2, C3	Not Tested

6.1.1.3 Method select (short fid)

Test Area Reference: API_1_SVW_SLCTS

6.1.1.3.1 Conformance Requirements

The method with the following header shall be compliant to its definition in the API.

```
public void select(short fid)
            throws SIMViewException
```

Normal execution

CRRN1: If the desired file is selected, no exception is thrown.

CRRN2: After selecting a DF/MF no EF is selected.

CRRN3: After selecting a linear fixed EF no record is selected.

CRRN4: After selecting a cyclic EF the first record which is the last updated record is selected.

CRRN5: The current files (file context) of any other applets shall not be changed [03.19 - §5.2]. This will be tested during the testing of the framework.

CRRN6: The file with a File-ID that matches fid shall be found according to the following selection rules:

1) An immediate child EF or DF of the current MF/DF can be selected,
2) A sibling DF of the current DF can be selected,
3) The current MF/DF it self can be selected,
4) The parent MF/DF of the current DF can be selected,
5) The MF can always be selected.

Parameter errors

No requirements.

Context errors

CRRC1: If the file with a File-ID which matches fid could not be found according to the selection rules listed in CCRN6, an instance of SIMViewException shall be thrown. The reason code shall be SIMViewException.FILE_NOT_FOUND.

CRRC2: If the method call causes a memory problem (e.g. memory access error), an instance of SIMViewException shall be thrown. The reason code shall be SIMViewException.MEMORY_PROBLEM.

CRRC3: If the method call causes an error to occur that is not expected and thus not handled, an instance of SIMViewException shall be thrown. The reason code shall be SIMViewException.INTERNAL_ERROR.

6.1.1.3.2 Test Suite Files

Additional requirements for the GSM personalisation: None

Test Script:	API_1_SVW_SLCTS_1.scr
Test Applet:	API_1_SVW_SLCTS_1.java
Load Script:	API_1_SVW_SLCTS_1.ldr
Cleanup Script:	API_1_SVW_SLCTS_1.clr

6.1.1.3.3 Test Procedure

Id	Description	API Expectation	APDU Expectation
0	**SIM Initialisation**	Responses ignored.	
1	**Select EF_{ICCID} in MF (Transparent EF)** fid = SIMView.FID_EF_ICCID select()	No exception shall be thrown.	
2	**EF not selected after MF/DF selection** 1 - fid = SIMView.FID_MF select() fid = SIMView.FID_EF_ICCID select() 2 - fid = SIMView.FID_MF select() readBinary()	1 - No exception shall be thrown. 2 - Shall throw sim.access.SIMViewException with reason code NO_EF_SELECTED.	
3	**No record is selected after selecting linear fixed EF** 1 - fid = SIMView.FID_MF select() 2 - fid = FID_DF_SIMTEST select() 3 - fid =FID_EF_LARU select() 4 – recNumber = 0 mode = REC_ACC_MODE_ABSOLUTE_CURRENT readRecord()	1 – No exception shall be thrown. 2 – No exception shall be thrown. 3 – No exception shall be thrown. 4 – Shall throw sim.access.SIMViewException with reason code RECORD_NUMBER_NOT_AVAILABLE.	
4	**Record pointer in selected cyclic EF** 1 – fid = SIMView.FID_MF select() 2 - fid = ~~SIMView.~~FID_DF_~~GSM~~SIMTEST select() 3 - fid = ~~SIMView.~~FID_EF_~~ACM~~CARU select() 4 – byte[] data1 = { 1,2,3 } updateRecord(data1) 5 - fid = ~~SIMView.~~FID_EF_~~ACM~~CARU select() readRecord(data2) compare data1 to data2	1 - No exception shall be thrown. 2 - No exception shall be thrown. 3 - No exception shall be thrown. 4 - No exception shall be thrown. 5 - The contents of data1 and data2 shall be identical.	
5	**Selection possibilities** 1 - fid = SIMView.FID_MF select() 2 – fid = SIMView.FID_DF_TELECOM select() 3 - fid = SIMView.FID_DF_GRAPHICS select() 4 - fid = SIMView.FID_DF_TELECOM select() 5 - fid = SIMView.FID_DF_GRAPHICS select() 6 - fid = SIMView.FID_MF select() 7 - fid = SIMView.FID_DF_GSM select() 8 - fid = SIMView.FID_DF_TELECOM select() 9 - fid = SIMView.FID_DF_TELECOM select()	1 – No exception shall be thrown. 2 – No exception shall be thrown. 3 – No exception shall be thrown. 4 – No exception shall be thrown. 5 – No exception shall be thrown. 6 – No exception shall be thrown. 7 – No exception shall be thrown. 8 – No exception shall be thrown. 9 – No exception shall be thrown.	
6	**No selection of unreachable file** 1 - fid = SIMView.FID_MF select() 2 - fid = SIMView.FID_EF_ACM select()	1 – No exception shall be thrown. 2 – Shall throw sim.access.SIMViewException with reason code FILE_NOT_FOUND.	

6.1.1.3.4 Test Coverage

CRR Number	Test Case Number
N1	1
N2	2
N3	3
N4	4
N6	5
C1	6

CRR Number	Test Case Number
C2, C3	Not Tested

...

6.3 SIM Toolkit Framework

6.3.1 Minimum Handler Availability

This test area tests the rules that define the minimum requirements for the availability of the system handlers.

6.3.1.1 ProactiveHandler

Test Area Reference: FWK_MHA_ PAHD

6.3.1.1.1 Conformance Requirement

Normal Execution

CRRN1: If a proactive session is not ongoing the ProactiveHandler is available from the invocation to the termination of the processToolkit method for the following events:

EVENT_FORMATTED_SMS_PP_ENV

EVENT_UNFORMATTED_SMS_PP_ENV

EVENT_UNFORMATTED_SMS_CB

EVENT_MENU_SELECTION

EVENT_MENU_SELECTION_HELP_REQUEST

EVENT_TIMER_EXPIRATION

EVENT_EVENT_DOWNLOAD_MT_CALL

EVENT_EVENT_DOWNLOAD_CALL_CONNECTED

EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED

EVENT_EVENT_DOWNLOAD_LOCATION_STATUS

EVENT_EVENT_DOWNLOAD_USER_ACTIVITY

EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE

EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS

EVENT_UNRECOGNIZED_ENVELOPE

EVENT_STATUS_COMMAND

~~EVENT_FORMATTED_SMS_PP_UPD~~

~~EVENT_UNFORMATTED_SMS_PP_UPD~~

EVENT_CALL_CONTROL

EVENT_SMS_MO_CONTROL

EVENT_PROFILE_DOWNLOAD

6.3.1.1.2 Test Suite Files

Test Script: FWK_MHA_ PAHD_1.scr

Test Applet: FWK_MHA_ PAHD_1.java

FWK_MHA_PAHD_2.java

Load Script: FWK_MHA_ PAHD_1.ldr

Cleanup Script: FWK_MHA_ PAHD_1.clr

Parameter File: FWK_MHA_ PAHD_1.par

6.3.1.1.3 Test Procedure

Id	Description	API /Framework_Expectation	APDU Expectation
1	**Applets registration to all events and Proactive Handler availability with EVENT_PROFILE_DOWNLOAD** Applet1 is registered to all events defined in [7]. Using the methods initMenuEntry () for EVENT_MENU_SELECTION, requestPollInterval () for EVENT_STATUS_COMMAND, allocateTimer () for EVENT_TIMER_EXPIRATION and setEventList () for the rest of the events. Applet2 is registered to all events defined in [7], except EVENT_CALL_CONTROL_BY_SIM and EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM. Using the methods initMenuEntry () for EVENT_MENU_SELECTION, requestPollInterval () for EVENT_STATUS_COMMAND, allocateTimer () for EVENT_TIMER_EXPIRATION and setEventList () for the rest of the events. The priority of applet1 is higher than priority of applet2 1-Terminal Profile command is sent to SIM without the facility of SET_EVENT_LIST, POLL_INTERVAL,SET UP IDLE MODE TEXT and SET UP MENU. 2-Applet1 gets the Proactive Handler Applet1 is deregistered to EVENT_PROFILE_DOWNLOAD 3-Applet2 gets the Proactive Handler Applet2 is deregistered to EVENT_PROFILE_DOWNLOAD	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown	
2	**Proactive Handler availability with EVENT_MENU_SELECTION_HELP_REQUEST** Perform SIM initialization with all the facilities supported 1-Envelope menu selection with help request is sent to the SIM 2-Applet1 gets the Proactive Handler 3-Envelope menu selection with help request is sent to the SIM 4-Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown 3-Applet2 is triggered 4- No exception is thrown	

Id	Description	API /Framework Expectation	APDU Expectation

Id	Description	API /Framework_Expectation	APDU Expectation
3	**Proactive Handler availability with EVENT_MENU_SELECTION** 1-Envelope menu selection is sent to the SIM 2-Applet1 gets the Proactive Handler 3-Envelope menu selection is sent to the SIM 4-Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown. 3- Applet2 is triggered 4-No exception is thrown.	
4	**Proactive Handler availability with EVENT_FORMATTED_SMS_PP_ENV** 1-Envelope dataDownLoad formatted is sent to the SIM 2-Applet1 gets the Proactive Handler 3-Envelope dataDownLoad formatted is sent to the SIM 4-Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown. 3- Applet2 is triggered 4-No exception is thrown.	
5	**Proactive Handler availability with EVENT_UNFORMATTED_SMS_PP_ENV** 1-Envelope dataDownLoad unformatted is sent to the SIM 2-Applet1 gets the Proactive Handler 3-Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown.	
6	**Proactive Handler availability with EVENT_UNFORMATTED_CELL BROADCAST** 1-Envelope cell broadcast unformatted is sent to the SIM 2-Applet1 gets the Proactive Handler 3-Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown Applet2 is triggered 3-No exception is thrown	
7	**Proactive Handler availability with EVENT_TIMER_EXPIRATION** 1-Timer Id =1 Envelope Timer Expiration is sent to the SIM 2-Applet1 gets the Proactive Handler 3-Timer id=2 Envelope Timer Expiration is sent to the SIM 4- Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown. 3- Applet2 is triggered 4-No exception is thrown	
8	**Proactive Handler availability with**		

Id	Description	API /Framework_Expectation	APDU Expectation
	EVENT_CALL_CONTROL_BY_SIM 1-Envelope call control by SIM is sent to the SIM 2-Applet1 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown.	
9	**Proactive Handler availability with EVENT_MO_SHORT_MESSAGE_CONTROL** 1-Envelope mo short message control by SIM is sent to the SIM 2-Applet1 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown	
10	**Proactive Handler availability with EVENT_EVENT_DOWNLOAD_MT_CALL** 1-Envelope event download mt call is sent to the SIM 2-Applet1 gets the Proactive Handler 3-Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown	
11	**Proactive Handler availability with EVENT_EVENT_DOWNLOAD_CALL_CONNECTED** 1-Envelope event download call connected is sent to the SIM 2-Applet1 gets the Proactive Handler 3-Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown	
12	**Proactive Handler availability with EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED** 1-Envelope event download call disconnected is sent to the SIM 2-Applet1 gets the Proactive Handler 3-Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown.	
13	**Applets triggering with EVENT_EVENT_LOCATION_STATUS** 1-Envelope event download location status is sent to the SIM 2-Applet1 gets the Proactive Handler 3-Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown	
14	**Proactive Handler availability with**		

Id	Description	API /Framework_Expectation	APDU Expectation
	EVENT_EVENT_DOWNLOAD_USER_ACTIVITY 1-Envelope event download user activity is sent to SIM 2-Applet1 gets the Proactive Handler 3-Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown Applet2 is triggered 3-No exception is thrown	
15	**Proactive Handler availability with EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE** 1-Envelope event download idle screen available is sent to the SIM 2-Applet1 gets the Proactive Handler 3- Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown	
16	**Proactive Handler availability with EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS** 1-Envelope event download card reader status is sent to the SIM 2-Applet1 gets the Proactive Handler 3-Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown	
17	**Proactive Handler availability with EVENT_STATUS_COMMAND** 1-Status command is sent to the SIM 2-Applet1 gets the Proactive Handler 3- Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown.	
18	**Proactive Handler availability with UNRECOGNIZED_ENVELOPE** 1-An unrecognized Envelope (BER TLV Tag unrecognized) is sent to the SIM 2-Applet1 gets the Proactive Handler 3-Applet2 gets the Proactive Handler	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown	

6.3.1.1.4 Test Coverage

CRR Number	Test Case Number
CRRN1	4
CRRN1	5
CRRN1	6
CRRN1	2
CRRN1	3
CRRN1	8
CRRN1	9
CRRN1	7
CRRN1	10
CRRN1	11
CRRN1	12
CRRN1	13
CRRN1	14
CRRN1	15
CRRN1	16
CRRN1	18
CRRN1	17
CRRN1	1

6.3.1.2 ProactiveResponseHandler

Test Area Reference: FWK_MHA_ PRHD

6.3.1.2.1 Conformance Requirement

Normal Execution

CRRN1: The ProactiveResponseHandler is available after the first call to the ProactiveHandler.send method to the termination of the processToolkit method for the following events:

EVENT_FORMATTED_SMS_PP_ENV

EVENT_UNFORMATTED_SMS_PP_ENV

EVENT_UNFORMATTED_SMS_CB

EVENT_MENU_SELECTION

EVENT_MENU_SELECTION_HELP_REQUEST

EVENT_TIMER_EXPIRATION

EVENT_EVENT_DOWNLOAD_MT_CALL

EVENT_EVENT_DOWNLOAD_CALL_CONNECTED

EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED

EVENT_EVENT_DOWNLOAD_LOCATION_STATUS

EVENT_EVENT_DOWNLOAD_USER_ACTIVITY

EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE

EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS

EVENT_UNRECOGNIZED_ENVELOPE

EVENT_STATUS_COMMAND

~~EVENT_FORMATTED_SMS_PP_UPD~~

~~EVENT_UNFORMATTED_SMS_PP_UPD~~

EVENT_CALL_CONTROL

EVENT_SMS_MO_CONTROL

EVENT_PROFILE_DOWNLOAD

6.3.1.2.2 Test Suite Files

Test Script:	FWK_MHA_ PRHD_1.scr
Test Applet:	FWK_MHA_ PRHD_1.java
	FWK_MHA_PRHD_2.java
Load Script:	FWK_MHA_ PRHD_1.ldr
Cleanup Script:	FWK_MHA_ PRHD_1.clr
Parameter File:	FWK_MHA_ PRHD_1.par

6.3.1.2.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applets registration to all events and Proactive Response Handler availability with EVENT_PROFILE_DOWNLOAD** 1- Applet1 is registered to all events defined in [7], applet2 is registered to all events defined in [7] except EVENT_CALL_CONTROL_BY_SIM and EVENT_MO_SMS_CONTROL_BY_SIM. Using the methods initMenuEntry() for EVENT_MENU_SELECTION, requestPollInterval() for EVENT_STATUS_COMMAND, allocateTimer() for EVENT_TIMER_EXPIRATION and setEventList() for the rest of the events. 1-Terminal Profile command is sent to the SIM without the facility of SET_EVENT_LIST and POLL_INTERVAL, ,SET UP IDLE MODE TEXT and SET UP MENU. Applet1 builds a proactive command DISPLAY TEXT. 2- ProactiveHandler.send() method is called 3- ProactiveResponseHandler.getTheHandler() method is called Applet1 is deregistered to EVENT_PROFILE_DOWNLOAD Applet1 execution is finished Applet2 builds a proactive command DISPLAY TEXT. 4- ProactiveHandler.send() method is called	1-Applet1 is triggered No exception is thrown 3- No exception is thrown Applet2 is triggered	2- :The proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE 4- :The proactive command DISPLAY TEXT is fetched

Id	Description	API/Framework Expectation	APDU Expectation
		5- No exception is thrown	TERMINAL RESPONSE
	5- ProactiveResponseHandler.getTheHandler() method is called Applet1 is deregistered to EVENT_PROFILE_DOWNLOAD		
2	**Proactive Response Handler availability with EVENT_MENU_SELECTION_HELP_REQUEST** Perform SIM initialization with all the facilities supported 1-Envelope menu selection with help request is sent to the SIM	1- Applet1 is triggered	
	Applet1 builds a proactive command DISPLAY TEXT 2- ProactiveHandler.send() method is called		2- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE
	3- ProactiveResponseHandler.getTheHandler() method is called Applet1 execution is finished Envelope menu selection with help request is sent to the SIM	3- No exception is thrown Applet2 is triggered	
	Applet2 builds a proactive command DISPLAY TEXT 4- ProactiveHandler.send() method is called		4- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE
	5- ProactiveResponseHandler.getTheHandler() method is called	5- No exception is thrown	
3	**Proactive Response Handler availability with EVENT_MENU_SELECTION** 1-Envelope menu selection is sent to the SIM	1- Applet1 is triggered	
	Applet1 builds a proactive command DISPLAY TEXT 2- ProactiveHandler.send() method is called		2- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE
	3-ProactiveResponseHandler.getTheHandler() method is called Applet1 execution is finished 4-Envelope menu selection is sent to the SIM	3- No exception is thrown 4-Applet2 is triggered	
	Applet2 builds a proactive command DISPLAY TEXT		

Id	Description	API/Framework Expectation	APDU Expectation
	TEXT 5- ProactiveHandler.send() method is called 6-ProactiveResponseHandler.getTheHandler() method is called	6- No exception is thrown	5- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE
4	**Proactive Response Handler availability with EVENT_FORMATTED_SMS_PP_ENV** 1-Envelope dataDownLoad formatted is sent to the SIM Appet builds a proactive command DISPLAY TEXT 2-ProactiveHandler.send() method is called 3-ProactiveResponseHandler.getTheHandler() method is called Applet1 execution is finished 4-Envelope dataDownLoad formatted is sent to the SIM Applet2 builds a proactive command DISPLAY TEXT 5-ProactiveHandler.send() method is called 6-ProactiveResponseHandler.getTheHandler() method is called	1-Applet1 is triggered 3- No exception is thrown 4-Applet2 is triggered 6- No exception is thrown	2- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE 5- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE
5	**Proactive Response Handler availability with EVENT_UNFORMATTED_SMS_PP_ENV** 1-Envelope dataDownLoad unformatted is sent to the SIM Applet1 builds a proactive command DISPLAY TEXT 2- ProactiveHandler.send() method is called 3-ProactiveResponseHandler.getTheHandler() method is called Applet1 execution is finished Applet2 builds a proactive command DISPLAY TEXT 4- ProactiveHandler.send() method is called	1- Applet1 is triggered 3- No exception is thrown Applet2 is triggered	2- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE 4- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE

Id	Description	API/Framework Expectation	APDU Expectation
	5- ProactiveResponseHandler.getTheHandler() method is called	5- No exception is thrown	
6	**Proactive Response Handler availability with EVENT_UNFORMATTED_SMS_CB** 1-Envelope call broadcast unformatted is sent to the SIM Applet1 builds a proactive command DISPLAY TEXT 2- ProactiveHandler.send() method is called 3-ProactiveResponseHandler.getTheHandler() method is called. Applet1 execution is finished Applet2 builds a proactive command DISPLAY TEXT 4- ProactiveHandler.send() method is called 5- ProactiveResponseHandler.getTheHandler() method is called	1- Applet1 is triggered 3- No exception is thrown Applet2 is triggered 5- No exception is thrown	2- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE 4- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE
7	**Proactive Response Handler availability with EVENT_TIMER_EXPIRATION** Timer id=1 1-Envelope Timer Expiration is sent to the SIM Applet builds a proactive command DISPLAY TEXT 2-ProactiveHandler.send() method is called 3-ProactiveResponseHandler.getTheHandler() method is called Applet1 execution is finished Timer id=2 Envelope Timer Expiration is sent to the SIM Applet builds a proactive command DISPLAY TEXT 4-ProactiveHandler.send() method is called 5-ProactiveResponseHandler.getTheHandler() method is called	1-Applet1 is triggered 3- No exception is thrown Applet2 is triggered 4- No exception is thrown	2- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE 5- A proactive command

Id	Description	API/Framework Expectation	APDU Expectation
	Applet2 execution is finished		DISPLAY TEXT is fetched TERMINAL RESPONSE
8	**Proactive Response Handler availability with EVENT_CALL_CONTROL_BY_SIM** 1-Envelope call control by sim is sent to the SIM Applet builds a proactive command DISPLAY TEXT 2-ProactiveHandler.send() method is called 3-ProactiveResponseHandler.getTheHandler() method is called	1- Applet1 is triggered 3-No exception is thrown	2- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE
9	**Proactive Response Handler availability with MO_SHORT_MESSAGE_CONTROL_BY_SIM** 1-Envelope mo short message control by sim is sent to the SIM Applet builds a proactive command DISPLAY TEXT 2-ProactiveHandler.send() method is called 3-ProactiveResponseHandler.getTheHandler() method is called Applet1 execution is finished	1- Applet1 is triggered 3-No exception is thrown	2- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE

Id	Description	API/Framework Expectation	APDU Expectation
10	**Proactive Response Handler availability with EVENT_EVENT_DOWNLOAD_MT_CALL** 1-Envelope event download mt call is sent to the SIM Applet1 builds a proactive command DISPLAY TEXT 2-ProactiveHandler.send() method is called 3-ProactiveResponseHandler.getTheHandler() method is called. Applet1 execution is finished Applet2 builds a proactive command DISPLAY TEXT 4- ProactiveHandler.send() method is called 5- ProactiveResponseHandler.getTheHandler() method is called	1- Applet1 is triggered 3- No exception is thrown Applet2 is triggered 5- No exception is thrown	2- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE 4- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE
11	**Proactive Response Handler availability with EVENT_EVENT_DOWNLOAD_CALL_CONNECTED** 1-Envelope event download call connected is sent to the SIM Applet1 builds a proactive command DISPLAY TEXT 2-ProactiveHandler.send() method is called 3-ProactiveResponseHandler.getTheHandler() method is called Applet1 execution is finished Applet builds a proactive command DISPLAY TEXT 4- ProactiveHandler.send() method is called 5- ProactiveResponseHandler.getTheHandler() method is called	1- Applet1 is triggered 3- No exception is thrown Applet2 is triggered 5- No exception is thrown	2- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE 4- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE

Id	Description	API/Framework Expectation	APDU Expectation
12	**Proactive Response Handler availability with EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED** 1-Envelope event download call disconnected is sent to the SIM Applet1 builds a proactive command DISPLAY TEXT 2-ProactiveHandler.send() method is called 3-ProactiveResponseHandler.getTheHandler() method is called Applet1 execution is finished Applet2 builds a proactive command DISPLAY TEXT 4- ProactiveHandler.send() method is called 5- ProactiveResponseHandler.getTheHandler() method is called	1- Applet1 is triggered 2- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE 3- No exception is thrown Applet2 is triggered 4- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE 5- No exception is thrown
13	**Proactive Response Handler availability with EVENT_EVENT_DOWNLOAD_CALL_CONNECTED** 1-Envelope event download location status is sent to the SIM Applet1 builds a proactive command DISPLAY TEXT 2-ProactiveHandler.send() method is called 3-ProactiveResponseHandler.getTheHandler() method is called Applet1 execution is finished Applet2 builds a proactive command DISPLAY TEXT 4- ProactiveHandler.send() method is called 5-	1- Applet1 is triggered 2-A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE 3-No exception is thrown Applet2 is triggered 4-A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE 5-No exception is thrown

Id	Description	API/Framework Expectation	APDU Expectation
	ProactiveResponseHandler.getTheHandler() method is called		
14	**Proactive Response Handler availability with EVENT_EVENT_DOWNLOAD_USER_ACTIVITY** 1-Envelope event download user activity is sent to the SIM Applet1 builds a proactive command DISPLAY TEXT 2-ProactiveHandler.send() method is called 3-ProactiveResponseHandler.getTheHandler() method is called Applet1 execution is finished Applet2 builds a proactive command DISPLAY TEXT 4- ProactiveHandler.send() method is called 5-ProactiveResponseHandler.getTheHandler() method is called	1- Applet1 is triggered 3-No exception is thrown Applet2 is triggered 5-No exception is thrown	2-A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE 4-A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE
15	**Proactive Response Handler availability with EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE** 1-Envelope event download idle screen available is sent to the SIM Applet1 builds a proactive command DISPLAY TEXT 2-ProactiveHandler.send() method is called 3-ProactiveResponseHandler.getTheHandler() method is called Applet1 execution is finished Applet2 builds a proactive command DISPLAY TEXT 4- ProactiveHandler.send() method is called 5-ProactiveResponseHandler.getTheHandler() method is called	1- Applet1 is triggered 3- No exception is thrown Applet2 is triggered 5- No exception is thrown	2- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE 4- A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE

Id	Description	API/Framework Expectation	APDU Expectation
16	**Proactive Response Handler availability with EVENT_EVENT_DOWNLOAD_CALL_CONNECTED**		
	1-Envelope event download card reader status is sent to the SIM	1- Applet1 is triggered	
	Applet1 builds a proactive command DISPLAY TEXT 2-ProactiveHandler.send() method is called		2-A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE
	3-ProactiveResponseHandler.getTheHandler() method is called Applet1 execution is finished	3-No exception is thrown Applet2 is triggered	
	Applet2 builds a proactive command DISPLAY TEXT 4- ProactiveHandler.send() method is called		4-A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE
	5-ProactiveResponseHandler.getTheHandler() method is called	5-No exception is thrown	
17	**Proactive Response Handler availability with EVENT_STATUS_COMMAND**		
	1-Status command is sent to the SIM	1- Applet1 is triggered	
	Applet1 builds a proactive command DISPLAY TEXT 2-ProactiveHandler.send() method is called		2-A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE
	3-ProactiveResponseHandler.getTheHandler() method is called Applet1 execution is finished	3-No exception is thrown Applet2 is triggered	
	Applet2 builds a proactive command DISPLAY TEXT 4- ProactiveHandler.send() method is called		4-A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE
	5-ProactiveResponseHandler.getTheHandler() method is called	5-No exception is thrown	

Id	Description	API/Framework Expectation	APDU Expectation
18	**Proactive Response Handler availability with UNRECOGNIZED_ENVELOPE** 1-An unrecognized Envelope is sent to the SIM Applet1 builds a proactive command DISPLAY TEXT 2-ProactiveHandler.send() method is called 3-ProactiveResponseHandler.getTheHandler() method is called Applet1 execution is finished Applet2 builds a proactive command DISPLAY TEXT 4- ProactiveHandler.send() method is called 5- ProactiveResponseHandler.getTheHandler() method is called	1- Applet1 is triggered 3- No exception is thrown Applet2 is triggered 5- No exception is thrown	2-A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE 4-A proactive command DISPLAY TEXT is fetched TERMINAL RESPONSE

6.3.1.2.4 Test Coverage

CR Number	Test Case Number
CRRN1	4
CRRN1	5
CRRN1	6
CRRN1	3
CRRN1	2
CRRN1	8
CRRN1	9
CRRN1	7
CRRN1	10
CRRN1	11
CRRN1	12
CRRN1	13
CRRN1	14
CRRN1	15
CRRN1	16
CRRN1	18
CRRN1	17
CRRN1	1

6.3.1.3 EnvelopeHandler

Test Area Reference: FWK_MHA_ ENHD

6.3.1.3.1 Conformance Requirement

Normal Execution

CRRN1: The EnvelopeHandler and its content are available for all toolkit applets triggered from the invocation to the termination of their processToolkit method for the following events:.

EVENT_FORMATTED_SMS_PP_ENV

EVENT_UNFORMATTED_SMS_PP_ENV

EVENT_UNFORMATTED_SMS_CB

EVENT_MENU_SELECTION

EVENT_MENU_SELECTION_HELP_REQUEST

EVENT_TIMER_EXPIRATION

EVENT_EVENT_DOWNLOAD_MT_CALL

EVENT_EVENT_DOWNLOAD_CALL_CONNECTED

EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED

EVENT_EVENT_DOWNLOAD_LOCATION_STATUS

EVENT_EVENT_DOWNLOAD_USER_ACTIVITY

EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE

EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS

EVENT_UNRECOGNIZED_ENVELOPE

~~EVENT_FORMATTED_SMS_PP_UPD~~

~~EVENT_UNFORMATTED_SMS_PP_UPD~~

EVENT_CALL_CONTROL

EVENT_SMS_MO_CONTROL

Context Errors

CRRC1: The EnvelopeHandler and its content are not available for any toolkit applet triggered from the invocation to the termination of their processToolkit method for the following events:

EVENT_STATUS_COMMAND

EVENT_PROFILE_DOWNLOAD

6.3.1.3.2 Test Suite Files

Test Script:	FWK_MHA_ ENHD_1.scr
Test Applet:	FWK_MHA_ ENHD_1.java
	FWK_MHA_ENHD_2.java
Load Script:	FWK_MHA_ ENHD_1.ldr
Cleanup Script:	FWK_MHA_ ENHD_1.clr
Parameter File:	FWK_MHA_ ENHD_1.par

6.3.1.3.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet1 and Applet2 registration and Envelope Handler availability with EVENT_PROFILE_DOWNLOAD** 1- Applet1 is registered to all events defined [7]. Using the methods initMenuEntry() for EVENT_MENU_SELECTION, requestPollInterval() for EVENT_STATUS_COMMAND, allocateTimer() for EVENT_TIMER_EXPIRATION and setEventList() for the rest of the events. Applet2 is registered to all events defined [7] except EVENT_CALL_CONTROL_BY_SIM and EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM. Using the methods initMenuEntry() for EVENT_MENU_SELECTION, requestPollInterval() for EVENT_STATUS_COMMAND, allocateTimer for EVENT_TIMER_EXPIRATION and setEventList for the rest of the events. 2-Terminal Profile command is sent to SIM without the facility of SET_EVENT_LIST ,SETUP_IDLE_MODE_TEXT ,POLL_INTERVAL and SETUP MENU 3-EnvelopeHandler.getTheHandler() method is called by Applet1 Applet1 is deregistered to EVENT_PROFILE_DOWNLOAD 4-EnvelopeHandler.getTheHandler() method is called by Applet2 Applet2 is deregistered to EVENT_PROFILE_DOWNLOAD	1- No exception is thrown 2- Applet1 is triggered 3- A Toolkit exception HANDLER_NOT_AVAILABLE is thrown Applet2 is triggered 4- A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	
2	**Envelope Handler availability with EVENT_MENU_SELECTION_HELP_REQUEST** . Perform SIM initialization with all the facilities supported Envelope menu selection with help request is sent to the SIM Applet1 is triggered 1-EnvelopeHandler.getTheHandler() method is called by Applet1 2-Envelope menu selection with help request is sent to the SIM 3-EnvelopeHandler.getTheHandler() method is called by Applet2	1-No exception is thrown. Applet1 finalizes. 2- Applet2 is triggered 3-No exception is thrown.	
3	**Envelope Handler availability with EVENT_MENU_SELECTION** 1-Envelope menu selection is sent to the		

Id	Description	API/Framework Expectation	APDU Expectation
	SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1 Applet1 finalizes. 3-Envelope menu selection is sent to the SIM 4-EnvelopeHandler.getTheHandler() method is called by Applet2	1- Applet1 is triggered 2-No exception is thrown. 3- Applet2 is triggered 4-No exception is thrown.	

Id	Description	API/Framework Expectation	APDU Expectation
4	**Envelope Handler availability with EVENT_FORMATTED_SMS_PP_ENV** 1-A EVENT_FORMATTED_SMS_PP_ENV envelope is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1 Applet1 finalizes. 3-A EVENT_FORMATTED_SMS_PP_ENV envelope is sent to the SIM 4-EnvelopeHandler.getTheHandler() method is called by Applet2	1- Applet1 is triggered 2-No exception is thrown. 3- Applet2 is triggered 4-No exception is thrown.	
5	**Envelope Handler availability with EVENT_UNFORMATTED_SMS_PP_ENV** 1-An unformatted sms pp envelope is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1 Applet1 finalizes. 3-EnvelopeHandler.getTheHandler() method is called by Applet2	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown.	
6	**Envelope Handler availability with EVENT_UNFORMATTED_CB** 1-Envelope cell broadcast unformatted is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1 Applet1 finalizes. 3-EnvelopeHandler.getTheHandler() method is called by Applet2	1- Applet1 is triggered 2-No exception is thrown Applet2 is triggered 3-No exception is thrown	
7	**Envelope Handler availability with EVENT_TIMER_EXPIRATION** Timer id=1 1-Envelope Timer Expiration is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1 Applet1 finalizes. Timer id=2 3-Envelope Timer Expiration is sent to the SIM 4-EnvelopeHandler.getTheHandler() method is called by Applet2 Applet2 finalizes.	1- Applet1 is triggered 2-No exception is thrown. 3- Applet2 is triggered 4-No exception is thrown.	
8	**Envelope Handler availability with EVENT_CALL_CONTROL_BY_SIM**		

Id	Description	API/Framework Expectation	APDU Expectation
	1-Envelope call control by sim is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1	1- Applet1 is triggered 2-No exception is thrown.	

Id	Description	API/Framework Expectation	APDU Expectation
9	**Envelope Handler availability with EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM** 1-Envelope mo short message control by sim is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1.	1- Applet1 is triggered 2-No exception is throw	
10	**Envelope Handler availability with EVENT_EVENT_DOWNLOAD_MT_CALL** 1-Envelope event download mt call is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1 Applet1 finalizes. 3-EnvelopeHandler.getTheHandler() method is called by Applet2	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown.	
11	**Envelope Handler availability with EVENT_EVENT_DOWNLOAD_CALL_CONNECTED** 1-Envelope event download call connected is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1 Applet1 finalizes. 3-EnvelopeHandler.getTheHandler() method is called by Applet2	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown.	
12	**Envelope Handler availability with EVENT_EVENT_DOWNLOAD_CALL_DISCONECTTED** 1-Envelope event download call disconnected is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1 Applet1 finalizes. 3-EnvelopeHandler.getTheHandler() method is called by Applet2	1- Applet1 is triggered. 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown.	
13	**Envelope Handler availiability with EVENT_EVENT_DOWNLOAD_LOCATION_STATUS** 1-Envelope event download location status is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1 Applet1 finalizes. 3-EnvelopeHandler.getTheHandler() method	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered	

Id	Description	API/Framework Expectation	APDU Expectation
	is called by Applet2	3-No exception is thrown.	

Id	Description	API/Framework_Expectation	APDU Expectation
14	**Envelope Handler availiability with EVENT_EVENT_DOWNLOAD_USER_ACTIVITY** 1-Envelope event download user activity is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1 Applet1 finalizes. 3-EnvelopeHandler.getTheHandler() method is called by Applet2	1- Applet1 is triggered 2-No exception is thrown Applet2 is triggered 3-No exception is thrown	
15	**Envelope Handler availability with EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE** 1-Envelope event download idle screen available is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1 Applet1 finalizes. 3-EnvelopeHandler.getTheHandler() method is called by Applet2	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown.	
16	**Envelope Handler availiability with EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS** 1-Envelope event download card reader status is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1 Applet1 finalizes. 3-EnvelopeHandler.getTheHandler() method is called by Applet2	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown.	
17	**Envelope Handler availaibility with EVENT_STATUS_COMMAND** 1-Status command is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1 3-EnvelopeHandler.getTheHandler() method is called by Applet2	1-Applet1 is triggered 2-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown Applet2 is triggered 3-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	

Id	Description	API/Framework Expectation	APDU Expectation
18	**Envelope Handler availiability with EVENT_ UNRECOGNIZED_ENVELOPE** 1-An unrecognized Envelope is sent to the SIM 2-EnvelopeHandler.getTheHandler() method is called by Applet1 Applet1 finalizes. 3-EnvelopeHandler.getTheHandler() method is called by Applet2	1- Applet1 is triggered 2-No exception is thrown. Applet2 is triggered 3-No exception is thrown.	

6.3.1.3.4 Test Coverage

CR Number	Test Case Number
CRRN1	14,15,16,17,18,19,20,21
CRRN2	14,15,16,17,18,19,20,21
CRRC1	1,2,3,4,5,6,7,8,9,10,11,12,13

6.3.1.4 EnvelopeResponseHandler

Test Area Reference: FWK_MHA_ ERHD

6.3.1.4.1 Conformance Requirement

Normal Execution

CRRN1: The handler is available for all triggered toolkit applets from the invocation of the processToolkit method of the toolkit applet until a toolkit applet has posted an envelope response or the first invocation of the ProactiveHandler.send method for the following events:.

EVENT_FORMATTED_SMS_PP_ENV

EVENT_UNFORMATTED_SMS_PP_ENV

EVENT_CALL_CONTROL

EVENT_SMS_MO_CONTROL

EVENT_UNRECOGNIZED_ENVELOPE

CRRN2: After a call to the post method the handler is not longer available

CRRN3: After a call to the send method the handler is not longer available

Context Errors

CRRC1: The handler is not available for the following events:

EVENT_UNFORMATTED_SMS_CB

EVENT_MENU_SELECTION

EVENT_MENU_SELECTION_HELP_REQUEST

EVENT_TIMER_EXPIRATION

EVENT_EVENT_DOWNLOAD_MT_CALL

EVENT_EVENT_DOWNLOAD_CALL_CONNECTED

EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED

EVENT_EVENT_DOWNLOAD_LOCATION_STATUS

EVENT_EVENT_DOWNLOAD_USER_ACTIVITY

EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE

EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS

EVENT_STATUS_COMMAND

~~EVENT_FORMATTED_SMS_PP_UPD~~

~~EVENT_UNFORMATTED_SMS_PP_UPD~~

EVENT_PROFILE_DOWNLOAD

6.3.1.4.2 Test Suite Files

Test Script:	FWK_MHA_ ERHD_1.scr
Test Applet:	FWK_MHA_ ERHD_1.java
	FWK_MHA_ERHD_2.java
Load Script:	FWK_MHA_ ERHD_1.ldr
Cleanup Script:	FWK_MHA_ ERHD_1.clr
Parameter File:	FWK_MHA_ ERHD_1.par

6.3.1.4.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Toolkit Applet1 and Toolkit Applet2 registration and Envelope Response Handler availability with EVENT_PROFILE_DOWNLOAD** 1- The aApplet1 Toolkit 1 is registered to all events defined in [7]. Using the methods initMenuEntry() for EVENT_MENU_SELECTION, requestPollInterval() for EVENT_STATUS_COMMAND, allocateTimer() for EVENT_TIMER_EXPIRATION and setEventList() for the rest of the events. The aApplet2 Toolkit 2 is registered to EVENT_UNFORMATTED_SMS_PP_ENV and EVENT_UNRECOGNIZED_ENVELOPE. 2-Terminal Profile command is sent to SIM without the facility of SET_EVENT_LIST ,SETUP_IDLE_MODE_TEXT, SETUP_MENU and POLL_INTERVAL. Applet1 is triggered 32-EnvelopeResponseHandler.getTheHandler() method is called by Applet1 Applet1 is deregistered to EVENT_PROFILE_DOWNLOAD	1-No exception is thrown 2- Applet1 is triggered 32-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	

Id	Description	API/Framework Expectation	APDU Expectation
2	**Envelope Response Handler availiability with EVENT_MENU_SELECTION_HELP_REQUEST** Perform SIM initialization with all the facilities supported The applet1 is triggered. 1-Envelope menu selection with help request is sent to the SIM 2-EnvelopeResponseHandler.getTheHandler() method is called by Applet1	1- The aApplet1 is triggered. 2-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	
3	**Envelope Response Handler availiability with EVENT_MENU_SELECTION** 1-A envelope menu selection is sent to the SIM The Applet1 is triggered 21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1	1- The Applet1 is triggered 2-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	
4	**Envelope Response Handler availability with EVENT_UNFORMATTED_CB** 1-Envelope cell broadcast unformatted is sent to the SIM The applet1 is triggered. 21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1	1- The aApplet1 is triggered. 21-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	
5	**Envelope Response Handler availiability with EVENT_TIMER_EXPIRATION** 1-Envelope Timer Expiration is sent to the SIM The applet1 is triggered. 21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1	1- The aApplet1 is triggered. 21-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	
6	**Envelope Response Handler availiability with EVENT_EVENT_DOWNLOAD_MT_CALL** 1-Envelope event download mt call is sent to the SIM The applet1 is triggered. 21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1	1- The aApplet1 is triggered. 12-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	

Id	Description	API/Framework Expectation	APDU Expectation
7	**Envelope Response Handler availability with EVENT_EVENT_DOWNLOAD_CALL_CONNECTED** 1-Envelope event download call connected is sent to the SIM The applet1 is triggered. 21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1	1- The aApplet1 is triggered. 21-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	
8	**Envelope Response Handler availiability with EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED** 1-Envelope event download call disconnected is sent to the SIM The applet1 is triggered. 21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1	1- The aApplet1 is triggered. 21-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	
9	**Envelope Response Handler availability with EVENT_EVENT_DOWNLOAD_LOCATION_STATUS** 1-Envelope event download location status is sent to the SIM The applet1 is triggered. 21-The aApplet A obtains the Envelope Response Handler	1- The aApplet1 is triggered. 21-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	
10	**Envelope Response Handler availability with EVENT_EVENT_DOWNLOAD_USER_ACTIVITY** 1-Envelope event download user activity is sent to the SIM The applet1 is triggered. 21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1	1- The aApplet1 is triggered. 21-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	
11	**Envelope Response Handler availability with EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE** 1-Envelope event download idle screen available is sent to the SIM The applet1 is triggered. 21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1	1- The aApplet1 is triggered. 21-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	
12	**Envelope Response Handler availiability with EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS**		

Id	Description	API/Framework Expectation	APDU Expectation
	1-Envelope event download card reader status is sent to the SIM Applet1 is triggered 21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1	1- Applet1 is triggered 21-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	
13	**Envelope Response Handler availiability wiith EVENT_STATUS_COMMAND** 1-Status command is sent to the SIM Applet1 is triggered 21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1	1- Applet1 is triggered 21- A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	
14	**Envelope Response Handler availiability with EVENT_FORMATTED_SMS_PP_ENV** 1-A formatted sms pp envelope is sent to the SIM Applet1 is triggered 21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1 32-The aApplet1 builds an additional information for response packet and it calls the post method 43-The aApplet1 calls all methods of the Envelope Response Handler (including the inherited method) The Applet1 finalizes 5-A EVENT_FORMATTED_SMS_PP_ENV envelope is sent to the SIM Applet1 is triggered 64-EnvelopeResponseHandler.getTheHandler() method is called by Applet1 75-The aApplet1 builds a proactive command and it calls the send() method 86-The aApplet1 calls all methods of the Envelope Response Handler (including the inherited method)	1- Applet1 is triggered 21-No exception is thrown. 43- A Toolkit exception HANDLER_NOT_AVAILABLE is thrown for each method 5- Applet1 is triggered 64-No Exception is thrown 86-Toolkit exception HANDLER_NOT_AVAILABLE is thrown for each method	32-The response packet is sent 75-The proactive command is sent
15	**Envelope Response Handler availability with EVENT_UNFORMATTED_SMS_PP_ENV** 1-A unformatted sms pp envelope is sent to the SIM 21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1 32-The aApplet1 builds the envelope response and it calls the post() method	1- Applet1 is triggered 21-No exception is thrown.	32-The envelope response

Id	Description	API/Framework Expectation	APDU Expectation
			is sent
	43- The aApplet1 calls all methods of the Envelope Response Handler (including the inherited method)	43-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown for each method	
	The Applet1 finalizes Applet2 is triggered.	Applet2 is triggered.	
	54-EnvelopeResponseHandler.getTheHandler() method is called	54-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown.	
	Applet2 finalizes.		
	6-A unformatted sms pp envelope is sent to the SIM Applet1 is triggered	6- Applet1 is triggered	
	75-EnvelopeResponseHandler.getTheHandler() method is called.	75- No exception is thrown.	
	86-The aApplet1 builds a proactive command and it calls the send() method		86-The proactive command is fetched and the Terminal response is issued.
	97-The aApplet1 calls all methods of the Envelope Response Handler (including the inherited method)	97-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown for each method	
	The applet1 finalizes and the applet2 is triggered	The aApplet1 finalizes and the aApplet2 is triggered	
	810- EnvelopeResponseHandler.getTheHandler() method is called by Applet2	108-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown	

Id	Description	API/Framework Expectation	APDU Expectation
16	**Envelope Response Handler availability with EVENT_CALL_CONTROL_BY_SIM** 1-Envelope call control by sim is sent to the SIM 21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1 32-The aApplet1 builds the envelope response and it calls the postAsBERTLV() method 43-The aApplet1 calls all methods of the Envelope Response Handler (including the inherited method) The Applet1 finalizes 5-Envelope call control by sim is sent to the SIM 64-EnvelopeResponseHandler.getTheHandler() method is called by Applet1 75-The aApplet1 builds a proactive command and it calls the send() method 86-The aApplet1 calls all methods of the Envelope Response Handler (including the inherited method)	1- The aApplet1 is triggered 12-No exception is thrown. 43-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown for each method 5-The aApplet1 is triggered 64-No Exception is thrown 8-6-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown for each method	32-The envelope response is sent 75-The proactive command is fetched and the Terminal response is issued
17	**Envelope Response Handler availability with EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM** 1-Envelope mo short message control by sim is sent to the SIM 21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1 32-The aApplet1 builds the envelope response and it calls the postAsBERTLV() method 43-The aApplet1 calls all methods of the Envelope Response Handler (including the inherited method) The Applet1 finalizes 5-Envelope mo short message control by sim is sent to the SIM 64-EnvelopeResponseHandler.getTheHandler() method is called by Applet1 75-The aApplet1 builds a proactive command and it calls the send method 86-The aApplet1 calls all methods of the Envelope Response Handler (including the inherited method)	1- The aApplet1 is triggered 21-No exception is thrown. 43-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown for each method 5- The aApplet1 is triggered 64- No exception is thrown 86-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown for each method	32-The envelope response is sent 57-The proactive command is fetched and the Terminal Response is issued
18	**Envelope Response Handler availability with EVENT_UNRECOGNIZED_ENVELOPE** 1-An unrecognized Envelope is sent to the SIM	1- The aApplet1 is triggered	

Id	Description	API/Framework Expectation	APDU Expectation
	21-EnvelopeResponseHandler.getTheHandler() method is called by Applet1	21-No exception is thrown.	
	32-The aApplet1 builds the envelope response and it calls the postAsBERTLV() or post method		32-The envelope response is sent
	43-The aApplet1 calls all methods of Envelope Response Handler (including the inherited method)	34-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown for each method	
	The Applet1 finalizes	Applet2 is triggered.	
	Applet2 is triggered.		
	54-EnvelopeResponseHandler.getTheHandler() method is called	54-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown for each method	
	Applet2 finalizes	6-Applet1 is triggered.	
	6-An unrecognized Envelope is sent to the SIM		
	75-EnvelopeResponseHandler.getTheHandler() method is called	75-No exception is thrown.	
	86-The aApplet1 builds a proactive command and it calls the send() method		86-The proactive command is fetched and the Terminal response is issued
	97-The aApplet1 calls all methods of the Envelope Response Handler (including the inherited method)	97-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown for each method	
	The applet1 finalizes and the applet2 is triggered	The aApplet1 finalizes and the aapplet2 is triggered	
	810-EnvelopeResponseHandler.getTheHandler() method is called by Applet2	108-A Toolkit exception HANDLER_NOT_AVAILABLE is thrown for each method	

Id	Description	API/Framework Expectation	APDU Expectation
19	**The envelope response is sent when a proactive session is ongoing** 1-A formatted SMS PP envelope is sent to the SIM. Applet1 is triggered. 21-Proactive command DISPLAY TEXT is built and it calls the send() method. 3-A call control by sim envelope is sent to the SIM. Applet1 is triggered 42-EnvelopeResponseHandler.getTheHandler() method is called by Applet1 53-The aApplet1 builds the envelope response and it calls the postAsBERTLV	1- Applet1 is triggered. 3- Applet1 is triggered 42-No exception is thrown	21-91 XX 53-The envelope response is sent 9F YY GET RESPONSE Data 91 XX Fetch DISPLAY TEXT Terminal Response DISPLAY TEXT

Note: Due to an inconsistency in [7] specification it is not possible to cover the test case when an applet try to post data in multitriggering

6.3.1.4.4 Test Coverage

CR Number	*Test Case Number*
CRRN1	14,15,16,17,18, 19
CRRN2	14,15,16,17,18, 19
CRRN3	14,15,16,17,18, 19
CRRC1	1,2,3,4,5,6,7,8,9,10,11,12,13

6.3.2 Handler Integrity

6.3.2.1 ProactiveHandler

Test Area Reference: FWK_HIN_ PAHD

6.3.2.1.1 Conformance Requirement

Normal Execution

CRRN1: At the processToolkit invocation the TLV-List is cleared.

CRRN2: After a call to ProactiveHandler.send method the handler will remain unchanged until the ProactiveHandler.init or appendTLV method are called.

6.3.2.1.2 Test Suite Files:

Test Script: FWK_HIN_ PAHD_1.scr

Test Applet: FWK_ HIN_ PAHD_1.java

FWK_HIN_PAHD_2.java

Load Script: FWK_ HIN_ PAHD_1.ldr

Cleanup Script: FWK_ HIN_ PAHD_1.clr

Parameter File: FWK_ HIN_ PAHD_1.par

6.3.2.1.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**At the processToolkit invocation the TLV-List is cleared** Applet1 and Applet2 are registered to EVENT_UNFORMATTED_SMS_PP_ENV. 1-An envelope containing an unformatted sms pp is sent to the SIM 2-ProactiveHandler.getLength() method is called by Applet1	1- Applet1 is triggered. 2-The return value is 0	
2	**TLV-List change after the init method invocation** ProactiveHandler.init() method is called by Applet1 1-ProactiveHandler.getLength() method is called by Applet1	1-The return value is 9	
3	**The TLV-List remains unchanged after the send() method invocation** 1-ProactiveHandler.send() method is called by Applet1 2-ProactiveHandler.getLength() method is called by Applet1 It's checked that the content is the same than before the calling to send method using ProactiveHandler.copyValue and Util.arrayCompare methods Applet1 finalizes	2-The return value is 9, and its contents is the same than before the calling to send method	1-The proactive command is fetched and the terminal response is issued.
4	**At the processToolkit invocation the TLV-List is cleared** 1-ProactiveHandler.getLength() method is called by Applet2 2-ProactiveHandler.getValueLength() method is called by Applet2	Applet2 is triggered 1-The return value is 0 2-ToolkitException UNAVAILABLE_ELEMENT is thrown	

6.3.2.1.4 Test Coverage

CR Number	***Test Case Number***

CR Number	Test Case Number
CRRN1	1, 2, 3, 4
CRRN2	3

6.3.2.2 ProactiveResponseHandler

Test Area Reference: FWK_HIN_ PRHD

6.3.2.2.1 Conformance Requirement

Normal Execution

CRRN1: The ProactiveResponseHandler content is changed after the call to ProactiveHandler.send method and remains unchanged until next call to the ProactiveHandler.send method.

CRRN2: The ProactiveResponseHandler may not be available before the first call to ProactiveHandler.send method, if available the content is cleared.

6.3.2.2.2 Test Suite Files

Test Script:	FWK_HIN_ PRHD_1.scr
Test Applet:	FWK_ HIN_ PRHD_1.java
Load Script:	FWK_ HIN_ PRHD_1.ldr
Cleanup Script:	FWK_ HIN_ PRHD_1.clr
Parameter File:	FWK_ HIN_ PRHD_1.par

6.3.2.2.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet registration and ProactiveResponseHandler obtaining** 1-Applet is registered to all events defined in [7]. Using the methods initMenuEntry for EVENT_MENU_SELECTION, requestPollInterval() for EVENT_STATUS_COMMAND, allocateTimer() for EVENT_TIMER_EXPIRATION and setEventList() for the rest of the events. Terminal Profile command is sent to the SIM without the facility of SET_EVENT_LIST ,SETUP_IDLE_MODE_TEXT, SETUP_MENU and POLL_INTERVAL. For each event: 2-ProactiveResponseHandler.getTheHandler() is called If handler is available, ProactiveResponseHandler.getLength() is called	1-No exception is thrown Applet is triggered. 2-Behaviour 1: Toolkit Exception HANDLER_NOT_AVAILABLE is thrown. Behaviour 2: No exception is thrown, the return value is 0	
2	**The ProactiveResponseHandler remains unchange after send method invocation until next send method invocation** 1-Applet builds a proactive command ProactiveHandler.send() method is called 2-ProactiveResponseHandler.getLength() method is called 3-ProactiveHandler.init() method is called 4-ProactiveHandler.send() method is called 5-ProactiveResponseHandler.getLength() method is called	1- The ProactiveResponseHandler contains the terminal response 2-The return value is 12 3-No exception is thrown and the Proactive Response Handler remains unchanged 4- The ProactiveResponseHandler contains the terminal response of the second proactive command 5-The return value is 15	1-A proactive command is fetched The terminal response is sent with length 12 4-A proactive command is fetched The terminal response is sent with length 15

6.3.2.2.4 Test Coverage

CR Number	***Test Case Number***
CRRN1	1,2
CRRN2	1

6.3.2.3 EnvelopeHandler

Test Area Reference: FWK_HIN_ ENHD

6.3.2.3.1 Conformance Requirement

Normal Execution

CRRN1: The EnvelopeHandler and its content are available for all triggered toolkit applets, from the invocation to the termination of their processToolkit method~~content shall have the same value during the processToolkit~~

CRRN2: The SIM Toolkit Framework guarantees that all triggered toolkit applets receive the data.

6.3.2.3.2 Test Suite Files

Test Script:	FWK_HIN_ ENHD_1.scr
Test Applet:	FWK_ HIN_ ENHD_1.java
Load Script:	FWK_ HIN_ ENHD_1.ldr
Cleanup Script:	FWK_ HIN_ ENHD_1.clr
Parameter File:	FWK_ HIN_ ENHD_1.par

6.3.2.3.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet initialization and Envelope Handler integrity checks with EVENT_MENU_SELECTION_HELP_REQUEST** 1- Applet is registered to all events defined in [7] except EVENT_PROFILE_DOWNLOAD and EVENT_STATUS_COMMAND. Using the methods initMenuEntry() for EVENT_MENU_SELECTION, allocateTimer()for EVENT_TIMER_EXPIRATION, and setEventList() for the rest of the events. Perform SIM initialization with all the facilities supported 2-Envelope menu selection with help request is sent to the SIM 3-EnvelopeHandler.getTheHandler() method is called 4-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy() The EnvelopeHandler.findTLV() method is called with TAG_HELP_REQUEST 5-A proactive command DISPLAY TEXT is sent 6-Envelope call control by sim is sent to SIM EnvelopeHandler.getTheHandler() method is called 7- It's checked that the contents of the envelope handler is the envelope call	1-No exception is thrown 2- Applet is triggered 3-No exception is thrown. 4-No exception is thrown 6- Applet is triggered 7-No exception is thrown and the handler contains the envelope call	5-91 xx.

Id	Description	API/Framework Expectation	APDU Expectation
	control using EnvelopeHandler.copy() and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished. Check that the TAG_HELP_REQUEST is the TLV selected 8-The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()	control by SIM 8-The contents of the envelope handler shall be the same as stored in buffer 1	A proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM
2	**Envelope Handler integrity checks with EVENT_MENU_SELECTION** 1-An envelope menu selection is sent to SIM 2-EnvelopeHandler.getTheHandler() method is called 3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy() The EnvelopeHandler.findTLV() method is called with TAG_ITEM_IDENTIFIER 4-A proactive command DISPLAY TEXT is sent 5-Envelope call control by sim is sent to SIM EnvelopeHandler.getTheHandler() method is called 6- It's checked the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy() and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished. It's checked that the TAG_ITEM_IDENTIFIER is the TLV selected 7- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()	1- Applet is triggered 2-No exception is thrown. 3-No exception is thrown. 5Applet is triggered 6- No exception is thrown and the handler contains the envelope call control by SIM 7- The contents of the envelope handler shall be the same as stored	4-91 XX Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM

Id	Description	API/Framework Expectation	APDU Expectation
		in buffer 1	

Id	Description	API/Framework Expectation	APDU Expectation
3	**Envelope Handler integrity checks with EVENT_FORMATTED_SMS_PP_ENV** 1-A formatted sms pp envelope is sent to SIM 2-EnvelopeHandler.getTheHandler() method is called 3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy( ) The EnvelopeHandler.findTLV() method is called with TAG_SMS_TPDU 4-A proactive command DISPLAY TEXT is sent 5-Envelope call control by sim is sent to SIM EnvelopeHandler.getTheHandler() method is called 6-It's checked that the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy( ) and Util.arrayCompare methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished. It's checked that the TAG_SMS_TPDU is the TLV selected 7- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()	1- Applet is triggered 2-No exception is thrown. 3-No exception is thrown. 5- Applet is triggered 6- No exception is thrown and the handler contains the envelope call control by SIM 7- The contents of the envelope handler shall be the same as stored in buffer 1	4-91 XX Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM
4	**Envelope Handler integrity checks with EVENT_UNFORMATTED_SMS_PP_ENV** 1-A unformatted sms pp envelope is sent to SIM 2-EnvelopeHandler.getTheHandler() method is called 3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy( ) The EnvelopeHandler.findTLV method is called with TAG_DEVICE_IDENTITIES 4-A proactive command DISPLAY TEXT is sent 5-Envelope call control by sim is sent to SIM EnvelopeHandler.getTheHandler() method is called	1- Applet is triggered 2-No exception is thrown. 3-No exception is thrown. 5- Applet is triggered	4-91 XX

Id	Description	API/Framework Expectation	APDU Expectation
	6-It's checked that the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy() and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished.	6- No exception is thrown and the handler contains the envelope call control by SIM	Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM
	It's checked that the TAG_DEVICE_IDENTITIES is the TLV selected 7- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()	7- The contents of the envelope handler shall be the same as stored in buffer 1.	
5	**Envelope Handler integrity checks with EVENT_UNFORMATTED_SMS_CB** 1-A unformatted cellbroadcast envelope is sent to SIM	1- Applet is triggered	
	2-EnvelopeHandler.getTheHandler() method is called	2-No exception is thrown.	
	3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy() The EnvelopeHandler.findTLV() method is called with TAG_CELLBROADCAST_PAGE	3-No exception is thrown.	
	4-A proactive command DISPLAY TEXT is sent		4-91 XX
	5-Envelope call control by sim is sent to SIM EnvelopeHandler.getTheHandler() method is called	5- Applet is triggered	
	6-It's checked that the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished.	6- No exception is thrown and the handler contains the envelope call control by SIM	Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM
	It's checked that the TAG_CELLBROADCAST_PAGE is the TLV selected 7- The contents of EnvelopeHandler are compared with buffer1 using	7- The contents of the envelope handler shall be the same as stored in buffer 1.	

Id	Description	API/Framework Expectation	APDU Expectation
	Util.arrayCompare()		
6	**Envelope Handler integrity checks with EVENT_TIMER_EXPIRATION** 1-A timer expiration envelope is sent to SIM 2-EnvelopeHandler.getTheHandler() method is called 3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy() The EnvelopeHandler.findTLV() method is called with TAG_TIMER_ID 4-A proactive command DISPLAY TEXT is sent 5-Envelope call control by sim is sent to SIM EnvelopeHandler.getTheHandler() method is called 6-It's checked that the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy() and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished. It's checked that the TAG_TIMER_IDE is the TLV selected 7- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()	1- Applet is triggered 2-No exception is thrown. 3-No exception is thrown. 5- Applet is triggered 6- No exception is thrown and the handler contains the envelope call control by SIM 7- The contents of the envelope handler shall be the same as stored in buffer 1	4-91 XX Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM
7	**Envelope Handler integrity checks with EVENT_CALL_CONTROL_BY_SIM** 1-A call control envelope is sent to SIM 2-EnvelopeHandler.getTheHandler() method is called 3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy() The EnvelopeHandler.findTLV() method is called with TAG_ADDRESS 4-A proactive command DISPLAY TEXT is sent 5-Envelope call control by sim is sent to SIM	1- Applet is triggered 2-No exception is thrown. 3-No exception is thrown. 5- Applet is triggered	4-91 XX

Id	Description	API/Framework Expectation	APDU Expectation
	EnvelopeHandler.getTheHandler() method is called 6-It's checked that the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy() and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished.	6- No exception is thrown and the handler contains the envelope call control by SIM	Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM
	It's checked that the TAG_ADDRESS is the TLV selected 7- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()	7- The contents of the envelope handler shall be the same as stored in buffer 1	
8	**Envelope Handler integrity checks with EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM** 1-A mo short message control by sim envelope is sent to SIM	1- Applet is triggered	
	2-EnvelopeHandler.getTheHandler() method is called	2-No exception is thrown.	
	3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy() The EnvelopeHandler.findTLV() method is called with TAG_ADDRESS	3-No exception is thrown.	
	4-A proactive command DISPLAY TEXT is sent		4-91 XX
	5-Envelope call control by sim is sent to SIM EnvelopeHandler.getTheHandler() method is called	5- Applet is triggered	
	6-It's checked that the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy() and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished.	6- No exception is thrown and the handler contains the envelope call control by SIM	Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM
	It's checked that the TAG_ADDRESS is the TLV selected 7- The contents of EnvelopeHandler are	7- The contents of the envelope	

Id	Description	API/Framework Expectation	APDU Expectation
	compared with buffer1 using Util.arrayCompare()	handler shall be the same as stored in buffer 1.	
9	**Envelope Handler integrity checks with EVENT_EVENT_DOWNLOAD_MT_CALL** 1-A event download mt call envelope is sent to SIM 2-EnvelopeHandler.getTheHandler() method is called 3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy() The EnvelopeHandler.findTLV() method is called with TAG_ADDRESS 4-A proactive command DISPLAY TEXT is sent 5-Envelope call control by sim is sent to SIM EnvelopeHandler.getTheHandler() method is called 6-It's checked that the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy() and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished. It's checked that the TAG_ADDRESS is the TLV selected 7- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()	1- 2-No exception is thrown. 3-No exception is thrown. 5- Applet is triggered 6- No exception is thrown and the handler contains the envelope call control by SIM 7- The contents of the envelope handler shall be the same as stored in buffer 1	4-91 XX Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM
10	**Envelope Handler integrity checks with EVENT_EVENT_DOWNLOAD_CALL_CONNECTED** 1-A event download call connected envelope is sent to SIM 2-EnvelopeHandler.getTheHandler() method is called 3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy() The EnvelopeHandler.findTLV() method is called with TAG_ADDRESS 4-A proactive command DISPLAY TEXT is sent	1- Applet is triggered 2-No exception is thrown. 3-No exception is thrown.	4-91 XX

Id	Description	API/Framework Expectation	APDU Expectation
	5-Envelope call control by sim is sent to SIM EnvelopeHandler.getTheHandler() method is called 6-It's checked that the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished. It's checked that the TAG_ADDRESS is the TLV selected 7- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()	5- Applet is triggered 6- No exception is thrown and the handler contains the envelope call control by SIM 7- The contents of the envelope handler shall be the same as stored in buffer 1.	Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM
11	**Envelope Handler integrity checks with EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED** 1-A event download call disconnected envelope is sent to SIM 2-EnvelopeHandler.getTheHandler() method is called 3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy() The EnvelopeHandler.findTLV() method is called with TAG_ADDRESS 4-A proactive command DISPLAY TEXT is sent 5-Envelope call control by sim is sent to SIM EnvelopeHandler.getTheHandler() method is called 6-It's checked that the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished.	1- Applet is triggered 2-No exception is thrown. 3-No exception is thrown. 5- Applet is triggered 6- No exception is thrown and the handler contains the envelope call control by SIM	4-91 XX Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM

Id	Description	API/Framework Expectation	APDU Expectation
	It's checked that the TAG_ADDRESS is the TLV selected 7- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()	7- The contents of the envelope handler shall be the same as stored in buffer 1.	
12	**Envelope Handler integrity checks with EVENT_EVENT_DOWNLOAD_LOCATION_STATUS** 1-A event download location status envelope is sent to SIM 2-EnvelopeHandler.getTheHandler() method is called 3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy() The EnvelopeHandler.findTLV() method is called with TAG_LOCATION_STATUS 4-A proactive command DISPLAY TEXT is sent 5-Envelope call control by sim is sent to SIM EnvelopeHandler.getTheHandler() method is called 6-It's checked that the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy() and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished. It's checked that the TAG_LOCATION_STATUS is the TLV selected 7- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()	1- Applet is triggered 2-No exception is thrown. 3-No exception is thrown. 5- Applet is triggered 6- No exception is thrown and the handler contains the envelope call control by SIM 7- The contents of the envelope handler shall be the same as stored in buffer 1	4-91 XX Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM

Id	Description	API/Framework Expectation	APDU Expectation
13	**Envelope Handler integrity checks with EVENT_EVENT_DOWNLOAD_USER_ACTIVITY** 1-A event download user activity envelope is sent to SIM 2-EnvelopeHandler.getTheHandler() method is called 3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy() The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES 4-A proactive command DISPLAY TEXT is sent 5-Envelope call control by sim is sent to SIM EnvelopeHandler.getTheHandler() method is called 6-It’s checked that the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy() and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished. It's checked that the TAG_DEVICE_IDENTITIES is the TLV selected 7- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()	1- Applet is triggered 2-No exception is thrown. 3-No exception is thrown. 5- Applet is triggered 6- No exception is thrown and the handler contains the envelope call control by SIM 7- The contents of the envelope handler shall be the same as stored in buffer 1	4-91 XX Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM
14	**Envelope Handler integrity checks with EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE** 1-A event download idle screen available envelope is sent to SIM 2-EnvelopeHandler.getTheHandler() method is called 3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy() The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES 4-A proactive command DISPLAY TEXT is sent 5-Envelope call control by sim is sent to SIM	1- Applet is triggered 2-No exception is thrown. 3-No exception is thrown. 5- Applet is triggered	4-91 XX

Id	Description	API/Framework Expectation	APDU Expectation
	EnvelopeHandler.getTheHandler() method is called 6-It's checked that the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy() and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished. It's checked that the TAG_DEVICE_IDENTITIES is the TLV selected 7- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()	6- No exception is thrown and the handler contains the envelope call control by SIM 7- The contents of the envelope handler shall be the same as stored in buffer 1	Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM
15	**Envelope Handler integrity checks with EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS** 1-A event download card reader status envelope is sent to SIM 2-EnvelopeHandler.getTheHandler() method is called 3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy() The EnvelopeHandler.findTLV() method is called with TAG_CARD_READER_STATUS 4-A proactive command DISPLAY TEXT is sent 5-Envelope call control by sim is sent to SIM EnvelopeHandler.getTheHandler() method is called 6-It's checked that the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES It's checked that the TAG_CARD_READER_STATUS is the TLV selected	1- Applet is triggered 2-No exception is thrown. 3-No exception is thrown. 5- Applet is triggered 6- No exception is thrown and the handler contains the envelope call control by SIM	4-91 XX Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM

Id	Description	API/Framework Expectation	APDU Expectation
	7- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()	7- The contents of the envelope handler shall be the same as stored in buffer 1	
16	**Envelope Handler integrity checks with UNRECOGNIZED_ENVELOPE** 1-A unrecognized envelope is sent to SIM 2-EnvelopeHandler.getTheHandler() method is called 3-Copy the contents of the envelope handler in buffer 1 using EnvelopeHandler.copy() 4-A proactive command DISPLAY TEXT is sent 5-Envelope call control by sim is sent to SIM EnvelopeHandler.getTheHandler() method is called The EnvelopeHandler.getValueLength() is called 6-It's checked that the contents of the envelope handler is the envelope call control using EnvelopeHandler.copy() and Util.arrayCompare() methods The EnvelopeHandler.findTLV() method is called with TAG_DEVICE_IDENTITIES Call Control execution is finished. 7- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()	1- Applet is triggered 2-No exception is thrown. 3-No exception is thrown. 5- Applet is triggered 6- No exception is thrown and the handler contains the envelope call control by SIM 7- The contents of the envelope handler shall be the same as stored in buffer 1	4-91 XX Proactive command Display Text is fetched The terminal Response of DISPLAY TEXT is sent to the SIM

6.3.2.3.4 Test Coverage

CR Number	Test Case Number
CRRN1	1,2,3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16
CRRN2	1,2,3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16

6.3.3 Applet Triggering

6.3.3.1 EVENT_PROFILE_DOWNLOAD

Test Area Reference: FWK_APT_EPDW

6.3.3.1.1 Conformance Requirement

Normal Execution

CRRN1: Upon the reception of Terminal Profile command by the SIM, the STF stores the ME Profile and then triggers the registered toolkit applets.

CRRN2: The applet is not triggered by the EVENT_PROFILE_DOWNLOAD once it has deregistered from this event.

CRRN3: The STF shall not reply busy to a Terminal Profile command

6.3.3.1.2 Test Suite Files

Test Script:	FWK_APT_EPDW_1.scr
Test Applet:	FWK_APT_EPDW_1.java
	FWK_APT_EPDW_2.java
	FWK_APT_EPDW_3.java
Load Script:	FWK_APT_EPDW_1.ldr
Cleanup Script:	FWK_APT_EPDW_1.clr
Parameter File:	FWK_APT_EPDW_1.par

6.3.3.1.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applets registration to EVENT_PROFILE_DOWNLOAD and triggering** Applet1 is registered to the EVENT_PROFILE_DOWNLOAD Applet2 is registered to the EVENT_PROFILE_DOWNLOAD Applet3 is not registered to the EVENT_PROFILE_DOWNLOAD and is registered to EVENT_FORMATTED_SMS_PP_ENV. 1-Terminal Profile command is sent to SIM Applet1 execution is finished Applet2 execution is finished	1- Applet1 is triggered Applet2 is triggered Applet3 is not triggered	
2	**The STF shall not reply busy to a Terminal Profile command** 1-Formatted sms pp envelope is sent to SIM Applet3 builds a REFRESH proactive command in sim initialization mode 2-ProactiveHandler.send() method is called by applet3	Applet3 is triggered by the EVENT_FORMATTED_SMS_PP_ENV Applet3 is suspended until the	2-A proactive command is sent

Id	Description	API/Framework Expectation	APDU Expectation
	3-Terminal Profile command is sent to SIM Applet1 calls Toolkit Registry.clearEvent(EVENT_PROFILE_DOWNLOAD) 4-Applet2 calls Toolkit Registry.clearEvent(EVENT_PROFILE_DOWNLOAD) ToolkitRegistry.setEvent(EVENT_PROFILE_DOWNLOAD) method is called Applet3 execution finish	terminal response 3- Applet1 is triggered by EVENT_PROFILE_DOWNLOAD 4- Applet2 is triggered by EVENT_PROFILE_DOWNLOAD	The terminal Response of the proactive command is sent
3	**Deregistered applets are not triggered** Terminal Profile command is sent to SIM	Applet3 is triggered (Applet1 and Applet2 are not triggered)	

6.3.3.1.4 Test Coverage

CR Number	Test Case Number
CRRN1	1
CRRN2	3
CRRN3	2

6.3.3.2 EVENT_MENU_SELECTION

Test Area Reference: FWK_APT_EMSE

6.3.3.2.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_MENU_SELECTION when an Envelope Menu Selection is received with the item identifier of a menu entry of this applet if no proactive session is ongoing.

6.3.3.2.2 Test Suite Files

Test Script: FWK_APT_EMSE_1.scr

Test Applet: FWK_APT_EMSE_1.java

FWK_APT_EMSE_2.java

Load Script: FWK_APT_EMSE_1.ldr

Cleanup Script: FWK_APT_EMSE_1.clr

Parameter File: FWK_APT_EMSE_1.par

6.3.3.2.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet registration to EVENT_MENU_SELECTION and triggering** ToolkitRegistry.initMenuEntry() method is called in the constructor of applet1 and Applet2. For applet1: MenuEntry="Applet1" Offset=0 Length=menuEntry.length HelpSupported=false IconQualifier=0 IconIdentifier=0 For applet2: MenuEntry="Applet2" Offset=0 Length=menuEntry.length HelpSupported=false IconQualifier=0 IconIdentifier=0 event= EVENT_MENU_SELECTION 1-ToolkitRegistry.isEventSet() is called in constructor. Perform SIM initialization the facility SET UP MENU and without the facilities SET EVENT LIST and POLL INTEVAL features 2-Item Identifier = 1 Event Menu Selection envelope is sent to the SIM with the item identifier of a menu entry of applet 3-Item Identifier = 2 Event Menu Selection envelope is sent to the SIM with the item identifier of a menu entry of applet	1-The method must return true. 2- Applet1 is triggered and applet2 is not triggered 3-Applet2 is triggered and applet1 is not triggered	

6.3.3.2.4 Test Coverage

CR Number	Test Case Number
CRRN1	1

6.3.3.3 EVENT_MENU_SELECTION_HELP_REQUEST

Test Area Reference: FWK_APT_EMSH

6.3.3.3.1 Conformance Requirement

Normal Execution

CRRN1: If and ENVELOPE (MENU_SELECTION_HELP_SUPPORTED) command is received for one entry supporting help, then STF shall trigger the corresponding applet.

6.3.3.3.2 Test Suite Files

Test Script: FWK_APT_EMSH_1.scr

Test Applet: FWK_APT_EMSH_1.java

FWK_APT_EMSH_2.java

Load Script: FWK_APT_EMSH_1.ldr

Cleanup Script: FWK_APT_EMSH_1.clr

Parameter File: FWK_APT_EMSH_1.par

~~none~~

6.3.3.3.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet registration to EVENT_MENU_SELECTION_HELP_REQUEST and triggering** ToolkitRegistry.InitMenuEntry() method is called in the constructor of applet1 and Applet2. For Applet1: MenuEntry="Applet1" Offset=0 Length=menuEntry.length HelpSupported=true IconQualifier=0 IconIdentifier=0 For Applet2: MenuEntry="Applet2" Offset=0 Length=menuEntry.length HelpSupported=true IconQualifier=0 IconIdentifier=0 event= EVENT_MENU_SELECTION_HELP_REQUEST 1-ToolkitRegistry.isEventSet() is called in constructor. Perform SIM initialization the facility SET UP MENU and without the facilities SET EVENT LIST and POLL INTEVAL features 2-Item identifier =1 Menu Selection Help Request envelope is sent to the SIM with the item identifier of a menu entry of applet 3-Item identifier =2 Menu Selection Help Request envelope is sent to the SIM with the item identifier of a menu entry of applet	1-The command must return true. 2-Applet1 is triggered and applet2 is not triggered 3-Applet2 is triggered and applet1 is not triggered	

6.3.3.3.4 Test Coverage

CR Number	Test Case Number
CRRN1	1

6.3.3.4 EVENT_FORMATTED_SMS_PP_ENV

Test Area Reference: FWK_APT_EFSE

6.3.3.4.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_FORMATTED_SMS_PP_ENV once it has registered to this event and Formatted Envelope DataDownLoad with the corresponding TAR, defined at the applet loading, is received and no proactive session is ongoing

CRRN2: The applet is not triggered by the EVENT_FORMATTED_SMS_PP_ENV once it has deregistered from this event.

6.3.3.4.2 Test Suite Files

Test Script: FWK_APT_EFSE_1.scr

Test Applet: FWK_APT_EFSE_1.java

Load Script: FWK_APT_EFSE_1.ldr

Cleanup Script: FWK_APT_EFSE_1.clr

Parameter File: FWK_APT_EFSE_1.par

6.3.3.4.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet registration to EVENT *FORMATTED* SMS PP ENV and triggering** Applet is registered to EVENT_FORMATTED_SMS_PP_ENV and EVENT_UNRECOGNIZED_ENVELOPE 1-An Envelope EVENT_FORMATTED_SMS_PP_ENV is sent to the SIM.	1-Applet is triggered	
2	**Applet deregistration** ToolkitRegistry.clearEvent() method is called for EVENT_FORMATTED_SMS_PP_ENV 2-A formatted sms pp envelope is sent to the SIM. An unrecognized envelope is sent to the sim ToolkitRegistry.setEvent() method is called for EVENT_FORMATTED_SMS_PP_ENV 3-An Envelope FORMATTED_SMS_PP_ENV is sent to the SIM	2-Applet is not triggered 3- Applet is triggered	

6.3.3.4.4 Test Coverage

CR Number	Test Case Number
CRRN1	1
CRRN2	2

6.3.3.5 EVENT_UNFORMATTED_SMS_PP_ENV

Test Area Reference: FWK_APT_EUSE

6.3.3.5.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_ UNFORMATTED_SMS_PP_ENV once it has registered to this event and an Unformatted Envelope DataDownLoad is received if no proactive session is ongoing

CRRN2: The applet is not triggered by the EVENT_ UNFORMATTED_SMS_PP_ENV once it has deregistered from this event.

6.3.3.5.2 Test Suite Files

Test Script:	FWK_APT_EUSE_1.scr
Test Applet:	FWK_APT_EUSE_1.java
Load Script:	FWK_APT_EUSE_1.ldr
Cleanup Script:	FWK_APT_EUSE_1.clr
Parameter File:	FWK_APT_EUSE_1.par

6.3.3.5.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet registration to EVENT_UNFORMATTED_SMS_PP_ENV and triggering** Applet is registered to the EVENT_UNFORMATTED_SMS_PP_ENV and ENVENT_FORMATTED_SMS_PP_ENV. 1-Toolkit Registry.isEventSet() method is called for EVENT_UNFORMATTED_SMS_PP_ENV 2-An Envelope UNFORMATTED_SMS_PP_ENV is sent to the SIM.	1-The method returns true 2-Applet is triggered	
2	**Applet deregistration** Toolkit Registry.clearEvent()method is called for EVENT_UNFORMATTED_SMS_PP_ENV 1-An unformatted sms pp envelope is sent to the SIM. A formatted sms pp envelope is sent to the sim Toolkit Registry.setEvent() method is called for EVENT_UNFORMATTED_SMS_PP_ENV 2-An Envelope UNFORMATTED_SMS_PP_ENV is sent to the SIM	1- Applet isn't triggered 2- Applet is triggered	

6.3.3.5.4 Test Coverage

CR Number	*Test Case Number*
CRRN1	1
CRRN2	2

6.3.3.6 EVENT_CALL_CONTROL_BY_SIM

Test Area Reference: FWK_APT_ECCN

6.3.3.6.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_CALL_CONTROL_BY_SIM once it has registered to this event and an Envelope Call Control is received

CRRN2: The applet is not triggered by the EVENT_CALL_CONTROL_BY_SIM once it has deregistered from this event.

6.3.3.6.2 Test Suite Files

Test Script: FWK_APT_ECCN_1.scr

Test Applet: FWK_APT_ECCN_1.java

Load Script: FWK_APT_ECCN_1.ldr

Cleanup Script: FWK_APT_ECCN_1.clr

Parameter File: FWK_APT_ECCN_1.par

6.3.3.6.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applets registration to EVENT_CALL_CONTROL_BY_SIM and triggering** Applet1 is registered to EVENT_CALL_CONTROL_BY_SIM. Applet2 is registered to EVENT_FORMATTED_SMS_PP_ENV 1-An Envelope Call control by SIM is sent to SIM Applet1 execution is finished	1- Applet1 is triggered	
2	**Applet deregistration and registration of the third applet to EVENT_CALL-CONTROL_BY_SIM.** 1-An Envelope Formatted SMS PP envelope is sent to SIM Applet2 contructs a DISPLAY TEXT proactive command. 2-ProactiveHandler.send() method is called 3-An Envelope Call control by SIM envelope is sent to SIM ToolkitRegistry.clearEvent() is called for EVENT_CALL_CONTROL_BY_SIM. Applet1 finalizes. ToolkitRegistry.setEvent() method is called for EVENT_CALL_CONTROL_BY_SIM. Applet2 is finished	1-Applet2 is triggered by EVENT_FORMATTED_SMS_PP_ENV. 3- Applet1 is triggered	2- A proactive command DISPLAY TEXT is sent and applet is suspended until the terminal response TERMINAL RESPONSE of DISPLAY TEXT is sent to the SIM
3	**Applet triggering** An Envelope Call control by SIM envelope is sent ot SIM	Applet2 is triggered. (Applet1 is not triggered)	

6.3.3.6.4 Test Coverage

CR Number	Test Case Number

CR Number	Test Case Number
CRRN1	1,2
CRRN2	3

6.3.3.7 EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM

Test Area Reference: FWK_APT_EMCN

6.3.3.7.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM once it has registered to this event and an Envelope MO Short Message Control.

CRRN2: The applet is not triggered by the EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM once it has deregistered from this event.

6.3.3.7.2 Test Suite Files

Test Script:	FWK_APT_EMCN_1.scr
Test Applet:	FWK_APT_EMCN_1.java
	FWK_APT_EMCN_2.java
Load Script:	FWK_APT_EMCN_1.ldr
Cleanup Script:	FWK_APT_EMCN_1.clr
Parameter File:	FWK_APT_EMCN_1.par

6.3.3.7.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet registration to EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM and triggering** Applet1 is reggistered to EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM. Applet2 is registered to EVENT_FORMATTED_SMS_PP_ENV. 1-An Envelope MO short message envelope is sent to SIM Applet1 execution is finished	1- Applet1 is triggered.	
2	**Applet deregistration and registration of the third applet to EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM.** **The STF shall not reply busy to a call control envelope** 1-An Envelope formatted SMS PP envelope is sent to SIM. Applet2 builds a DISPLAY TEXT proactive command. 2-ProactiveHAndler.send() method is called. 3-An Envelope MO Short message envelope is sent to SIM ToolkitRegistry.clearEvent() for EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM. Applet1 finalizes. ToolkitRegistry.setEvent() method is called for EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM. Applet2 execution finished.	1- Applet2 is triggered. 3-Applet1 is triggered.	2- A Proactive command DISPLAY TEXT is sent and applet is suspended until the terminal response TERMINAL RESPONSE of DISPLAY TEXT is sent to the SIM
3	**Applet3 triggering** An Envelope MO SMS control by SIM envelope is sent ot SIM	Applet2 is triggered. (Applet1 is not triggered)	

6.3.3.7.4 Test Coverage

CR Number	Test Case Number
CRRN1	1,2
CRRN2	3

6.3.3.8 EVENT_TIMER_EXPIRATION

Test Area Reference: FWK_APT_ETEX

6.3.3.8.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_TIMER_EXPIRATION once it has been registered to this event and an Envelope Timer Expiration with a Timer Identifier of the applet is received if no proactive session is ongoing.

CRRN2: The applet is not triggered by the EVENT_TIMER_EXPIRATION once it has been deregistered from this event.

6.3.3.8.2 Test Suite Files

Test Script:	FWK_APT_ETEX_1.scr
Test Applet:	FWK_APT_ETEX_1.java
Load Script:	FWK_APT_ETEX_1.ldr
Cleanup Script:	FWK_APT_ETEX_1.clr
Parameter File:	FWK_APT_ETEX_1.par

6.3.3.8.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet registration to EVENT_TIMER_EXPIRATION and triggering** Applet is registered to the EVENT_TIMER_EXPIRATION using the allocateTimer() method and to EVENT_FORMATTED_SMS_PP_ENV. event= EVENT_TIMER_EXPIRATION 1-Toolkit Registry.isEventSet() method is called. 2-An Envelope TIMER_EXPIRATION is sent to the SIM.	1-The method returns true 2- Applet is triggered.	
2	**Applet deregistration** Timer id=1 Toolkit Registry.ReleaseTimer() method is called 1-An Envelope timer expiration is sent to the SIM. An Envelope formated sms pp envelope is sent to the sim Toolkit Registry.AllocateTimer() method is called 2-An Envelope TIMER_EXPIRATION is sent to the SIM.	1- Applet isn't triggered 2- Applet is triggered	

6.3.3.8.4 Test Coverage

CR Number	Test Case Number
CRRN1	1
CRRN2	2

6.3.3.9 EVENT_UNFORMATTED_SMS_CB

Test Area Reference: FWK_APT_EUCB

6.3.3.9.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_UNFORMATTED_SMS_CB once it has registered to this event and an Envelope Cell Broadcast DownLoad is received.

CRRN2: The applet is not triggered by the EVENT_UNFORMATTED_SMS_CB once it has deregistered from this event.

6.3.3.9.2 Test Suite Files

Test Script: FWK_APT_EUCB_1.scr

Test Applet: FWK_APT_EUCB_1.java

Load Script: FWK_APT_EUCB_1.ldr

Cleanup Script: FWK_APT_EUCB_1.clr

Parameter File: FWK_APT_EUCB_1.par

6.3.3.9.3 Test Procedure

Id	Description	API Expectation	APDU Expectation
1	**Applet registration to EVENT_UNFORMATTED_SMS_CB and triggering** Applet is registered to the EVENT_UNFORMATTED_SMS_CB and EVENT_FORMATTED_SMS_PP_ENV. event= EVENT_UNFORMATTED_SMS_CB 1-Toolkit Registry.isEventSet() method is called. 2-An Envelope UNFORMATTED_SMS_CB is sent to the SIM.	1- Method returns true. 2- Applet is triggered	
2	**Applet deregistration** Toolkit Registry.ClearEvent()method is called for EVENT_UNFORMATTED_SMS_CB 1-An Envelope UNFORMATTED_SMS_CB is sent to the SIM. An Envelpe formatted sms pp envelope is sent to the sim event= EVENT_UNFORMATTED_SMS_CB	1- Applet isn't triggered	

Id	Description	API Expectation	APDU Expectation
	Toolkit Registry.setEvent() method is called for EVENT_UNFORMATTED_SMS_CB 2-An Envelope UNFORMATTED_SMS_CB is sent to the SIM.	2-Applet is triggered	

6.3.3.9.4 Test Coverage

CR Number	Test Case Number
CRRN1	1
CRRN2	2

6.3.3.10 EVENT_EVENT_DOWNLOAD_MT_CALL

Test Area Reference: FWK_APT_EDMC

6.3.3.10.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_EVENT_DOWNLOAD_MT_CALL once it has registered to this event and an Envelope Event DownLoad MT Call is received.

CRRN2: The applet is not triggered by the EVENT_EVENT_DOWNLOAD_MT_CALL once it has deregistered from this event.

6.3.3.10.2 Test Suite Files

Test Script: FWK_APT_EMSE_1.scr

Test Applet: FWK_APT_EMSE_1.java

Load Script: FWK_APT_EMSE_1.ldr

Cleanup Script: FWK_APT_EMSE_1.clr

Parameter File: FWK_APT_EMSE_1.par

6.3.3.10.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet registration to EVENT_EVENT_DOWNLOAD_MT_CALL and triggering** Applet is registered to the EVENT_EVENT_DOWNLOAD_MT_CALL and to EVENT_FORMATTED_SMS_PP_ENV. event= EVENT_EVENT_DOWNLOAD_MT_CALL 1-Toolkit Registry.isEventSet() method is called. 2-An Envelope EVENT_DOWNLOAD_MT_CALL is sent to the SIM.	1- The method returns true 2-Applet is triggered	
2	**Applet deregistration** event= EVENT_EVENT_DOWNLOAD_MT_CALL Toolkit Registry.clearEvent()method is called		

Id	Description	API/Framework Expectation	APDU Expectation
	called Perform SIM initialization with all the facilities supported 1-An Envelope EVENT_DOWNLOAD_MT_CALL is sent to the SIM. An Envelpe formatted sms pp envelope is sent to the sim event= EVENT_EVENT_DOWNLOAD_MT_CALL Toolkit Registry.setEvent() method is called Perform SIM initialization with all the facilities supported 2-An Envelope EVENT_DOWNLOAD_MT_CALL is sent to the SIM.	1-Applet isn't triggered 2- Applet is triggered	

6.3.3.10.4 Test Coverage

CR Number	Test Case Number
CRRN1	1
CRRN2	2

6.3.3.11 EVENT_EVENT_DOWNLOAD_CALL_CONNECTED

Test Area Reference: FWK_APT_EDCC

6.3.3.11.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_EVENT_DOWNLOAD_CALL_CONNECTED once it has registered to this event and an Envelope Event DownLoad Call Connected is received.

CRRN2: The applet is not triggered by the EVENT_EVENT_DOWNLOAD_CALL_CONNECTED once it has deregistered from this event.

6.3.3.11.2 Test Suite Files

Test Script: FWK_APT_EDCC_1.scr

Test Applet: FWK_APT_EDCC_1.java

Load Script: FWK_APT_EDCC_1.ldr

Clean-up Script: FWK_APT_EDCC_1.clr

6.3.3.11.3 Test Procedure

1	**Applet registration to EVENT_EVENT_DOWNLOAD_CALL_CONNECTED and triggering** Applet is registered to the EVENT_EVENT_DOWNLOAD_CALL_CONNECTED and to EVENT_FORMATTED_SMS_PP_ENV. event= EVENT_EVENT_DOWNLOAD_CALL_CONNECTED 1-Toolkit Registry.isEventSet() method is called. 2-An Envelope EVENT_DOWNLOAD_CALL_CONNECTED is sent to the SIM.	1-Method returns true 2- Applet is triggered.	
2	**Applet deregistration** event=EVENT_EVENT_DOWNLOAD_CALL_CONNECTED Toolkit Registry.clearEvent()method is called Perform SIM initialization with all the facilities supported 1-A call connected event dowload is sent to the SIM. An Envelope formatted sms pp envelope is sent to the sim Event= EVENT_EVENT_DOWNLOAD_CALL_CONNECTED Toolkit Registry.setEvent() method is called Perform SIM initialization with all the facilities supported 2-An Envelope EVENT_DOWNLOAD_CALL_CONNECTED is sent to the SIM.	1-Applet isn't triggered 2-Applet is triggered	

6.3.3.11.4 Test Coverage

CR Number	***Test Case Number***
CRRN1	1
CRRN2	2

6.3.3.12 EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED

Test Area Reference: FWK_APT_EDCD

6.3.3.12.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED once it has registered to this event and an Envelope Event DownLoad Call Disconnected is received.

CRRN2: The applet is not triggered by the EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED once it has deregistered from this event.

6.3.3.12.2 Test Suite Files

Test Script: FWK_APT_EDCD_1.scr

Test Applet: FWK_APT_EDCD_1.java

Load Script: FWK_APT_EDCD_1.ldr

Cleanup Script: FWK_APT_EDCD_1.clr

Parameter File: FWK_APT_EDCD_1.par

6.3.3.12.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet registration to EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED and triggering** Applet is registered to the EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED and to EVENT_FORMATTED_SMS_PP_ENV. Event=EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED 1-Toolkit Registry.isEventSet() method is called. 2-An Envelope EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED is sent to the SIM.	1-Method returns true 2- Applet is triggered.	
2	**Applet deregistration** Event= EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED Toolkit Registry.clearEvent()method is called Perform SIM initialization with all the facilities supported 1-An Envelope EVENT_DOWNLOAD_CALL_DISCONNECTED is sent to the SIM. a formatted sms pp envelope is sent to the sim. Event= EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED Toolkit Registry.setEvent() method is called Perform SIM initialization with all the facilities supported 2-An Envelope EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED is sent to the SIM.	1-Applet isn't triggered 2- Applet is triggered	

6.3.3.12.4 Test Coverage

CR Number	Test Case Number
CRRN1	1
CRRN2	2

6.3.3.13 EVENT_EVENT_DOWNLOAD_LOCATION_STATUS

Test Area Reference: FWK_APT_EDLS

6.3.3.13.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_EVENT_DOWNLOAD_LOCATION_STATUS once it has registered to this event and an Envelope Event DownLoad Location Status is received.

CRRN2: The applet is not triggered by the EVENT_EVENT_DOWNLOAD_LOCATION_STATUS once it has deregistered from this event.

6.3.3.13.2 Test Suite Files

Test Script:	FWK_APT_EDLS_1.scr
Test Applet:	FWK_APT_EDLS_1.java
Load Script:	FWK_APT_EDLS_1.ldr
Cleanup Script:	FWK_APT_EDLS_1.clr
Parameter File:	FWK_APT_EDLS_1.par

6.3.3.13.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet registration to EVENT_EVENT_DOWNLOAD_LOACTION_STATUS and triggering** Applet is registered to the EVENT_EVENT_DOWNLOAD_LOCATION_STATUS and to EVENT_FORMATTED_SMS_PP_ENV. Event=EVENT_EVENT_DOWNLOAD_LOCATION_STATUS 1-Toolkit Registry.isEventSet() method is called. 2-An Envelope EVENT_EVENT_DOWNLOAD_LOCATION_STATUS is sent to the SIM.	1-Method returns true 2- applet is triggered.	
2	**Applet deregistration** Event=EVENT_EVENT_DOWNLOAD_LOCATION_STATUS Toolkit Registry.clearEvent()method is called Perform SIM initialization with all the facilities supported 1-An Envelope EVENT_DOWNLOAD_LOCATION_STATUS is sent to the SIM. a formatted sms pp envelope is sent to the sim Event= EVENT_EVENT_DOWNLOAD_LOCATION_STATUS Toolkit Registry.setEvent() method is called	1-Applet isn't triggered	

Id	Description	API/Framework Expectation	APDU Expectation
	Perform SIM initialization with all the facilities supported 2-An Envelope EVENT_DOWNLOAD_LOCATION_STATUS is sent to the SIM.	2- Applet is triggered	

6.3.3.13.4 Test Coverage

CR Number	Test Case Number
CRRN1	1
CRRN2	2

6.3.3.14 EVENT_EVENT_DOWNLOAD_USER_ACTIVITY

Test Area Reference: FWK_APT_EDUA

6.3.3.14.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_EVENT_DOWNLOAD_USER_ACTIVITY once it has registered to this event and an Envelope Event DownLoad User Activity is received.

CRRN2: The applet is not triggered by the EVENT_EVENT_DOWNLOAD_USER_ACTIVITY once it has deregistered from this event.

6.3.3.14.2 Test Suite Files

Test Script: FWK_APT_EDUA_1.scr

Test Applet: FWK_APT_EDUA_1.java

Load Script: FWK_APT_EDUA_1.ldr

Cleanup Script: FWK_APT_EDUA_1.clr

Parameter File: FWK_APT_EDUA_1.par

6.3.3.14.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet registration to EVENT_EVENT_DOWNLOAD_USER_ACTIVITY and triggering** Applet is registered to the EVENT_EVENT_DOWNLOAD_USER_ACTIVITY and to EVENT_FORMATTED_SMS_PP_ENV. Event= EVENT_EVENT_DOWNLOAD_USER_ACTIVITY 1-Toolkit Registry.isEventSet() method is called. 2-An Envelope EVENT_DOWNLOAD_USER_ACTIVITY is sent to the SIM.	1-Method returns true 2- Applet is triggered	
2	**Applet deregistration** Event= EVENT_EVENT_DOWNLOAD_USER_ACTIVITY Toolkit Registry.clearEvent()method is called Perform SIM initialization with all the facilities supported 1-An Envelope EVENT_DOWNLOAD_USER_ACTIVITY is sent to the SIM. a formatted sms pp envelope is sent to the sim Event= EVENT_EVENT_DOWNLOAD_USER_ACTIVITY Toolkit Registry.setEvent() method is called Perform SIM initialization with all the facilities supported 2-An Envelope EVENT_DOWNLOAD_USER_ACTIVITY is sent to the SIM.	1- Applet isn't triggered 2- Applet is triggered	

6.3.3.14.4 Test Coverage

CR Number	*Test Case Number*
CRRN1	1
CRRN2	2

6.3.3.15 EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE

Test Area Reference: FWK_APT_EDIS

6.3.3.15.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE once it has registered to this event and an Envelope Event DownLoad Idle Screen Available is received.

CRRN2: The applet is not triggered by the EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE once it has deregistered from this event.

6.3.3.15.2 Test Suite Files

Test Script: FWK_APT_EDIS_1.scr

Test Applet: FWK_APT_EDIS_1.java

Load Script: FWK_APT_EDIS_1.ldr

Cleanup Script: FWK_APT_EDIS_1.clr

Parameter File: FWK_APT_EDIS_1.par

6.3.3.15.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet registration to EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE and triggering** Applet is registered to the EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE and to EVENT_FORMATTED_SMS_PP_ENV Event= EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE 1-Toolkit Registry.isEventSet() method is called. 2-An Envelope EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE is sent to the SIM.	1-Method retuns true 2- Applet is triggered	
2	**Applet deregistration** Event=EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE Toolkit Registry.clearEvent()method is called Perform SIM initialization with all the facilities supported 1-An Envelope EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE is sent to the SIM. a formatted sms pp envelope is sent to the sim Event= EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE Toolkit Registry.setEvent() method is called Perform SIM initialization with all the facilities supported 2-An Envelope EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE is sent to the SIM.	1- Applet isn't triggered 2- Applet is triggered	

6.3.3.15.4 Test Coverage

CR Number	Test Case Number
CRRN1	1
CRRN2	2

6.3.3.16 EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS

Test Area Reference: FWK_APT_EDCR

6.3.3.16.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS once it has registered to this event and Envelope Event DownLoad Card Reader Status is received.

CRRN2: The applet is not triggered by the EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS once it has deregistered from this event.

6.3.3.16.2 Test Suite Files

Test Script:	FWK_APT_EDCR_1.scr
Test Applet:	FWK_APT_EDCR_1.java
Load Script:	FWK_APT_EDCR_1.ldr
Cleanup Script:	FWK_APT_EDCR_1.clr
Parameter File:	FWK_APT_EDCR_1.par

6.3.3.16.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet registration to EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS and triggering** Applet is registered to the EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS and to EVENT_FORMATTED_SMS_PP_ENV Event=EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS 1-Toolkit Registry.isEventSet() method is called. 2-An Envelope EVENT_DOWNLOAD_CARD_READER_STATUS is sent to the SIM.	1-Method returns true 2- Applet is triggered	
2	**Applet deregistration** Event= EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS Toolkit Registry.clearEvent()method is called Perform SIM initialization with all the facilities supported 1-An Envelope EVENT_DOWNLOAD_CARD_READER_STATUS is sent to the SIM. An Envelope formatted sms pp envelope is sent to the sim Event= EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS	1- Applet isn't triggered	

Id	Description	API/Framework Expectation	APDU Expectation
	Toolkit Registry.setEvent() method is called Perform SIM initialization with all the facilities supported 2-An Envelope EVENT_DOWNLOAD_CARD_READER_STATUS is sent to the SIM.	2- Applet is triggered	

6.3.3.16.4 Test Coverage

CR Number	Test Case Number
CRRN1	1
CRRN2	2

6.3.3.17 EVENT_UNRECOGNIZED_ENVELOPE

Test Area Reference: FWK_APT_EUEV

6.3.3.17.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_UNRECOGNIZED_ENVELOPE once it has registered to this event and an Unrecognized Envelope is received.

CRRN2: The applet is not triggered by the EVENT_UNRECOGNIZED_ENVELOPE once it has deregistered from this event.

6.3.3.17.2 Test Suite Files

Test Script: FWK_APT_EUEN_1.scr

Test Applet: FWK_APT_EUEN_1.java

Load Script: FWK_APT_EUEN_1.ldr

Cleanup Script: FWK_APT_EUEN_1.clr

Parameter File: FWK_APT_EUEN_1.par

6.3.3.17.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet registration to EVENT_UNRECOGNIZED_ENVELOPE and triggering** Applet is registered to the EVENT_UNRECOGNIZED_ENVELOPE and to EVENT_FORMMATTED_SMS_PP_ENV Event= EVENT_UNRECOGNIZED_ENVELOPE 1-Toolkit Registry.isEventSet() method is called. 2-An Envelope UNRECOGNIZED_ENVELOPE is sent to the SIM.	1-Method returns true 2- Applet is triggered	
2	**Applet deregistration** Event= EVENT_UNRECOGNIZED_ENVELOPE Toolkit Registry.clearEvent()method is called 1-An Envelope UNRECOGNIZED_ENVELOPE is sent to the SIM. a formatted sms pp envelope is sent to the sim Event= EVENT_UNRECOGNIZED_ENVELOPE Toolkit Registry.setEvent() method is called 2-An Envelope UNRECOGNIZED_ENVELOPE is sent to the SIM.	1- Applet isn't triggered 2- Applet is triggered	

6.3.3.17.4 Test Coverage

CR Number	*Test Case Number*
CRRN1	1
CRRN2	2

6.3.3.18 EVENT_STATUS_COMMAND

Test Area Reference: FWK_APT_ESTC

6.3.3.18.1 Conformance Requirement

Normal Execution

CRRN1: The applet is triggered by the EVENT_STATUS_COMMAND once it has registered to this event and a Status Command is received.

CRRN2: The applet is not triggered by the EVENT_STATUS_COMMAND once it has deregistered from this event.

6.3.3.18.2 Test Suite Files

Test Script: FWK_APT_ESTC_1.scr

Test Applet: FWK_APT_ESTC_1.java

FWK_APT_ESTC_2.java

FWK_APT_ESTC_3.java

Load Script: FWK_APT_ESTC_1.ldr

Cleanup Script: FWK_APT_ESTC_1.clr

Parameter File: FWK_APT_ESTC_1.par

6.3.3.18.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applets registration to EVENT_STATUS_COMMAND and triggering** Applet1 is registered to EVENT_STATUS_COMMAND using the requestPollInterval() command. Applet2 is registered to EVENT_STATUS_COMMAND using the RequestPollInterval() command. Applet3 is registered to EVENT_FORMATTED_SMS_PP_ENV. 1-A status command is sent to SIM Applet1 execution is finished Applet2 execution is finished	1- Applet1 is triggered. Applet2 is triggered. Applet 3 is not triggered	
2	**Applet deregistration and registration of the third applet to EVENT_STATUS_COMMAND. The STF shall not reply busy to a call control envelope** 1-A formatted sms pp envelope is sent to SIM Applet3 builds a DISPLAY TEXT. 2- ProactiveHandler.send() is called 3-A status command is sent to SIM. requestPollInteval with POLL_NO_DURATION is called Applet1 finalized requestPollInteval with POLL_NO_DURATION is called Applet2 finalized requestPollInterval() method is called. Applet3 execution finished.	1- Applet3 is triggered. 3- Applet1 is triggered. Applet2 is triggered.	2-A proactive command DISPLAY TEXT is sent and applet is suspended until the terminal response TERMINAL RESPONSE of DISPLAY TEXT is sent to the SIM

Id	Description	API/Framework Expectation	APDU Expectation
3	**Applet3 triggering** Perform SIM initialization with all the facilities supported Status command is sent to SIM.	Applet3 is triggered. (Applet1 and Applet2 are not triggered)	

6.3.3.18.4 Test Coverage

CR Number	Test Case Number
CRRN1	1,2
CRRN2	3

6.3.4 Proactive Command Sending by the STF

6.3.4.1 System Proactive Commands

Test Area Reference: FWK_PCS_SPCO

6.3.4.1.1 Conformance Requirements

Normal Execution

CRRN1: When a toolkit applet changes a menu entry of its registry object, the SIM Toolkit Framework shall dynamically* update the menu stored in the ME during the current card session

CRRN2: The STF shall use the data of the EFsume file when issuing the SET UP MENU proactive command.

CRRN3: For all EVENT_EVENT_DOWNLOAD_*: When a toolkit applet changes one or more of these requested events of its registry object, the STF shall dynamically* update the event list stored in the ME during the current card session by SET UP EVENT LIST proactive command.

*The STF shall send its system proactive command as soon as no proactive session is pending and all the applets registered to the current events have been triggered and have returned from the processToolkit method invocation.

6.3.4.1.2 Test Suite Files

Test Script: FWK_PCS_SPCO_1.scr

Test Applet: FWK_PCS_SPCO_1.java

Load Script: FWK_PCS_SPCO_1.ldr

Cleanup Script: FWK_PCS_SPCO_1.clr

Parameter File: FWK_PCS_SPCO_1.par

6.3.4.1.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Install Applet 1, Registered to the EVENT_EVENT_DOWNLOAD_MT_CALL and EVENT_EVENT_DOWNLOAD_LOCATION_STATUS** Perform SIM initialization with EVENT DOWNLOAD facilities supported~~**Install/install**~~		setEventList proactive command **[Event list]= '19020003'** or **'99020003'**

Id	Description	API/Framework Expectation	APDU Expectation
	~~Applet~~ ~~Constructor: ToolkitRegistry.setEvent~~		
2	~~TEST is triggered~~Trigger the applet by ENVELOPE (~~MENU SELECTION~~SMS_FORMATTED_PP) command Clear the events and build a display text command~~ToolkitRegistry.clearEvent~~		1. DISPLAY TEXT Proactive command 2. SET UP EVENT LIST Proactive command [CommandQualifier]= 00h

6.3.4.1.4 Test Coverage

CRR number	Test case number
N1	see: chapter 6.2.9.2, CRRN1, chapter 6.2.9.4, CRRN3, chapter 6.2.9.5 CRRN4, chapter 6.2.9.8 CRRN1
N2	see: chapter 6.2.9.2 CRRN1, chapter 6.2.9.8 CRRN1
N3	1,2

6.3.4.2 Interaction with GSM commands

Test Area Reference: FWK_PCS_IGCO

6.3.4.2.1 Conformance Requirements

Normal Execution

CRRN1: The STF shall process a GSM command even when a proactive command is pending (before and after the FETCH command until the terminal response). The STF shall answer with the SW1 and SW2 described in [3] and [4].

6.3.4.2.2 Test Suite Files

Test Script: FWK_PCS_IGCO_1.scr

Test Applet: FWK_PCS_IGCO_1.java

Load Script: FWK_PCS_IGCO_1.ldr

Cleanup Script: FWK_PCS_IGCO_1.clr

Parameter File: FWK_PCS_IGCO_1.par

6.3.4.2.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Interaction with GSM Commands after TERMINAL PROFILE in connection with FETCH and TERMINAL RESPONSE** Applet ~~calls initMenuEntry~~is registered to Menu Selection ~~ATR~~RST TERMINAL PROFILE (Profile : supports all facilities except: SET UP EVENT LIST, POLL INTERVAL and POLLING OFF) 1- System issues a proactive command SETUP_MENU		1- 91xx

Id	Description	API/Framework Expectation	APDU Expectation
	2- SELECT MF 3- GET RESPONSE (6 Bytes) 4- Failed SELECT File 5- FETCH 6- SELECT MF 7- GET RESPONSE (6 Bytes) 8- TERMINAL RESPONSE		2- 9Fxx 3- 91xx 4- 9404 5- Proactive Command: SETUP MENU 6- 9Fxx 7- 9000 8- 9000
2	**Interaction with GSM Commands after ENVELOPE (MENU SELECTION) in connection with FETCH and TERMINAL RESPONSE** Menu Entry ID = 0x01 1- SELECT MF 2- GET RESPONSE (6 Bytes) 3- Failed SELECT File 4- FETCH 5- SELECT MF 6- GET RESPONSE (6 Bytes) 7- TERMINAL RESPONSE		1- 9FXX 2- 91XX 3- 9404 4- Proactive Command: DISPLAY TEXT 5- 9FXX 6- 9000 7- 9000
3	**Interaction with GSM Commands after TERMINAL RESPONSE in proactive command session in connection with FETCH and TERMINAL RESPONSE** Menu Entry ID = 0x02 1- SELECT MF 2- GET RESPONSE (6 Bytes) 3- FETCH 4- SELECT MF 5- GET RESPONSE (6 Bytes) 6- Failed SELECT File 7- TERMINAL RESPONSE 8- SELECT MF 9- GET RESPONSE (6 Bytes) 10-Failed SELECT File 11-FETCH 12-SELECT MF 13-GET RESPONSE (6 Bytes) 14-TERMINAL RESPONSE		1- 9FXX 2- 91XX 3- Proactive Command: DISPLAY TEXT 4- 9FXX 5- 9000 6- 9404 7- 9000 8- 9FXX 9- 91XX 10-9404 11-Proactive Command: DISPLAY TEXT 12-9FXX 13-9000 14-9000

6.3.4.2.4 Test Coverage

CRR number	Test case number
N1	1,2,3

6.3.5 Exception Handling

6.3.5.1 Hide Exceptions from the ME

Test Area Reference: FWK_EXH_HEME

6.3.5.1.1 Conformance Requirements

Normal Execution

CRRN1 : A toolkit applet may throw an exception, but this error will not be sent to the ME.

* Because the behaviour of the SIM is not exactly defined for the above CRRN, there are no tests defined here yet.

6.3.5.2 Interaction with Multiple Triggering

Test Area Reference: FWK_EXH_IMTG

6.3.5.2.1 Conformance Requirements

Normal Execution:

CRRN1 : An exception thrown by a toolkit applet, will not influence toolkit applets registered to the same event

6.3.5.2.2 Test Suite Files

Test Script: FWK_EXH_IMTGR _1.scr

Test Applet: : FWK_EXH_IMTGR _1.java

Load Script: : FWK_EXH_IMTGR_1.ldr

Cleanup Script: FWK_EXH_IMTGR_1.clr

Parameter File: FWK_EXH_IMTGR_1.par

6.3.5.2.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
0	**Load/install 2 toolkit applets registered to EVENT_STATUS_COMMAND, EVENT_PROFILE_DOWNLOAD, EVENT_UNRECOGNISED_ENVELOPE, EVENT_EVENT_DOWNLOAD_MT_CALL, EVENT_UNFORMATTED_SMS_PP_ENV, EVENT_UNFORMATTED_SMS_PP_UPD, EVENT_UNFORMATTED_SMS_CB** `applet1: Priority= 0x01,` `applet2: Priority= 0x02,` `(i.e. applet1 is triggered before applet2)`		
1	**STATUS_COMMAND is sent** Applet 1 is triggered Applet 2 is triggered	Applet1: `NullPointerException` is thrown	
2	**PROFILE_DOWNLOAD is sent** Applet 1 is triggered Applet 2 is triggered	Applet1: `NullPointerException` is thrown	
3	**UNRECOGNISED_ENVELOPE is sent**		

Id	Description	API/Framework Expectation	APDU Expectation
	Applet 1 is triggered Applet 2 is triggered	Applet1: `NullPointerException` is thrown	
4	**EVENT_DOWNLOAD_MT_CALL is sent** Applet 1 is triggered Applet 2 is triggered	Applet1: `NullPointerException` is thrown	
5	**UNFORMATTED_SMS_PP_ENV is sent** Applet 1 is triggered Applet 2 is triggered	Applet1: `NullPointerException` is thrown	
6	**UNFORMATTED_SMS_PP_UPD is sent** Applet 1 is triggered Applet 2 is triggered	Applet1: `NullPointerException` is thrown	
7	**UNFORMATTED_SMS_CB is sent** Applet 1 is triggered Applet 2 is triggered	Applet1: `NullPointerException` is thrown	

6.3.5.2.4 Test Coverage

CRR Number	Test Case Number
CRRN1	1,2,3,4,5,6,7

6.3.6 Framework Security Management

Security Parameters

The table that follows contains the security parameters that shall be used when the 03.48 security is required in the test cases developed in the current section.

Parameter	Value in hexadecimal
KIC	11
KID	11
CNTR	00 00 00 00 01
Key for ciphering	01 41 42 7F DA E8 91 A7
Key for RC/CC/DS	01 23 45 67 89 AB CD EF

If a parameter is not listed explicitly in the above table, the default values of section 4.7.3.1 apply.

6.3.6.1 Input Data

Test Area Reference: FWK_FWS_INDA

6.3.6.1.1 Conformance Requirements

Normal Execution

CRRN1: If the SIM receives an envelope APDU containing an SMS_DATADOWNLOAD BER TLV formatted according to GSM03.48, the SIM Toolkit Framework shall verify the GSM03.48 security of the SMS TPDU.

CRRN2: The toolkit applet will only be triggered if the TAR is known and the security verified.

6.3.6.1.2 Test Area Files

Test Script:	FWK_FWS_INDA_1.scr
Test Applet:	FWK_FWS_INDA_1.java
	FWK_FWS_INDA_2.java
Load Script:	FWK_FWS_INDA_1.ldr
Cleanup Script:	FWK_FWS_INDA_1.clr
Parameter File:	FWK_FWS_INDA_1.par

6.3.6.1.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Framework checks the Cryptographic checksum and deciphers the dataApplet1 is loaded and installed** ~~**Load and install applet 1**~~ 1-Envelope(SMS-PP) 03.48 formatted is sent to the SIM with this features:~~Envelope(SMS-PP) 03.48 formatted~~ Ciphering; Cryptographic checksum; No proof of receipt; Data = 01	The applet is triggered.	
2	**Framework checks the Cryptographic checksum and deciphers the data** Applet2 is installed 1-Envelope(SMS-PP) 03.48 formatted is sent to the SIM with this features:~~**Envelope(SMS-PP) 03.48 formatted**~~ Ciphering; Cryptographic checksum; No proof of receipt; TAR of Applet 1 Data = 02 2-Envelope(SMS-PP) 03.48 formatted is sent to the SIM with this features:~~**Envelope(SMS-PP) 03.48 formatted**~~ No ciphering; No cryptographic checksum; No proof of receipt; TAR of Applet 2 Data = 03	This Envelope(SMS-PP) triggers Applet 1 This Envelope(SMS-PP) triggers Applet 2	The SIM answers to the Envelope with status words 9000 The SIM answers to the Envelope with status words 9000
3	**Envelope(SMS-PP) 03.48 formatted with wrong cryptographic checksum**	No applet is triggered	The SIM answers to the Envelope with status words

Id	Description	API/Framework Expectation	APDU Expectation
	No ciphering; Wrong Cryptographic checksum; No proof of receipt; TAR of Applet 1 Data = 04		9000

6.3.6.1.4 Test Coverage

CRR Number	Test Case Number
CRRN1	1,2,3
CRRN2	3

6.3.6.2 Output Data

Test Area Reference: FWK_FWS_OUDA

6.3.6.2.1 Conformance Requirements

Normal Execution

CRRN1: The SIM Toolkit Framework shall secure and send the response packet.

6.3.6.2.2 Test Area Files

Test Script: FWK_FWS_OUDA_1.scr

Test Applet: FWK_FWS_OUDA_1.java

~~FWK_FWS_OUDA_2.java~~

Load Script: FWK_FWS_OUDA_1.ldr

Cleanup Script: FWK_FWS_OUDA_1.clr

Parameter File: FWK_FWS_OUDA_1.par

6.3.6.2.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Envelope(SMS-PP) 03.48 formatted** Ciphering; Cryptographic checksum; proof of receipt response shall be sent using SMS-Deliver-Report; no security applied to proof of receipt Data in plain text = “APPLET1”	The applet is triggered and sends a “Display Text” proactive command with the data received in the Envelope.	The SIM answers to the Envelope with status words 9Fxx and a PoR is retrieved with a GetResponse command. The PoR has no application data. The SIM answers to the Get Response command with status words 91xx to issue a Display Text “APPLET1”.
2	**Envelope(SMS-PP) 03.48 formatted** Ciphering; Cryptographic checksum; proof of receipt response shall be sent using SMS-Deliver-Report; no security applied to proof of receipt Data in plain text = “APPLET1”	The applet posts application data. It does not call the ProactiveHandler.send() method	The SIM answers to the Envelope with status words 9Fxx and a PoR is retrieved with a GetResponse command. The PoR has the application data posted by the application. The SIM answers to the Get Response command with status words 9000.
3	**Envelope(SMS-PP) 03.48 formatted** Ciphering;	The applet posts application data and calls the	The SIM answers to the Envelope with status words

Id	Description	API/Framework Expectation	APDU Expectation
	Cryptographic checksum; proof of receipt response shall be sent using SMS-Deliver-Report; no security applied to proof of receipt Data in plain text = “TEST”	ProactiveHandler.send() method to send a “Display Text” proactive command with the data received in the Envelope.	9Fxx and a PoR is retrieved with a GetResponse command. The PoR has the application data posted by the application. The SIM answers to the Get Response command with status words 91xx to issue the Display Text “TEST”.
4	**Envelope(SMS-PP) 03.48 formatted** Ciphering; Cryptographic checksum; proof of receipt response shall be sent using SMS-Deliver-Report; proof of receipt shall be ciphered Data in plain text = “TEST”	The applet posts application data and calls the ProactiveHandler.send() method to send a “Display Text” proactive command with the data received in the Envelope.	The SIM answers to the Envelope with status words 9Fxx and a PoR is retrieved with a GetResponse command. The PoR has the application data posted by the application. The SIM answers to the Get Response command with status words 91xx to issue the Display Text “TEST”.
5	**Envelope(SMS-PP) 03.48 formatted** The Terminal Profile command shall be issued with the facility “'9EXX' response code for SIM data download error” enabled The Envelope(SMS-PP) formatted has to be issued with the following features: No ciphering; Wrong Cryptographic checksum; proof of receipt response shall be sent using SMS-Deliver-Report; no security applied to proof of receiptData in plain text = “TEST”	No applet is triggered	The SIM answers to the Envelope with status words 9Exx and a PoR is retrieved with a GetResponse command. The Response Status Code Octet shall be ‘01’.

6.3.6.2.4 Test Coverage

CRR Number	Test Case Number
CRRN1	1,2,3,4,5

6.3.7 Envelope Response Posting

6.3.7.1 EVENT_CALL_CONTROL_BY_SIM

Test Area Reference: FWK_ERP_ECCN

6.3.7.1.1 Conformance Requirements

Normal Execution

CRRN1: The SIM Toolkit Framework can’t reply busy when an Envelope(Call Control) is sent to the SIM.

6.3.7.1.1 Test Area Files

Test Script: FWK_ERP_ECCN_1.scr

Test Applet: FWK_ERP_ECCN_1.java

FWK_ERP_ECCN_2.java

FWK_ERP_ECCN_3.java

Load Script: FWK_ERP_ECCN_1.ldr

Cleanup Script: FWK_ERP_ECCN_1.clr

Parameter File: FWK_ERP_ECCN_1.par

6.3.7.1.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet 1 is registered on the EVENT_CALL_CONTROL_BY_SIM, Applet2 is registered and triggered on the EVENT_MENU_SELECTION.** 1-Applet2 invokes the method send()and no fetch is performed 2-Envelope(Call Control) is sent to the SIM 3-Applet1 calls the method EnvelopeResponseHandler.postASBERTLV() to change any incoming dialling number into +11 22 33 44. 4-A Fetch command is sent to the SIM 5-A Terminal Response command is sent to the SIM 6-Delete applet1 & applet2 7-Install applet3 ~~It calls the method EnvelopeResponseHandler.postASBERTLV() to change any incoming dialling number into +11 22 33 44.~~ **~~Applet 2 is registered and triggered on the EVENT_MENU_SELECTION.~~** ~~It is suspended (the method send() has been called and no fetch has been performed)~~ **~~Envelope(Call Control) is sent to the SIM~~** **~~A Fetch command is sent to the SIM.~~**	Applet2 is suspended ~~The~~ Applet ~~1~~ is triggered. ~~The~~ Applet ~~2~~'s execution shall continue.	The SIM answer 9Fxx to the Envelope(Call Control) The dialling number is retrieved with a GetResponse command. The SIM answers to the Get Response command with status words 91xx.

Id	Description	API/Framework Expectation	APDU Expectation
2	**Applet 3 is registered on both the events EVENT_CALL_CONTROL_BY_SIM and EVENT_MENU_SELECTION.** 1-Envelope Menu Selection is sent to the SIM. 2-Applet3 invokes the method send()and no fetch is performed) 3-Envelope(Call Control) is sent to the SIM 4-Applet3 calls the method EnvelopeResponseHandler.postASBERTLV() to change any incoming dialling number into +11 22 33 44. 5-A Fetch command is sent to the SIM 6-A Terminal Response command is sent to the SIM ~~It calls the method EnvelopeResponseHandler.postASBERTLV() to change any incoming dialling number into +11 22 33 44.~~ ~~**Applet 3 is triggered on the EVENT_MENU_SELECTION and is suspended on the send() method.**~~ ~~**Envelope(Call Control) is sent to the SIM**~~ ~~**A Fetch command is sent to the SIM.**~~	Applet3 is triggered on the EVENT_MENU_SELECTION Applet3 is suspended on the send() method ~~The a~~Applet ~~3~~ is triggered on the EVENT_CALL_CONTROL_BY_SIM. The A~~a~~pple3t's execution shall continue.	The SIM answer 9Fxx to the Envelope(Call Control) The dialling number is retrieved with a GetResponse command. The SIM answers to the Get Response command with status words 91xx.

6.3.7.1.4 Test Coverage

CRR Number	Test Case Number
CRRN1	1,2

6.3.7.2 EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM

Test Area Reference: FWK_ERP_EMCN

6.3.7.2.1 Conformance Requirements

Normal Execution

CRRN1: The SIM Toolkit Framework can't reply busy when an Envelope(MO-Short Message Control) is sent to the SIM.

6.3.7.2.2 Test Area Files

Test Script: FWK_ERP_EMCN_1.scr

Test Applet: FWK_ERP_EMCN_1.java

FWK_ERP_EMCN_2.java

FWK_ERP_EMCN_3.java

Load Script: FWK_ERP_EMCN_1.ldr

Cleanup Script: FWK_ERP_EMCN_1.clr

Parameter File: FWK_ERP_EMCN_1.par

6.3.7.2.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Applet 1 is registered on the EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM; Applet2 is registered and triggered on the EVENT_MENU_SELECTION.** . 1-Applet2 invokes the method send()and no fetch is performed) 2-Envelope(MO-SM control) is sent to the SIM 3-Applet1 calls the method EnvelopeResponseHandler.postASBERTLV() to change any incoming TP_Destination_Address and any RP_Destination_Address of the Service Center into +11 22 33 44 4-A Fetch command is sent to the SIM 5-A Terminal Response command is sent to the SIM 6-Delete applet1 & applet2 7-Install applet3~~.~~ ~~It calls the method EnvelopeResponseHandler.postASBERTLV() to change any incoming TP_Destination_Address into +11 22 33 44.~~ ~~**Applet 2 is registered and triggered on the EVENT_MENU_SELECTION.**~~ ~~It is suspended (the method send() has been called and no fetch has been performed)~~ ~~**An Envelope(MO-Short Message Control) is sent to the SIM**~~ ~~**A Fetch command is sent to the SIM.**~~	Applet2 is suspended ~~The a~~Applet 1 is triggered. The A~~a~~pplet's execution shall continue.	The SIM answers 9Fxx to the Envelope(MO-Short Message Control) The TP_Destination_Address is retrieved with a GetResponse command. The SIM answers to the Get Response command with status words 91xx.

Id	Description	API/Framework Expectation	APDU Expectation
2	**Applet 3 is registered on both the events EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM and EVENT_MENU_SELECTION.** 1-Applet3 invokes the method send()and no fetch is performed) 2-Envelope(MO-SM control) is sent to the SIM 3-Applet3 calls the method EnvelopeResponseHandler.postASBERTLV() to change any incoming TP_Destination_Address and any RP_Destination_Address of the Service Center into +11 22 33 44. 4-A Fetch command is sent to the SIM 5-A Terminal Response command is sent to the SIM ~~It calls the method EnvelopeResponseHandler.postASBERTLV() to change any incoming TP_Destination_Address into +11 22 33 44.~~ **~~Applet 3 is triggered on the EVENT_MENU_SELECTION and is suspended on the send() method.~~** **~~An Envelope(MO-Short Message Control) is sent to the SIM~~** **~~A Fetch command is sent to the SIM.~~**	Applet 3 is suspended on the send() method ~~The a~~Applet3 is triggered on the EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM. The ~~applet's~~ Applet3's execution shall continue.	The SIM answers 9Fxx to the Envelope(MO-Short Message Control) The TP_Destination_Address is retrieved with a GetResponse command. The SIM answers to the Get Response command with status words 91xx.

6.3.7.2.4 Test Coverage

CRR Number	Test Case Number
CRRN1	1,2

6.3.7.3 EVENT_UNRECOGNIZED_ENVELOPE

Test Area Reference: FWK_ERP_EUEN

6.3.7.3.1 Conformance Requirements

Normal Execution

CRRN1: The EnvelopeResponseHandler is available for the EVENT_UNRECOGNIZED_ENVELOPE.

6.3.7.3.2 Test Area Files

Test Script: FWK_ERP_EUEN_1.scr

Test Applet: FWK_ERP_EUEN_1.java

Load Script: FWK_ERP_EUEN_1.ldr

Cleanup Script: FWK_ERP_EUEN_1.clr

Parameter File: FWK_ERP_EUEN_1.par

6.3.7.3.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**An applet triggered on the EVENT_UNRECOGNIZED_ENVELOPE calls the EnvelopeResponseHandler.post() method**	The post() method returns no exception	The SIM answers to the Envelope with status words 9Fxx. The data retrieved with the GetResponse command are the ones posted by the applet.

6.3.7.3.4 Test Coverage

CRR Number	Test Case Number
CRRN1	1

6.3.8 Toolkit Installation

6.3.8.1 Timers Allocation

Test Area Reference: FWK_TIN_TMAL

6.3.8.1.1 Conformance Requirements

Normal execution

CRRN1: One toolkit applet can register to several timers, but a timer can only be allocated to one toolkit applet.

Context errors

CRRC1 : Allocated timers shall not exceed the maximum number of timers allowed for this applet instance defined during installation.

CRRC2 : The total number of timers allocated for all the applets shall not exceed 8.

6.3.8.1.2 Test suite files

Test Script: FWK_TIN_TMAL_1.scr

Test Applet: ~~1.~~ FWK_TIN_TMAL_1.java

~~2.~~ FWK_TIN_TMAL_2.java

~~3.~~ FWK_TIN_TMAL_3.java

Load Script: FWK_TIN_TMAL_1.ldr

Cleanup Script: FWK_TIN_TMAL_1.clr

Parameter File: FWK_TIN_TMAL_1.par

6.3.8.1.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**More than 8 timers at the instantiation of applet1 : check that applet1 is not installed or that it is not possible to allocate more than 8 timers.** Install for install of applet1 with maximum 9 timers allocated. applet1 is triggered : we allocate 9 timers applet1 is selected	Shall throw a ToolkitException with reason NO_TIMER_AVAILABLE only on the 9th allocateTimer()	The SIM answers to the Envelope with status words 90 00 2 behaviours may be expected : 1. applet1 is not found, status word 6X XX 2. applet1 has been installed and only 8 timers are allocated
	Reset the card and delete instance of applet1		
2	**Good installation of applet2** Install for install of applet2 (maximum 4 timers allocated).		The SIM answers to the Envelope with status words 90 00
3	**Allocate 4 timers Applet2**	No exception shall be thrown.	
4	**Allocate one more timer Applet2**	Shall throw a ToolkitException with reason NO_TIMER_AVAILABLE	
5	**Good installation of applet3** Install for install of applet3 (maximum 8 timers allocated).		The SIM answers to the Envelope with status words 90 00
6	**Allocate 4 timers Applet3**	No exception shall be thrown.	
7	**Allocate one more timer Applet3**	Shall throw a ToolkitException with reason NO_TIMER_AVAILABLE	
8	**Check that each timerId (allocated by applet2 and applet3) is between 1 and 8 and is different from each other**		

6.3.8.1.4 Test Coverage

CRR number	Test case number
N1	2,3,8
C1	1, 7
C2	4,5,6

6.3.8.2 Item Identifier

Test Area Reference: FWK_TIN_ITID

6.3.8.2.1 Conformance Requirements

Normal execution

CRRN1: If the requested item identifier in the range [1-127] is not already allocated, then this item identifier shall be allocated to the current applet.

CRRN2: If the requested item identifier is '00', the card shall take the first free value in the range [128,255].

Parameters error

CRRP1: If the requested item identifier is in the range [128,255], then the card shall reject the install command.

Context errors

CRRC1: If the requested item identifier in the range [1-127] is already allocated, then the card shall reject the install command.

6.3.8.2.2 Test suite files

Test Script: FWK_TIN_ITID_1.scr

Test Applet: FWK_TIN_ITID _1.java

FWK_TIN_ITID _2.java

FWK_TIN_ITID _3.java

Load Script: FWK_TIN_ITID _1.ldr

Cleanup Script: FWK_TIN_ITID _1.clr

Parameter File: FWK_TIN_ITID _1.par

6.3.8.2.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Bad installation of applet1** Install for install of applet1.The following parameters item Id equal to 128 applet1 is selected		applet1 is not found, status word 6X XX
2	**Good installation of applet1** Install for install of applet1. item Id = 1 for the first menu and 127 for the second one A Terminal Profile is sent to the card with only PROFILE_DOWNLOAD, SMS_PP_DOWNLOAD, MENU_SELECTION, SET_UP_MENU and COMMAND_RESULT facilities.		The SIM answers to the Envelope with status words 91xx to send back to the ME the 2 new menus. The menus are (position/itemId/text) 01/01/menu11 02/127/menu12
3	**Bad installation of applet2 Item identifier already allocated** Install for install of applet2. item Id = 127 applet2 is selected		applet2 is not found, status word 6X XX
4	**Good installation of applet2** Install for install of applet2. item Id = 0		The SIM answers to the Envelope with status words 91xx to send back to the ME the 3 menus. The menus are 01/01/menu11 02/127/menu12 03/128/menu21
5	**Good installation of applet3** Install for install of applet3. item Id = 0		The SIM answers to the Envelope with status words 91xx to send back to the ME the 4 menus. The menus are 01/01/menu11 02/127/menu12 03/128/menu21 04/129/menu31
6	**Good delete and installation of applet2** Delete instance of applet2 Perform a RESET and a Terminal Profile with the facilities of PROFILE_DOWNLOAD, SMS-		The SIM answers to the ~~Envelope~~ Terminal Profile with status words 91xx to send back to the ME the 3 menus.

Id	Description	API/Framework Expectation	APDU Expectation
	PP_DATA_DOWNLOAD, MENU_SELECTION, COMMAND_RESULT and SET_UP_MENU		The menus are 01/01/menu11 02/127/menu12 03/129/menu31
	Install for install of applet2. item Id = 0		The SIM answers to the Envelope with status words 91xx to send back to the ME the 4 menus. The menus are 01/01/menu11 02/127/menu12 03/128/menu21 04/129/menu31

6.3.8.2.4 Test Coverage

CRR number	Test case number
N1	2
N2	4,5,6
P1	1
C1	3

6.3.8.3 Item Position

Test Area Reference: FWK_TIN_ITPO

6.3.8.3.1 Conformance Requirements

Normal execution

CRRN1: The position of the new menu entries is an absolute position among the existing ones.

CRRN2: If the position identifier is 00h, the menu shall have the last position

6.3.8.3.2 Test suite files

Test Script: FWK_TIN_ITPO_1.scr

Test Applet: FWK_TIN_ITPO_1.java

FWK_TIN_ITPO_2.java

FWK_TIN_ITPO_3.java

Load Script: FWK_TIN_ITPO_1.ldr

Cleanup Script: FWK_TIN_ITPO_1.clr

Parameter File: FWK_TIN_ITPO_1.par

6.3.8.3.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Installation of applet1** Perform Install for install of applet1.Position/ItemId 01/01 0~~1~~2/02 A Terminal Profile is sent to the card		The menus are (position/itemId/text) 01/01/menu11 02/02/menu12
2	**Installation of applet2** Perform Install for install of applet2. Position/ItemId 03/03 04/04		The SIM answers to the Envelope with status words 91xx to send back to the ME the 4 menus. The menus are (position/itemId/text) 01/01/menu11 02/02/menu12 03/03/menu21 04/04/menu22
3	**Installation of applet3** Perform Install for install of applet3. Position/ItemId 00/05		The SIM answers to the Envelope with status words 91xx to send back to the ME the 5 menus. The menus are (position/itemId/text) 01/01/menu11 02/02/menu12 03/03/menu21 04/04/menu22 05/05/menu31

6.3.8.3.4 Test Coverage

Note : As Item Position management is not fully specified in the [7] or [8] all possible tests cannot be performed.

CRR number	Test case number
N1	1,2
N2	3

6.3.8.4~~2~~ Maximum Text Length for a menu entry

Test Area Reference: FWK_TIN_MLME

6.3.8.4~~2~~.1 Conformance Requirements

Normal execution

CRRN1: The maximum length of item text string is defined at the installation of the toolkit applet.

Parameters errors

CRRP1: If initMenuEntry length parameter is greater than the allocated space (Maximum Text Length for a menu entry), then a ToolkitException ALLOWED_LENGTH_EXCEEDED is thrown.

CRRP2: If changeMenuEntry length parameter is greater than the allocated space (Maximum Text Length for a menu entry), then a ToolkitException ALLOWED_LENGTH_EXCEEDED is thrown.

6.3.8.42.2 Test suite files

Test Script: FWK_TIN_MLME_1.scr

Test Applet: FWK_TIN_MLME_1.java

Load Script: FWK_TIN_MLME_1.ldr

Cleanup Script: FWK_TIN_MLME_1.clr

Parameter File: FWK_TIN_MLME_1.par

6.3.8.42.31 Test Procedure

Id	Description	API / Framework Expectation	APDU Expectation
10	**Installation of applet with 2 menus not exceeding the maximum text length** Install one applet with 2 menu entries allowed and max. text length equal to 10. initMenuEntry defined at the install (install) command MenuEntry = "MenuEntry1", "MenuEntry2" "Toolkitap1" Offset = 0 Length = 10 NextAction = '00' HelpSupported = false IconQualifier = '00' IconIdentifier = 0		
21	**initMenuEntry with a too large length** initMenuEntry with length equal to 11 MenuEntry = " MenuEntry03Toolkitap02" Offset = 0 Length = 11 NextAction = '00' HelpSupported = false IconQualifier = '00' IconIdentifier = 0	ToolkitException ALLOWED_LENGTH_EXCEEDED is thrown	
32	**initMenuEntry with a right length** initMenuEntry with length parameter equal to 10 MenuEntry = " MenuEntry3Toolkitap2" Offset = 0 Length = 10 NextAction = '00' HelpSupported = false IconQualifier = '00' IconIdentifier = 0		a SET UP MENU (2 items) is issued with TLV item length equal to 11 (Identifier + Text string of item)
4	**changeMenuEntry with a right length** Applet1 is triggered by a EVENT_MENU_SELECTION. changeMenuEntry of menu 1, with length parameter equal to 10 Id = '01' MenuEntry = "MenuEntry4" Offset = 0 Length = menuEntry.length NextAction = 0 HelpSupported = false IconQualifier = 0 IconIdentifier = 0 Return from processToolkit		a SET UP MENU (2 items) is issued with TLV item length equal to 11 (Identifier + Text string of item)
53	**changeMenuEntry with a too large length**	ToolkitException ALLOWED_LENGTH_EXCEEDED	Shall not receive a SET UP MENU different from the

Id	Description	API / Framework Expectation	APDU Expectation
	Applet 1 is triggered by a EVENT_MENU_SELECTION. ChangeMenuEntry of menu 1, with length parameter equal to 11 Id = '02' MenuEntry = " MenuEntry05Toolkitap04" Offset = 0 Length = menuEntry.length NextAction = 0 HelpSupported = false IconQualifier = 0 IconIdentifier = 0 Return from processToolkit	is thrown	previous one

6.3.8.42.43 Test Coverage

CRR number	Test case number
CRRN1	1, 32, 4
CRRP1	21
CRRP2	53

6.3.8.53 Maximum number of menu entries

Test Area Reference: FWK_TIN_NBME

6.3.8.53.1 Conformance Requirements

Normal execution

CRRN1: The maximum number of menu entries is defined at the installation of the toolkit applet and can be the maximum number of successful invocations of the method initMenuEntry .

Parameters errors

CRRP1: If the menu entry cannot be initialised (e.g. no more item data in applet loading parameter), a ToolkitException with the REGISTRY_ERROR reason code is thrown.

6.3.8.53.2 Test suite files

Test Script: FWK_TIN_NBME_1.scr

~~FWK_TIN_NBME_2.scr~~

Test Applet: FWK_TIN_NBME_1.java

FWK_TIN_NBME_2.java

Load Script: FWK_TIN_NBME_1.ldr

~~FWK_TIN_NBME_2.ldr~~

Cleanup Script: FWK_TIN_NBME_1.clr

~~FWK_TIN_NBME_2.clr~~

Parameter File: FWK_TIN_NBME_1.par

~~FWK_TIN_NBME_2.par~~

6.3.8.53.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Installation of applet with 3 menus**	No Exception is thrown	

Id	Description	API/Framework Expectation	APDU Expectation
	Install (install) applet with max. number of menu entry is '3', defined at the install (install) command. initMenuEntry for each menu entry allowed (3 times) MenuEntry = "menu1", "menu2", "menu3" Offset = 0 Length = 5 NextAction = '00' HelpSupported = false IconQualifier = '00' IconIdentifier = 0		
2	**init of a 4th menu** initMenuEntry one more time MenuEntry = "menu4" Offset = 0 Length = 5 NextAction = '00' HelpSupported = false IconQualifier = '00' IconIdentifier = 0	ToolkitException REGISTRY_ERROR is thrown	SET UP MENU (3 items) is issued with TLV item length equal to 6 (Identifier + Text string of item)
3	**Installation of 2nd applet with 0 menu** Install (install) another ~~instance of the same~~ applet, with max. number of menu entry is '0', defined at the install (install) command. initMenuEntry once MenuEntry = "menu1" Offset = 0 Length = 5 NextAction = '00' HelpSupported = false IconQualifier = '00' IconIdentifier = 0	ToolkitException REGISTRY_ERROR is thrown	Shall not receive a SET UP MENU different from the previous one

6.3.8.5~~3~~.4 Test Coverage

CRR number	Test case number
CRRN1	1
CRRP1	2, 3

6.3.8.6~~4~~ Access Domain

Test Area Reference: FWK_TIN_ACDO

6.3.8.6~~4~~.1 Conformance Requirements

Normal execution

CRRN1: The Access Domain parameter indicates the mechanism used to control the applet instance access to the GSM file System ('00' means full access to the GSM File System, 'FF' means no access to the GSM File System).

Parameters errors

CRRP1: If the Access Domain Parameter requested is not supported, the card shall return the Status Word '6A80', incorrect parameters in data field, to the Install(Install) command.

CRRP2: If an applet with Access Domain Parameter 'FF' (i.e. No Access to the GSM File System) tries to access a GSM file (e.g. invoke the updateBinary(..) method) the framework shall throw a SIMViewException with a AC_NOT_FULFILLED reason.

6.3.8.64.2 Test suite files

Test Script: FWK_TIN_ACDO_1.scr

Test Applet: FWK_TIN_ACDO_1.java

FWK_TIN_ACDO_2.java

FWK_TIN_ACDO_3.java

Load Script: FWK_TIN_ACDO_1.ldr

Cleanup Script: FWK_TIN_ACDO_1.clr

Parameter File: FWK_TIN_ACDO_1.par

6.3.8.64.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
0	Install (install) applet1 with: - Length of Access Domain field value is '1' - Access Domain Parameter value is '00' (full access to the GSM File System) Install (install) applet2 with: - Length of Access Domain field value is '1' - Access Domain Parameter value is 'FF' (No access to the GSM File System) Install (install) applet3 with: - Length of Access Domain field value is '1' - Access Domain Parameter value is '00' (full access to the GSM File System)		
1	**readBinary/readRecord method with full Access Domain Parameter** 1- Select EF-TARU file whose Read access condition is ALWAYS Perform the readBinary method: fileOffset = 0 resp = babRead[] respOffset = 0 respLength = 3 2- Select EF-SMS file whose Read access condition is CHV1 Perform the readRecord method: ~~fileOffset = 0~~ recNumber = 1 mode = REC_ACC_MODE_ABSOLUTE_CURRENT recOffset = 0 resp = babRead[] respOffset = 0 respLength = 3 3- Select EF-TRAC file whose Read access condition is CHV2 Perform the readBinary method: fileOffset = 0 resp = babRead[] respOffset = 0 respLength = 3 4- Select EF-SUME file Read access condition is ADM0 Perform the readBinary method: fileOffset = 0	1 to 4- no exception is thrown 5- SIMViewException AC_NOT_FULFILLED is thrown	

Id	Description	API/Framework Expectation	APDU Expectation
	resp = babRead[] respOffset = 0 respLength = 3 5- Select EF-TNR file whose Read access condition is NEVER Perform the readBinary method: fileOffset = 0 resp = babRead[] respOffset = 0 respLength = 3		
2	**updateBinary/updateRecord method with full Access Domain Parameter** For each case, send an Envelope that triggers the applet with the EVENT_UNFORMATTED_SMS_PP_ENV event. 1- Select EF-TNR file whose Update access condition is ALWAYS Perform the updateBinary method: fileOffset = 0 resp = babUpdate[FFFFFF] respOffset = 0 respLength = 3 2- Select EF-SMS file whose Update access condition is CHV1 Perform the updateRecord method: recNumber = 1 mode = REC_ACC_MODE_ABSOLUTE_CURRENT recOffset = 0fileOffset = 0 resp = babUpdate[] respOffset = 0 respLength = 3 3- Select EF-FDN file whose Update access condition is CHV2 Perform the updateBinary method: recNumber = 1 mode = REC_ACC_MODE_ABSOLUTE_CURRENT recOffset = 0fileOffset = 0 resp = babUpdate[] respOffset = 0 respLength = 3 4- Select EF-SUME file Update access condition is ADM0 Perform the updateBinary method: fileOffset = 0 resp = babUpdate[] respOffset = 0 respLength = 3 5- Select EF-TNU file whose Update access condition is NEVER Perform the updateBinary method: fileOffset = 0 resp = babUpdate[] respOffset = 0 respLength = 3	1 to 4- no exception is thrown 5- SIMViewException AC_NOT_FULFILLED is thrown	
3	**invalidate method with full Access Domain Parameter** 1- Select EF-TNR file whose Invalidate access condition is ALWAYS Perform the invalidate method 2- Select EF-TIAC file whose Invalidate access condition is CHV1 Perform the invalidate method	1 to 4- no exception is thrown 5- SIMViewException AC_NOT_FULFILLED is thrown	

Id	Description	API/Framework Expectation	APDU Expectation
	3- Select EF-ADN file whose Invalidate access condition is CHV2 Perform the invalidate method 4- Select EF-SUME file Invalidate access condition is ADM0 Perform the invalidate method 5- Select ~~EFTNU~~ EF-CNIV file whose Invalidate access condition is NEVER Perform the invalidate method		
4	**rehabilitate method with full Access Domain Parameter** 1- Select EF-TNR file whose Rehabilitate access condition is ALWAYS Perform the rehabilitate method 2- Select EF-IMSI file whose Rehabilitate access condition is CHV1 Perform the rehabilitate method 3- Select EF-ADN file whose Rehabilitate access condition is CHV2 Perform the rehabilitate method 4- Select EF-SUME file Rehabilitate access condition is ADM0 Perform the rehabilitate method 5- Select ~~EFTNU~~ EF-CNRI file whose Rehabilitate access condition is NEVER Perform the rehabilitate method	1 to 4- no exception is thrown 5- SIMViewException AC_NOT_FULFILLED is thrown	
5	**increase method with full Access Domain Parameter** 1- Select EF-CNU file whose Increase access condition is ALWAYS Perform the increase method: incr = abIncreaseValue[] incrOffset = 0 resp = abRead[] respOffset = 0 2- Select EF-ACM file whose Increase access condition is CHV1 Perform the increase method: incr = abIncreaseValue[] incrOffset = 0 resp = abRead[] respOffset = 0 3- Select EF-CIAC file whose Increase access condition is CHV2 Perform the increase method: incr = abIncreaseValue[] incrOffset = 0 resp = abRead[] respOffset = 0 4- Select EF-CIAA file Increase access condition is ADM0 Perform the increase method: incr = abIncreaseValue[] incrOffset = 0 resp = abRead[] respOffset = 0	1 to 4- no exception is thrown 5- SIMViewException AC_NOT_FULFILLED is thrown	

Id	Description	API/Framework Expectation	APDU Expectation
	5- Select EF-CNR file whose Increase access condition is NEVER Perform the increase method ~~Delete instance of applet~~Applet1 finalizes		
6	**readBinary method with no Access ~~to the GSM File System as Access~~ Domain Parameter** Send an Envelope that triggers the applet with the EVENT_UNFORMATTED_SMS_PP_ENV event. Select EF-TARU file whose Read access condition is ALWAYS Perform the readBinary method: fileOffset = 0 resp = abRead[] respOffset = 0 respLength = 3 ~~Install (install) with:~~ ~~- Length of Access Domain field value is '1'~~ ~~- Access Domain Parameter value is 'FF' (No access to the GSM File System)~~ ~~Select EFTARU file whose Read access condition is ALWAYS~~ ~~Perform the readBinary method:~~ ~~fileOffset = 0~~ ~~resp = baRead[]~~ ~~respOffset = 0~~ ~~respLength = 3~~ ~~Delete instance of applet~~	SIMViewException AC_NOT_FULFILLED is thrown	
7	**updateRecord method with no Access Domain Parameter** Send an Envelope that triggers the applet with the EVENT_UNFORMATTED_SMS_PP_ENV event. Select EF-SMS file whose Update access condition is CHV1 Perform the updateRecord method: fileOffset = 0 resp = abUpdate[] respOffset = 0 respLength = 3	SIMViewException AC_NOT_FULFILLED is thrown	
8	**invalidate method with no Access Domain Parameter** Send an Envelope that triggers the applet with the EVENT_UNFORMATTED_SMS_PP_ENV event. Select EF-ADN file whose Invalidate access condition is CHV2 Perform the invalidate method	SIMViewException AC_NOT_FULFILLED is thrown	
9	**rehabilitate method with no Access Domain Parameter** Send an Envelope that triggers the applet with the EVENT_UNFORMATTED_SMS_PP_ENV event. Select EF-SUME file Rehabilitate access condition is ADM0 Perform the rehabilitate method	SIMViewException AC_NOT_FULFILLED is thrown	
10	**increase method with no Access Domain Parameter**	SIMViewException AC_NOT_FULFILLED is thrown	

Id	Description	API/Framework Expectation	APDU Expectation
	Send an Envelope that triggers the applet with the EVENT_UNFORMATTED_SMS_PP_ENV event. Select EF-CNR file whose Increase access condition is NEVER Perform the increase method Applet2 finalizes Applet3 restore EF-SUME		

6.3.8.64.4 Test Coverage

Note : As Item Position management is not fully specified in the [7] or [8] all possible tests cannot be performed.

CRR number	Test case number
CRRN1	1, 2, 3, 4, 5
CRRP1	Not tested
CRRP2	6, 7, 8, 9, 10

6.3.8.75 Priority Level

Test Area Reference: FWK_TIN_PRLV

6.3.8.75.1 Conformance Requirements

Normal execution

CRRN1: The priority specifies the order of activation of an applet compared to the other applet registered to the same event ('01': Highest priority level, 'FF': Lowest priority level)

CRRN2: If two or more applets are registered to the same event and have the same priority level, the applets are activated according to their installation date (i.e. the most recent applet is activated first)

6.3.8.75.2 Test suite files

Test Script: FWK_TIN_PRLV_x1.scr, x from 1 to 12

Test Applet: FWK_TIN_PRLV_x1.java, x from 1 to 12, , 8A, 8B, 9A, 9B, 10A, 10B

Load Script: FWK_TIN_PRLV_x1.ldr, x from 1 to 12

Cleanup Script: FWK_TIN_PRLV_x1.clr, x from 1 to 12

Parameter File: FWK_TIN_PRLV_x1.par, x from 1 to 12, , 8A, 8B, 9A, 9B, 10A, 10B

6.3.8.75.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
0	All applets are registered on an EVENT_UNFORMATTED_SMS_PP_ENV event		

1	**Trigger 2 applets with 2 different maximum Priority Levels** Install (install) applet–1 with priority level '2' and applet–2 with priority level '1', from package fwk_tin_prlv_1~~P~~. Send an Envelope that triggers the 2 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. ~~Check that applet 2 is triggered before applet 1.~~ ~~A static variable is used to validate triggering order.~~ Delete applets instances and packages	A static variable is used to validate triggering order: applet2 is triggered before applet1		
2	**Trigger 2 applets with 2 different maximum Priority Levels** Install (install) applet–1 with priority level '1' and applet–2 with priority level '2', from package fwk_tin_prlv_2~~P~~. Send an Envelope that triggers the 2 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. ~~Check that applet 1 is triggered before applet 2.~~ ~~A static variable is used to validate triggering order.~~ Delete applets instances and packages	A static variable is used to validate triggering order: applet1 is triggered before applet2.		
3	**Trigger 2 applets with 2 different ~~'key'~~ Priority Levels** Install (install) applet–1 with priority level '80' and applet–2 with priority level '7F', from package fwk_tin_prlv_3~~P~~. Send an Envelope that triggers the 2 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. ~~Check that applet 2 is triggered before applet 1~~ ~~A static variable is used to validate triggering order.~~ Delete applets instances and packages	A static variable is used to validate triggering order: applet2 is triggered before applet1		
4	**Trigger 2 applets with 2 different Priority Levels** Install (install) applet1 with priority level '7F' and applet2 with priority level '80', from package fwk_tin_prlv_4. Send an Envelope that triggers the 2 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. Delete applets instances and packages	A static variable is used to validate triggering order: applet2 is triggered before applet1		

45	**Trigger 3 applets with the same Priority Level** Install (install) applet 1, 2, 3 in this order with same priority level from package fwk_tin_prlv_5P. Send an Envelope that triggers the 3 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. Check that applet 3 is triggered before applet 2 Check that applet 2 is triggered before applet 1. A static variable is used to validate triggering order. Delete applets instances and packages.	A static variable is used to validate triggering order: applet3 is triggered before applet2, and applet2 is triggered before applet1.	
65	**Trigger 2 applets from 2 classes, with 2 different Priority Level** Install (install) applet–1 from class AC with priority level '2' Install (install) applet–2 from class BD with priority level '1' A static variable is used to validate triggering order Send an Envelope that triggers the 2 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. Check that applet 2 is activated before applet 1 Delete applets instances and packages	A static variable is used to validate triggering order: applet2 is triggered before applet1	
76	**Trigger 2 applets from 2 classes, with the same Priority Level** Install (install) applet–1 from class AC with priority level '1' Install (install) applet–2 from class BD with priority level '1' A static variable is used to validate triggering order Send an Envelope that triggers the 2 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. Check that applet 2 is activated before applet 1 Delete applets instances and packages	A static variable is used to validate triggering order: applet2 is triggered before applet1	
87	**Trigger 2 applets from 2 packages, with 2 different Priority Level** Install package fwk_tin_prlv_8. Install (install) applet–1 from package fwk_tin_prlv_8AP with priority level '2' Install (install) applet–2 from package fwk_tin_prlv_8B Q with priority level '1' A static variable is used to validate triggering order Send an Envelope that triggers the 2 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. Check that applet 2 is activated before applet 1 Delete applets instances ad packages	A static variable is used to validate triggering order: applet2 is triggered before applet1	
98	**Trigger 2 applets from 2 packages, with the same Priority Level**		

	Install package fwk_tin_prlv_9. Install (install) applets 1 from package fwk_tin_prlv_9A and applet2 from package fwk_tin_prlv_9B ~~Install (install) applets 1 and 2~~ in this order, with same priority level Send an Envelope that triggers the 2 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. ~~Check that applet 10 is triggered before applet 9~~ Delete applets instances and packages	A static variable is used to validate triggering order: applet2 is triggered before applet1	
10~~9~~	**Trigger 4 applets from 2 packages** 1-Install packages fwk_tin_prlv_10, fwk_tin_prlv_10A and fwk_tin_prlv_10B. Install (install) 2 applets 1 then 2 from package fwk_tin_prlv_10A, with respectively priority levels 1 and 2. Send an Envelope that triggers the 2 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. ~~Check that applet 1 is triggered before applet 2~~ 2- Install (install) 2 applets 3 then 4 from package fwk_tin_prlv_10B, ~~–~~ with respectively priority levels 1 and 2. Send an Envelope that triggers the 4 applets. ~~Check that applet 3 is triggered before applet 1, 4, then 2.~~ Delete applets instances and packages	1- A static variable is used to validate triggering order: applet1 is triggered before applet2 2- Applet3 is triggered before applets 1, 4, then 2.	
11	**Trigger 4 applets with the same Priority Level then delete them one after another and trigger them each time** 1- Install (install) applet1, 2, 3, 4 in this order with same priority level from package fwk_tin_prlv_11. Send an Enveloppe that triggers the 4 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. Delete applet instance 4 2- Send an Enveloppe that triggers the 3 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. Delete applet instance 3 3- Send an Enveloppe that triggers the 2 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. Delete remaining applet instances and packages	1- A static variable is used to validate triggering order: applets are triggered in order 4, 3, 2, 1. 2- Applets are triggered in order 3, 2, 1. 3- Applets are triggered in order 2, 1.	
12	**Trigger 5 applets with different Priority Levels, alternating install and delete**		

	1- Install (install) applets 1, 2, 3, 4 in this order with respective priority levels 1, 2, 1, 2 Send an Enveloppe that triggers the 4 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. 2- Delete applet instance 1 and install (install) applet5 with priority level 2 Send an Enveloppe that triggers the 4 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event. 3- Re-install (install) applet1 with priority level 1 Send an Enveloppe that triggers the 5 applets with the EVENT_UNFORMATTED_SMS_PP_ENV event.	1- A static variable is used to validate triggering order: applets are triggered in order 3, 1, 4, 2 2- Applets are triggered in order 3, 5, 4, 2 3- Applets are triggered in order 1, 3, 5, 4, 2	

6.3.8.75.4 Test Coverage

CRR number	Test case number
CRRN1	1, 2, 3, 4, 65, 87, 10, 129
CRRN2	5, 7, 9, 114, 6, 8

6.3.9 File System Context

6.3.9.1 Initial Context

Test Area Reference: FWK_FSC_INIT

6.3.9.1.1 Conformance Requirements

Normal Execution

CRRN1: At the invocation of the processToolkit method of a toolkit applet, the current file is the MF.

6.3.9.1.2 Test Suite Files

Test Script: FWK_FSC_INIT_1.scr

Test Applet: FWK_FSC_INIT_1.java

FWK_FSC_INIT_2.java

Load Script: FWK_FSC_INIT_1.ldr

Cleanup Script: FWK_FSC_INIT_1.clr

FWK_FSC_INIT_2.clr

Parameter File: FWK_FSC_INIT_1.par

6.3.9.1.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
0	**SIM Initialisation**	Responses ignored.	
1	**MF is the selected DF in processToolkit()** An ENVELOPE APDU containing a formatted	No exception shall be thrown.	

Id	Description	API/Framework Expectation	APDU Expectation
	SMS PP for Applet 1 is issued to the SIM byte[] fci = new byte[10] fciOffset = 0 fciLength = 7 status()	Shall return 7. fci shall contain the following part of the FCI structure: < XX XX XX XX 3F 00 01 >	
2	**No EF is selected** rehabilitate ()	SIMView exception shall be thrown with reason NO_EF_SELECTED	
3	**MF is selected even when an applet triggered before selected any other file...** Applets 1 and 2 register to EVENT_DOWNLOAD_USER_ACTIVITY. Applet 1 has higher priority than Applet 2. An ENVELOPE “EVENT - USER ACTIVITY” is sent to the SIM 1 - Applet 1: - is triggered by event_event_download_user_activity - selects DF_GSM and EF_IMSI 2 - Applet 2: - is triggered by event_event_download_user_activity fciOffset = 0 fciLength = 7 status() 3 - rehabilitate ()	1 - No exception shall be thrown. 2 - No exception shall be thrown. Shall return 7. fci shall contain the following part of the FCI structure: < XX XX XX XX 3F 00 01 > 3 - SIMView exception shall be thrown with reason NO_EF_SELECTED	

6.3.9.1.4 Test Coverage

CRR Number	Test Case Number
CRRN1	1, 2, 3

6.3.9.2 Context Preservation (current file)

Test Area Reference: FWK_FSC_CUFI

6.3.9.2.1 Conformance Requirements

Normal execution

CRRN1: When calling the method select (), the current files (file context) of any other applets shall not be changed (see GSM 03.19 [] - §5.2).

CRRN2: The select() methods select a file without changing the current file of any other applet or of the subscriber session.

CRRN3: After invocation of ProactiveHandler.send() method: the current file context of the toolkit applet is unchanged (see GSM 03.19 [] - §5.2.).

6.3.9.2.2 Test Suite Files

Test Script: FWK_FSC_CUFI_1.scr

Test Applet: FWK_FSC_CUFI_1.java

FWK_FSC_CUFI_2.java

Load Script: FWK_FSC_CUFI_1.ldr

Cleanup Script: FWK_FSC_CUFI_1.clr

FWK_FSC_CUFI_2.clr

Parameter File: FWK_FSC_CUFI_1.par

6.3.9.2.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
0	**SIM Initialisation**	Responses ignored.	
1	**No change to file context by another applet** `Applet1 registers to EVENT_FORMATTED_SMS_PP_ENV.` `Applet2 registers to EVENT_CALL_CONTROL_BY_SIM` `1 - Applet 1:` `- is triggered by a formatted SMS` `- selects DF_SIMTEST and EF_TARU` `- fileOffset = 0; dataLength = 2; dataOffset = 0;` `- buffer = {0xCA, 0xFE }` `- updateBinary (): first 2 bytes of EF_TARU are written as ‘CA FE’.` `- issues a proactive command “Get Inkey”.` `2 - An ENVELOPE APDU containing a CALL CONTROL BY SIM is issued to the SIM` `Applet 2:` `- is triggered by a CALL CONTROL BY SIM` `- selects DF_TELECOM and EF_ADN.` `3 - The terminal response for Get Inkey reactivates Applet 1:` `- fileOffset = 0; respLength = 2; respOffset = 0;` `- readBinary () info buffer2`	1 - No exception shall be thrown. 2 - No exception shall be thrown. 3 - No exception shall be thrown. The value of buffer2 is { 0xCA, 0xFE }	A GET INKEY proactive command is fetched from the SIM
2	**No change to file context by subscriber session** `1 - Applet 1` `- issues a proactive command “Get Inkey”.` `2 - Subscriber session selects DF_TELECOM and EF_ADN.` `3 - The terminal response for Get Inkey reactivates Applet 1:` `- fileOffset = 0; respLength = 2; respOffset = 0;` `- readBinary () info buffer2`	1 - No exception shall be thrown. 3 - No exception shall be thrown. The value of buffer2 is { 0xCA, 0xFE }	1 - A GET INKEY proactive command is fetched from the SIM
3	**No change by applet of subscriber session context** `1 - Applet 1:` `- selects DF_SIMTEST and EF_TNU` `- issues a proactive command “Get Inkey”.` `2 - subscriber session reads record 1 of current file (shall be EF_ADN)` `3 - The terminal response for Get Inkey reactivates Applet 1, which terminates execution`	1 - No exception shall be thrown. 3 - No exception shall be thrown.	1 - A GET INKEY proactive command is fetched from the SIM 2 - READ RECORD absolute number 1 shall read “FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF” (from EF_{ADN})

6.3.9.2.4 Test Coverage

CRR Number	Test Case Number
CRRN1	1
CRRN2	1,2,3
CRRN3	1,2

6.3.9.3 Context Preservation (current record pointer)

Test Area Reference: FWK_FSC_CURE

6.3.9.3.1 Conformance Requirements

Normal execution

CRRN1: When the seek method is called by one applet, the record pointer of any other applet is not changed.

CRRN2: *updateRecord*: the current record pointer of other applets / subscriber shall not be changed in case of linear fixed EF

CRRN3: *updateRecord*: the record pointer of a cyclic EF shall be changed for all other applets / subscriber to the record number 1.

CRRN4: *readRecord:* read data bytes of the linear fixed or cyclic EF currently selected by the applet without changing the current record pointer of any other applet / subscriber.

CRRN5: *increase*: the last updated record of the cyclic EF currently selected becomes record number 1 for every other applet and subscriber session.

6.3.9.3.2 Test Suite Files

Test Script: FWK_FSC_CURE_1.scr

Test Applet: FWK_FSC_CURE_1.java

FWK_FSC_CURE_2.java

Load Script: FWK_FSC_CURE_1.ldr

Cleanup Script: FWK_FSC_CURE_1.clr

FWK_FSC_CURE_2.clr

Parameter File: FWK_FSC_CURE_1.par

6.3.9.3.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
0	**SIM Initialisation**	Responses ignored.	
1	**Seek without affecting another record pointer** aApplet1 registers to EVENT_FORMATTED_SMS_PP_ENV Applet 2 registers to EVENT_CALL_CONTROL_BY_SIM 1 - Applet 1: - is triggered by a formatted SMS event - selects DF_SIMTEST and EF_LARU - reads record 2 using NEXT so that the current record pointer is set to record 2 - issues a proactive command, e.g. Get Inkey. 2 - An ENVELOPE APDU containing a CALL CONTROL BY SIM is issued to the SIM	1 - No exception shall be thrown. 2 - No exception shall be thrown. 3 - No exception shall be thrown.	1 - A GET INKEY proactive command is fetched from the SIM

Id	Description	API/Framework Expectation	APDU Expectation
	Applet 2: - is triggered by a CALL CONTROL event - selects DF_SIMTEST and EF_LARU - performs a seek of pattern {0x55} from beginning forward, which finds record 1. - returns from processToolkit 3 - The terminal response for Get Inkey reactivates Applet 1: - call readRecord() using CURRENT - the record read should still be record 2 of EF_LARU, containing {0xAA, 0xAA, 0xAA, 0xAA}		
2	**updateRecord in linear fixed EF without affecting current pointer of others** 1 - Applet 1: - is triggered by a formatted SMS event - selects DF_SIMTEST and EF_LARU - reads record 2 using NEXT so that the current record pointer is set to record 2 - issues a proactive command, e.g. Get Inkey. 2 - An ENVELOPE APDU containing a CALL CONTROL BY SIM is issued to the SIM Applet 2: - is triggered by a CALL CONTROL BY SIM event - selects DF_SIMTEST and EF_LARU - updates record 1, by using mode “NEXT”. - returns from processToolkit 3 - The terminal response for Get Inkey reactivates Applet 1: - call readRecord() using CURRENT - the record read should still be record 2 of EF_LARU, containing {0xAA, 0xAA, 0xAA, 0xAA}	1 - No exception shall be thrown. 2 - No exception shall be thrown. 3 - No exception shall be thrown.	1 - A GET INKEY proactive command is fetched from the SIM
3	**readRecord in linear fixed EF without affecting current pointer of others** 1 - Applet 1: - is triggered by a formatted SMS event - selects DF_SIMTEST and EF_LARU - reads record 2 using NEXT so that the current record pointer is set to record 2 - issues a proactive command, e.g. Get Inkey. 2 - An ENVELOPE APDU containing a CALL CONTROL BY SIM is issued to the SIM Applet 2: - is triggered by a CALL CONTROL BY SIM event - selects DF_SIMTEST and EF_LARU - reads record 1, by using mode “NEXT”. - returns from processToolkit 3 - The terminal response for Get Inkey reactivates Applet 1: - call readRecord() using CURRENT - the record read should still be record 2 of EF_LARU, containing {0xAA, 0xAA, 0xAA, 0xAA}	1 - No exception shall be thrown. 2 - No exception shall be thrown. 3 - No exception shall be thrown.	1 - A GET INKEY proactive command is fetched from the SIM
~~4~~	~~**updateRecord cyclic EF: record pointer changed to record number 1**~~ ~~1 – The subscriber session selects DF_SIMTEST and EF_CARU~~ ~~– reads record, by using mode “NEXT”.~~ ~~2 – Applet 1:~~ ~~- is triggered by a formatted SMS event~~ ~~- selects DF_SIMTEST and EF_CARU~~	~~2 - No exception shall be thrown.~~ ~~3 - No exception shall be thrown.~~ ~~5 - No exception shall be thrown.~~ ~~Value “11 11 11” is read.~~	~~1 –~~ ~~The value “AA AA AA” is obtained as a response to READ RECORD.~~ ~~2 - A GET INKEY proactive command is fetched from the SIM~~

Id	Description	API/Framework Expectation	APDU Expectation
	- readRecord(), by using mode “NEXT”. - issues a proactive command, e.g. Get Inkey. 3 – An ENVELOPE APDU containing a CALL CONTROL BY SIM is issued to the SIM Applet 2: - is triggered by a CALL CONTROL BY SIM event - selects DF_SIMTEST and EF_CARU - updates record using “PREVIOUS” to ’11 11 11’ - returns from processToolkit 4 – The subscriber session - reads record, by using mode “CURRENT”. 5 – The terminal response for Get Inkey reactivates Applet 1: - readRecord() using mode “CURRENT”		4 - The value “11 11 11” is obtained as a response to READ RECORD.
5	**increase cyclic EF: last increased record becomes number 1** 1 – The subscriber session selects DF_SIMTEST and EF_CARU - reads record, by using mode “NEXT”. 2 – Applet 1: - is triggered by a formatted SMS event - selects DF_SIMTEST and EF_CARU - readRecord(), by using mode “NEXT”. - issues a proactive command, e.g. Get Inkey. 3 – An ENVELOPE APDU containing a CALL CONTROL BY SIM is issued to the SIM Applet 2: - is triggered by a CALL CONTROL BY SIM event - selects DF_SIMTEST and EF_CARU - increase() with an increment of ’11 11 11’ - returns from processToolkit 4 – The subscriber session - reads record, by using mode “CURRENT”. 5 – The terminal response for Get Inkey reactivates Applet 1: - readRecord() using mode “CURRENT”	2 - No exception shall be thrown. 3 - No exception shall be thrown. 5 - No exception shall be thrown. Value “22 22 22” is read.	1 - The value “55 55 55” is obtained as a response to READ RECORD. 2 - A GET INKEY proactive command is fetched from the SIM 4 - The value “22 22 22” is obtained as a response to READ RECORD.

6.3.9.3.4 Test Coverage

CRR Number	Test Case Number
CRRN1	1
CRRN2	2
CRRN3	not tested (see Note)4
CRRN4	3
CRRN5	not tested (see Note)5

Note: These requirements have not been tested because of an inconsistent behavior in 03.19, which is foreseen to be corrected in future releases.

6.3.10 Other parts transferred to framework from API

6.3.10.1 A handler is a temporary JCRE Entry Point object

Test Area Reference: FWK_API_HEPO

6.3.10.1.1 Conformance Requirement:

Normal execution

CRRN1: The EnvelopeHandler is a Temporary JCRE Entry Point Object (see Javacard 2.1 Runtime Environment (JCRE) Specification [12]).

CRRN2: The EnvelopeResponseHandler is a Temporary JCRE Entry Point Object (see Javacard 2.1 Runtime Environment (JCRE) Specification [12]).

CRRN3: The ProactiveHandler is a Temporary JCRE Entry Point Object (see Javacard 2.1 Runtime Environment (JCRE) Specification [12]).

CRRN4: The ProactiveResponseHandler is a Temporary JCRE Entry Point Object (see Javacard 2.1 Runtime Environment (JCRE) Specification [12]).

Parameters error

Context errors

6.3.10.1.2 Test suite files

Test Script: FWK_API_HEPO_1.scr

Test Applet: FWK_API_HEPO_1.java

Load Script: FWK_API_HEPO_1.ldr

Cleanup Script: FWK_API_HEPO_1.clr

Parameter File: FWK_API_HEPO_1.par

6.3.10.1.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**EnvelopeHandler.getTheHandler and store it in a static field of the toolkit applet**	SecurityException is thrown	
2	**EnvelopeHandler.getTheHandler and store it in a field of the toolkit applet**	SecurityException is thrown	
3	**EnvelopeResponseHandler.getTheHandler and store it in a static field of the toolkit applet**	SecurityException is thrown	
4	**EnvelopeResponseHandler.getTheHandler and store it in a field of the toolkit applet**	SecurityException is thrown	
5	**ProactiveHandler.getTheHandler and store it in a static field of the toolkit applet**	SecurityException is thrown	
6	**ProactiveHandler.getTheHandler and store it in a field of the toolkit applet**	SecurityException is thrown	
7	**Build and send a DISPLAY TEXT command to be able to get the reference of the ProactiveReponseHandler** **ProactiveResponseHandler.getTheHandler and store it in a static field of the toolkit applet**	 SecurityException is thrown	Proactive command fetched and terminal response is issued
8	**ProactiveResponseHandler.getTheHandler and store it in a field of the toolkit applet**	SecurityException is thrown	

6.3.10.1.4 Test Coverage

CRR number	Test case number
N1	1, 2
N2	3, 4
N3	5, 6
N4	7, 8

6.3.10.2 Transaction

Test Area Reference: FWK_API_TRAN

6.3.10.2.1 Conformance Requirement:

Normal execution

CRRN1: A pending toolkit applet transaction at the ProactiveHandler.send() method invocation is aborted..

6.3.10.2.2 Test suite files

Test Script: FWK_API_TRAN_1.scr

Test Applet: FWK_API_TRAN_1.java

Load Script: FWK_API_TRAN_1.ldr

Cleanup Script: FWK_API_TRAN_1.clr

Parameter File: FWK_API_TRAN_1.par

6.3.10.2.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**Verify that transaction is aborted when a proactive command is sent** Initialise a byte field with 0x05 Build a display text proactive command. beginTransaction() Update the byte with 0x02 send the proactive command Verify that the byte value is 0x05 JCSystem.getTransactionDepth()	 Shall return 0	 Proactive command fetched and terminal response is issued

6.3.10.2.4 Test Coverage

CRR number	Test case number
N1	1

6.3.10.3 Timer Id between Applets

Test Area Reference: FWK_API_TMID

6.3.10.3.1 Conformance Requirement:

Context errors

CRRC1: The method ToolkitRegistry.releaseTimer() shall throw a ToolkitException with INVALID_TIMER_ID reason if the timer is valid but isn't allocated to this applet.

6.3.10.3.2 Test suite files

Test Script: FWK_API_TMID_1.scr

Test Applet: FWK_API_TMID_1.java

Load Script: FWK_API_TMID_1.ldr

Cleanup Script: FWK_API_TMID_1.clr

Parameter File: FWK_API_TMID_1.par

6.3.10.3.3 Test Procedure

Id	Description	API/Framework Expectation	APDU Expectation
1	**During installation : First instance allocate a timer and store the returned value in a static field. Second instance allocate a timer. Trig second instance and try to releaseTimer() with the static field value.**	**releaseTimer() shall throw a ToolkitException with INVALID_TIMER_ID reason**	

6.3.10.3.4 Test Coverage

CRR number	Test case number
N1	1

Annex C (normative): Default Prepersonalisation

C.1 General Default Prepersonalisation

This table shows the default prepersonalisation, the file system and the files' content, that the test SIM cards shall contain unless otherwise stated.

Name	Identifier	Default Value	Special Features
EF_{ICCID}	2FE2	0F FF FF FF FF FF FF FF FF FF	This value is not compliant with GSM 11.11
EF_{IMSI}	6F07	FF FF FF FF FF FF FF FF FF	This value is not compliant with GSM 11.11
EF_{LP}	6F05	01 FF FF FF	
EF_{Kc}	6F20	FF FF FF FF FF FF FF FF 07	
$EF_{PLMNsel}$	6F30	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	
EF_{HPLMN}	6F31	05	
EF_{ACMmax}	6F37	00 00 00	Access condition UPDATE: CHV1
EF_{SST}	6F38	FF 3F C3 03 0C 00 FF 0F 00 33	
EF_{ACM}	6F39	00 00 00	Access condition UPDATE: CHV1
EF_{PUCT}	6F41	FF FF FF 00 00	Access condition UPDATE: CHV1
EF_{BCCH}	6F74	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	
EF_{ACC}	6F78	00 00	
EF_{FPLMN}	6F7B	FF FF FF FF FF FF FF FF FF FF FF FF	
EF_{LOCI}	6F7E	FF FF FF FF 00 F0 00 00 00 FF 01	
EF_{AD}	6FAD	00 FF FF	
EF_{Phase}	6FAE	03	
EF_{FDN}	6F3B	Default value in all the records: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	Records: 5
EF_{SMSP}	6F42	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	Records: 1
EF_{LND}	6F44	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	Records: 1
EF_{SMSS}	6F43	FF FF	
EF_{SMS}	6F3C	1st record: 00 FF … FF(length 176) 2nd record:00 FF … FF(length 176) 3rd record: 00 FF … FF(length 176)	Records: 3
EF_{ADN}	6F3A	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	Records: 1
EF_{CCP}	6F3D	FF FF FF FF FF FF FF FF FF FF FF FF FF FF	
EF_{MSISDN}	6F40	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	Records: 1
EF_{SDN}	6F491	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	Records: 1
EF_{SUME}	6F54	85 0C 54 4F 4F 4C 4B 49 54 20 54 45 53 54 FF FF FF FF	
EF_{CBMI}	6F45	FF FF	
EF_{IM}	4F20	FF FF FF FF FF FF FF FF FF FF FF	

The default value for the CHV1 shall be "0x31 0x31 0x31 0x31 0xFF 0xFF 0xFF 0xFF" and its state shall be 'disabled' during test applets execution.

C.2 Sim.Access.SimView test default prepersonalisation

C.2.1 $DF_{SIMTEST}$ (SIM Test)

Identifier: '0319'

C.2.2 EF_{TNR} (Transparent Never Read)

Identifier: ’6F01’		Structure: transparent		Mandatory
File size: 3 bytes		Update activity: low		
Access Conditions: READ NEVER UPDATE ALWAYS INVALIDATE ALWAYS REHABILITATE ALWAYS				
Bytes	Description	Default Value	M/O	Length
1 – 3	Test Data	AA AA AA	M	3 bytes

C.2.3 EF_{TNU} (Transparent Never Update)

Identifier: '6F02'		Structure: transparent		Mandatory
File size: 3 bytes		Update activity: low		
Access Conditions: READ ALWAYS UPDATE NEVER INVALIDATE ALWAYS REHABILITATE ALWAYS				
Bytes	Description	Default Value	M/O	Length
1 - 3	Test Data	55 55 55	M	3 bytes

C.2.4 EF_{TARU} (Transparent Always Read and Update)

Identifier: '6F03'		Structure: transparent		Mandatory
File size: 260 bytes		Update activity: low		
Access Conditions: READ ALWAYS UPDATE ALWAYS INVALIDATE ALWAYS REHABILITATE ALWAYS				
Bytes	Description	Default Value	M/O	Length
1 - 260	Test Data	FF ... FF	M	260 bytes

C.2.5 EF_{CNR} (Cyclic Never Read)

Identifier: '6F04'	Structure: cyclic	Mandatory
Record length: 3 bytes	Update activity: high	

Access Conditions:

READ	NEVER
UPDATE	ALWAYS
INCREASE	ALWAYS
INVALIDATE	ALWAYS
REHABILITATE	ALWAYS

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes
2	Test Data	00 00 00	M	3 bytes

C.2.6 EF_{CNU} (Cyclic Never Update)

Identifier: '6F05'	Structure: cyclic	Mandatory
Record length: 3 bytes	Update activity: high	

Access Conditions:

READ	ALWAYS
UPDATE	NEVER
INCREASE	NEVER
INVALIDATE	ALWAYS
REHABILITATE	ALWAYS

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes
~~3~~2	Test Data	00 00 00	M	3 bytes

C.2.7 EF_{CNIC} (Cyclic Never Increase)

Identifier: '6F06	Structure: cyclic	Mandatory
Record length: 3 bytes	Update activity: high	

Access Conditions:

READ	ALWAYS
UPDATE	ALWAYS
INCREASE	NEVER
INVALIDATE	ALWAYS
REHABILITATE	ALWAYS

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes
2	Test Data	00 00 00	M	3 bytes

C.2.8 EF_{CNIV} (Cyclic Never Invalidate)

Identifier: ’6F07	Structure: cyclic	Mandatory
Record length: 3 bytes	Update activity: high	

Access Conditions:

READ	ALWAYS
UPDATE	ALWAYS
INCREASE	ALWAYS
INVALIDATE	NEVER
REHABILITATE	ALWAYS

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes
2	Test Data	00 00 00	M	3 bytes

C.2.9 EF_{CNRH} (Cyclic Never Rehabilitate)

Identifier: ’6F08’	Structure: cyclic	Mandatory
Record length: 3 bytes	Update activity: high	

Access Conditions:

READ	ALWAYS
UPDATE	ALWAYS
INCREASE	ALWAYS
INVALIDATE	ALWAYS
REHABILITATE	NEVER

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes
2	Test Data	00 00 00	M	3 bytes

C.2.10 EF_{CARU} (Cyclic Always Read and Update)

Identifier: ’6F09’	Structure: cyclic	Mandatory
Record length: 3 bytes	Update activity: high	

Access Conditions:

READ	ALWAYS
UPDATE	ALWAYS
INCREASE	ALWAYS
INVALIDATE	ALWAYS
REHABILITATE	ALWAYS

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	55 55 55	M	3 bytes
2	Test Data	AA AA AA	M	3 bytes

C.2.11 EF_{LNR} (Linear Fixed Never Read)

Identifier: '6F0A'	Structure: linear fixed	Mandatory
Record length: 4 bytes	Update activity: low	

Access Conditions:

READ	NEVER
UPDATE	ALWAYS
INVALIDATE	ALWAYS
REHABILITATE	ALWAYS

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data - Record 1	FF FF FF FF	M	4 bytes
2	Test Data - Record 2	FF FF FF FF	M	4 bytes

C.2.12 EF_{LNU} (Linear Fixed Never Update)

Identifier: '6F0B'	Structure: linear fixed	Mandatory
Record length: 4 bytes	Update activity: low	

Access Conditions:

READ	ALWAYS
UPDATE	NEVER
INVALIDATE	ALWAYS
REHABILITATE	ALWAYS

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data - Record 1	FF FF FF FF	M	4 bytes
2	Test Data - Record 2	FF FF FF FF	M	4 bytes

C.2.13 EF_{LARU} (Linear Fixed Always Read and Update)

Identifier: '6F0C'	Structure: linear fixed	Mandatory
Record length: 4 bytes	Update activity: low	

Access Conditions:

READ	ALWAYS
UPDATE	ALWAYS
INVALIDATE	ALWAYS
REHABILITATE	ALWAYS

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data - Record 1	55 55 55 55	M	4 bytes
2	Test Data - Record 2	AA AA AA AA	M	4 bytes

C.2.14 EF_{CINA} (Cyclic Increase Not Allowed)

Identifier: '6F0D'	Structure: cyclic	Mandatory
Record length: 3 bytes	Update activity: high	

Access Conditions:

READ	ALWAYS
UPDATE	ALWAYS
INCREASE	ALWAYS (see note 1)
INVALIDATE	ALWAYS
REHABILITATE	ALWAYS

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes
2	Test Data	00 00 00	M	3 bytes

Note 1: This file will be personalised in a way such that increase is not allowed, as indicated by the FCI byte 8, bit 7 (GSM 11.11: FCI structure of an EF returned by the SELECT command)

C.2.15 EF_{TRAC} (Transparent Read Access Condition CHV2)

Identifier: '6F0E'	Structure: transparent	Mandatory
Record length: 3 bytes	Update activity: low	

Access Conditions:

READ	CHV2
UPDATE	ALWAYS
INCREASE	ALWAYS
INVALIDATE	ALWAYS
REHABILITATE	ALWAYS

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes

C.2.16 EF_{TIAC} (Transparent Invalidate Access Condition CHV1)

Identifier: '6F0F'	Structure: transparent	Mandatory
Record length: 3 bytes	Update activity: low	

Access Conditions:

READ	ALWAYS
UPDATE	ALWAYS
INCREASE	ALWAYS
INVALIDATE	CHV1
REHABILITATE	ALWAYS

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes

C.2.17 EF_{CIAC} (Cyclic Increase Access Condition CHV2)

Identifier: '6F10'	Structure: cyclic	Mandatory
Record length: 3 bytes	Update activity: low	

Access Conditions:

READ	ALWAYS
UPDATE	ALWAYS
INCREASE	CHV2
INVALIDATE	ALWAYS
REHABILITATE	ALWAYS

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes
2	Test Data	00 00 00	M	3 bytes

C.2.18 EF_{CIAA} (Cyclic Increase Access Condition ADM~~0~~)

Identifier: '6F11'	Structure: cyclic	Mandatory
Record length: 3 bytes	Update activity: low	

Access Conditions:

READ	ALWAYS
UPDATE	ALWAYS
INCREASE	ADM~~0~~
INVALIDATE	ALWAYS
REHABILITATE	ALWAYS

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes
2	Test Data	00 00 00	M	3 bytes

C.2.19 EF_{CNRI} (Cyclic Never Rehabilitate Invalidated)

Identifier: '6F12'	Structure: cyclic	Mandatory
Record length: 3 bytes	Update activity: low	

Access Conditions:

READ	ALWAYS
UPDATE	ALWAYS
INCREASE	ALWAYS
INVALIDATE	ALWAYS
REHABILITATE	NEVER

Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes
2	Test Data	00 00 00	M	3 bytes

The file status shall be invalidated as defined in [3]

Annex F (Normative):
AID numbering and acronyms for Framework tests

F.1 Toolkit Installation Parameters (TIN)

Test Area within the chapter	Acronyms	Numbering on 6 bits
Timer allocation	TMAL	000001
Item identifier	ITID	000010
Item position	ITPO	000011
Access conditions	ACCO	000100
Priority level	PRLV	000101
Maximum length for each menu entry	MLME	000110
Number of menu entries	NBME	000111
Memory space	MESP	001000

F.2 Minimum Handler Availability (MHA)

Test Area within the chapter	Acronyms	Numbering on 6 bits
ProactiveHandler	PAHD	000001
ProactiveResponseHandler	PRHD	000010
EnvelopeHandler	ENHD	000011
EnvelopeResponseHandler	ERHD	000100

F.3 Handler Integrity (HIN)

Test Area within the chapter	Acronyms	Numbering on 6 bits
ProactiveHandler	PAHD	000001
ProactiveResponseHandler	PRHD	000010
EnvelopeHandler	ENHD	000011
RFU (EnvelopeResponseHandler)	(ERHD)	000100

F.4 Applet Triggering (APT)

Test Area within the chapter	Acronyms	Numbering on 6 bits
EVENT_PROFILE_DOWNLOAD	EPDW	000001
EVENT_MENU_SELECTION	EMSE	000010
EVENT_MENU_SELECTION_HELP_REQUEST	EMSH	000011
EVENT_FORMATTED_SMS_PP_ENV	EFSE	000100
EVENT_UNFORMATTED_SMS_PP_ENV	EUSE	000101
EVENT_CALL_CONTROL_BY_SIM	ECCN	000110
EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM	EMCN	000111
EVENT_TIMER_EXPIRATION	ETEX	001000
EVENT_UNFORMATTED_SMS_CB	EUCB	001001
EVENT_EVENT_DOWNLOAD_MT_CALL	EDMC	001010
EVENT_EVENT_DOWNLOAD_CALL_CONNECTED	EDCC	001011
EVENT_EVENT_DOWNLOAD_CALL_DISCONNECTED	EDCD	001100
EVENT_EVENT_DOWNLOAD_LOCATION_STATUS	EDLS	001101
EVENT_EVENT_DOWNLOAD_USER_ACTIVITY	EDUA	001110
EVENT_EVENT_DOWNLOAD_IDLE_SCREEN_AVAILABLE	EDIS	001111
EVENT_EVENT_DOWNLOAD_CARD_READER_STATUS	EDCR	010000
RFU (EVENT_UNRECOGNIZED_ENVELOPE)	(EUEN)	010001
EVENT_STATUS_COMMAND	ESTC	010010

F.5 Proactive Command Sending (PCS)

Test Area within the chapter	Acronyms	Numbering on 6 bits
System Proactive commands	SPCO	000001
Interaction with GSM commands	IGCO	000010
Errors during proactive command sending	EPCS	000011

F.6 Envelope Response Posting (ERP)

Test Area within the chapter	Acronyms	Numbering on 6 bits
EVENT_CALL_CONTROL_BY_SIM	ECCN	000001
EVENT_MO_SHORT_MESSAGE_CONTROL_BY_SIM	EMCN	000010
EVENT_UNRECOGNIZED_ENVELOPE	EUEN	000011

F.7 Framework Security (FWS)

Test Area within the chapter	Acronyms	Numbering on 6 bits
Input data	INDA	000001
Output data	OUDA	000010

3GPP T3 Meeting #22
Marbella, Spain, 22 - 25 Jan 2002

Tdoc T3-020081

CR-Form-v3

CHANGE REQUEST

⌘ **43.019** **CR** **010** ⌘ rev **-** ⌘ Current version: **5.1.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM [X] ME/UE [ ] Radio Access Network [ ] Core Network [ ]

Title:	⌘	SET-UP-MENU command issued if all the items supporting help are disabled.
Source:	⌘	T3
Work item code:	⌘	USAT1-API-JAVA
Date:	⌘	23/01/02
Category:	⌘	**F**
Release:	⌘	REL-5

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
REL-4	*(Release 4)*
REL-5	*(Release 5)*

Reason for change:	⌘	Specify the behaviour when all the items supporting help are disabled
Summary of change:	⌘	Add a sentence in the *EVENT_MENU_SELECTION* clause.
Consequences if not approved:	⌘	

Clauses affected:	⌘	§ 6.2		
Other specs Affected:	⌘	[ ] Other core specifications [ ] Test specifications [ ] O&M Specifications	⌘	
Other comments:	⌘			

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.
2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.
3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

6.2 Applet Triggering

[..]

EVENT_MENU_SELECTION, EVENT_MENU_SELECTION_HELP_REQUEST

A toolkit applet might be activated upon selection in the ME's menu by the user, or request help on this specific menu.

In order to allow the user to choose in a menu, the SIM Toolkit Framework shall have previously issued a SET UP MENU proactive command. When a toolkit applet changes a menu entry of its registry object, the SIM Toolkit Framework shall dynamically update the menu stored in the ME during the current card session. The SIM Toolkit Framework shall use the data of the EFsume file when issuing the SET UP MENU proactive command.

The positions of the toolkit applet menu entries in the item list, the requested item identifiers and the associated limits (e.g. maximum length of item text string) are defined at the loading of the toolkit applet.

If at least one toolkit applet registers to EVENT_MENU_SELECTION_HELP_REQUEST, the SET UP MENU proactive command sent by the SIM Toolkit Framework shall indicate to the ME that help information is available unless all the menus entries that support help are disabled. A toolkit applet registered for one or more menu entries, may be triggered by the event EVENT_MENU_SELECTION_HELP_REQUEST, even if it is not registered to this event. A toolkit applet registered for one or more menu entries should provide help information.

CR-Form-v3

CHANGE REQUEST

⌘ **43.019** **CR** **011** ⌘ rev **-** ⌘ Current version: **5.1.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM [X] ME/UE [ ] Radio Access Network [ ] Core Network [ ]

Title:	⌘	Indication of the handler size to the applet
Source:	⌘	T3
Work item code:	⌘	USAT1-API-JAVA
Date:	⌘	23/01/2002
Category:	⌘	**B**
Release:	⌘	REL-5

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
REL-4	*(Release 4)*
REL-5	*(Release 5)*

Reason for change:	⌘	The applet writer cannot know the size of the system handler, this is required for the usage of some features like Bearer independent protocol The original CR in T3-010367 introduces a new method in the `public abstract` class `sim.toolkit.ViewHandler` the superclass of all handler classes. Unfortunately this would break the backward compatibility rules of the Java Card™ platform. Therefore this CR introduces this method in all final handler classes
Summary of change:	⌘	Addition of a method to get the size of the system handlers
Consequences if not approved:	⌘	

Clauses affected:	⌘	Annex A,B
Other specs affected:	⌘	[ ] Other core specifications ⌘ [ ] Test specifications [ ] O&M Specifications
Other comments:	⌘	

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at:
http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

List of changes to the API html and java source files Annex A and B

Class sim.toolkit.EnvelopeResponseHandler
+ getCapacity()
Class sim.toolkit.ProactiveHandler
+ getCapacity()
Class sim.toolkit.EnvelopeHandler
+ getCapacity()
Class sim.toolkit.ProactiveResponseHandler
+ getCapacity()

in all final handler classes as listed above

public short getCapacity()
throws ToolkitException

Returns the maximum size of the Simple TLV list managed by the handler.
Returns:
size in bytes
Throws:
ToolkitException - with the following reason codes:
HANDLER_NOT_AVAILABLE if the handler is busy

3GPP T3 Meeting #22
Marbella, Spain, 22 - 25 Jan 2002

Tdoc T3-020084

CR-Form-v3

CHANGE REQUEST

⌘ **43.019** **CR** **012** ⌘ rev **-** ⌘ Current version: **5.1.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM [X] ME/UE [ ] Radio Access Network [ ] Core Network [ ]

Title:	⌘	Clarification on framework behaviour for PoR using SMS SUBMIT
Source:	⌘	T3
Work item code:	⌘	USAT1-API-JAVA
Date:	⌘	23/01/02
Category:	⌘	**F**
Release:	⌘	REL-5

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
REL-4	*(Release 4)*
REL-5	*(Release 5)*

Reason for change:	⌘	Clarify framework behaviour when a PoR shall be sent using SMS-SUBMIT
Summary of change:	⌘	Add a paragraph describing the SIM Toolkit framework behaviour in the case of an EVENT_FORMATTED_SMS_PP_ENV event. Add a note which clarify the condition when busy status can be returned by the SIM Toolkit framework in the case of an EVENT_FORMATTED_SMS_PP_ENV event.
Consequences if not approved:	⌘	

Clauses affected:	⌘	**§6.2, §6.5, §6.6**
Other specs Affected:	⌘	[ ] Other core specifications [ ] Test specifications [ ] O&M Specifications
Other comments:	⌘	

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

6.2 Applet Triggering

[..]

EVENT_FORMATTED_SMS_PP_ENV

This event is triggered by an envelope APDU containing an SMS_DATADOWNLOAD BER TLV with an SMS_TPDU simple TLV according to TS 23.048[4].

The SIM Toolkit Framework shall:

- verify the TS 23.048[4] security of the SMS TPDU ;
- trigger the toolkit applet registered with the corresponding TAR defined at applet loading;
- take the optional Application Data posted by the triggered toolkit applet if present;
- secure and send the response packet using SMS-DELIVER-REPORT or SMS-SUBMIT .

The toolkit applet will only be triggered if the TAR is known and the security verified, application data will also be deciphered.

6.5 Envelope response handling

To allow a toolkit applet to answer to some specific events (e.g. EVENT_CALL_CONTROL_BY_SIM) the SIM Toolkit Framework shall provide the *sim.toolkit.ViewHandler.EditHandler.EnvelopeResponseHandler*.

The toolkit applet can then post a response to some events with the *post()* or the *postAsBERTLV()* methods, the toolkit applet can continue it's processing (e.g. prepare a proactive command) the SIM Toolkit Framework will return the response APDU defined by the toolkit applet (i.e. 9F xx ~~or~~, 9E xx or 91 xx).

Case of *EVENT_FORMATTED_SMS_PP_ENV:*

When the *post()* or the *postAsBERTLV()* method is invoked, the SIM Toolkit Framework shall, according to bit 6 of the second octet of the SPI defined in TS 23.048[4], build a SMS-DELIVER-REPORT or a SMS-SUBMIT (In that case the *statusType* method parameter is meaningless). If the SMS-SUBMIT is to be used, the SIM Toolkit Framework shall build and issue a Send Short Message proactive command as defined in TS 11.14 [3].

6.6 Handler availability

Table 1: Handler availability for each event

EVENT_	Reply busy	ProactiveHandler ProactiveResponseHandler	EnvelopeHandler	EnvelopeResponseHandler	Nb of triggered / registrered Applet
_FORMATTED_SMS_PP_ENV	Y (see Note 3)	Y	Y	Y	1 / n (per TAR)
_FORMATTED_SMS_PP_UPD	N	Y	Y	N	1 / n (per TAR)
_UNFORMATTED_SMS_PP_ENV	Y	Y	Y	Y	n / n
_UNFORMATTED_SMS_PP_UPD	N	Y	Y	N	n / n
_UNFORMATTED_SMS_CB	Y	Y	Y	N	n / n
_MENU_SELECTION	Y	Y	Y	N	1 / n (per Item Id)
_MENU_SELECTION_HELP_REQUEST	Y	Y	Y	N	1 / n (per Item Id)
_CALL_CONTROL	N	Y/N (see Note 2)	Y	Y	1 / 1
_SMS_MO_CONTROL	N	Y/N (see Note 2)	Y	Y	1 / 1
_TIMER_EXPIRATION	Y	Y	Y	N	1/ 8 (per timer) (see Note 1)
_EVENT_DOWNLOAD					
_MT_CALL	Y	Y	Y	N	n / n
_CALL_CONNECTED	Y	Y	Y	N	n / n
_CALL_DISCONNECTED	Y	Y	Y	N	n / n
_LOCATION_STATUS	Y	Y	Y	N	n / n
_USER_ACTIVITY	Y	Y	Y	N	n / n
_IDLE_SCREEN_AVAILABLE	Y	Y	Y	N	n / n
_CARD_READER_STATUS	Y	Y	Y	N	n / n
_UNRECOGNISED_ENVELOPE	Y	Y	Y	Y	n / n
_STATUS_COMMAND	N	Y/N (see Note 2)	N	N	n / n
_PROFILE_DOWNLOAD	N	Y/N (see Note 2)	N	N	n / n

NOTE 1: One toolkit applet can register to several timers, but a timer can only be allocated to one toolkit applet.
Note 2: Y/N means that handlers may / may not be available depending whether a proactive session is ongoing.
Note 3 : The framework may reply busy and not trigger the toolkit applet if a PoR using SMS SUBMIT is required in the incoming message and a proactive session is ongoing.

EnvelopeResponseHandler:

- The EnvelopeResponseHandler content must be posted before the first invocation of a ProactiveHandler.send method or before the termination of the processToolkit, so that the GSM applet can offer these data to the ME (eg 9Fxx/9Exx/91xx). After the first invocation of the ProactiveHandler.send method the EnvelopeResponseHandler is no more available

3GPP T3 Meeting #22
Marbella, Spain, 22 - 25 Jan 2002

Tdoc T3-020085

CR-Form-v3

CHANGE REQUEST

⌘ **43.019** **CR** **013** ⌘ rev **-** ⌘ Current version: **5.1.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM [X] ME/UE [ ] Radio Access Network [ ] Core Network [ ]

Title:	⌘	Introduction of Concatenated Short Messages in SMS Point to Point.
Source:	⌘	T3
Work item code:	⌘	USAT1-API-JAVA
Date:	⌘	23/01/2002
Category:	⌘	**B**
Release:	⌘	REL-5

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
REL-4	*(Release 4)*
REL-5	*(Release 5)*

Reason for change:	⌘	Current specification is limited to single SMS-PP
Summary of change:	⌘	Define/modify : - Framework behaviour upon concatenated short messages receipt. - API methods allowing Toolkit Applet to access concatenated short message data.
Consequences if not approved:	⌘	No standardize solution to process Concatenated Short Messages in SMS Point to Point.

Clauses affected:	⌘	§ 2, § 6.2, § 6.8, Annex A, Annex B
Other specs Affected:	⌘	[ ] Other core specifications ⌘ [ ] Test specifications [ ] O&M Specifications
Other comments:	⌘	

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

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.
- A non-specific reference to an ETS shall also be taken to refer to later versions published as an EN with the same number.
- For this Release 1999 document, references to GSM documents are for Release 1999 versions (version 8.x.y).

[1] 3GPP TR 21.905: "Abbreviations and acronyms".

[2] 3GPP TS 51.011: "Specification of the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface".

[3] 3GPP TS 11.14: "Specification of the SIM Application Toolkit for the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface".

[4] 3GPP TS 23.048: "Security Mechanisms for the SIM application toolkit; Stage 2".

[5] ISO/IEC 7816-3 (1997) " Identification cards - Integrated circuit(s) cards with contacts, Part 3: Electronic signals and transmission protocols".

[6] 3GPP TS 42.019: "Subscriber Identity Module Application Programming Interface (SIM API); Service description; Stage 1".

[7] SUN Java Card Specification "Java Card 2.1 API Specification".

[8] SUN Java Card Specification "Java Card 2.1 Runtime Environment Specification".

[9] SUN Java Card Specification "Java Card 2.1 VM Architecture Specification".

SUN Java Card Specifications can be downloaded at http://java.sun.com/products/javacard

[10] ETSI TS 101 220 "Integrated Circuit Cards (ICC); ETSI numbering system for telecommunication; Application providers (AID)".

[11] 3GPP TS 23.040: “Technical realization of the Short Message Service (SMS)”

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

6.2 Applet Triggering

[..]

EVENT_FORMATTED_SMS_PP_ENV, EVENT_UNFORMATTED_SMS_PP_ENV, EVENT_FORMATTED_SMS_PP_UPD, EVENT_UNFORMATTED_SMS_PP_UPD

A toolkit applet can be activated upon the reception of a short message.

There are two ways for a card to receive an ~~SMS :~~SMS: via the Envelope SMS-PP Data Download or the Update Record EFsms instruction.

The reception of the SMS by the toolkit applet cannot be guaranteed for the Update Record EFsms instruction.

The received SMS may ~~be :~~be:
- formatted according to TS 23.048[4] or an other protocol to identify explicitly the toolkit applet for which the message is ~~sent ;~~sent;
- unformatted or using a toolkit applet specific protocol the SIM Toolkit Framework will pass this data to all registered toolkit applets.

The Short Message may be received as Concatenated Short Messages as defined in TS 23.040[11]. It is the responsibility of the SIM Toolkit Framework to link single Short Messages together to re-assemble the original message before any further processing. The original Short Message shall be placed in one SMS TPDU TLV (with TP-UDL field coded on one octet) included in the EnvelopeHandler. The concatenation control headers used to re-assemble the short messages in the correct order shall not be present in the SMS TPDU. The TP-elements of the SMS TPDU and the Address (TS-Service-Centre-Address) shall correspond to the ones in the last received Short Message (independently of the Sequence number of Information-Element-Data).

The minimum requirement for the SIM Toolkit Framework is to process a concatenated short message with the following properties:
- the Information Element Identifier is equal to the 8-bit reference number.
- it contains uncompressed 8 bit data or uncompressed UCS2 data.

EVENT_FORMATTED_SMS_PP_ENV

~~This event is triggered by an envelope APDU containing an SMS_DATADOWNLOAD BER TLV with an SMS_TPDU simple TLV according to TS 23.048[4].~~

This event is generated when a Short Message Point to Point (Single or Concatenated) is received by Envelope SMS-PP download APDU(s) and is formatted according to TS 23.048[4].

The SIM Toolkit Framework shall:
- verify the~~ TS 23.048[4]~~ security of the Short Message as per TS 23.048[4]~~SMS TPDU~~ ;
- trigger the toolkit applet registered with the corresponding TAR defined at applet loading;
- take the optional Application Data posted by the triggered toolkit applet if present;
- secure and send the response packet.

~~The toolkit applet will only be triggered if the TAR is known and the security verified, application data will also be deciphered.~~

When the toolkit applet is triggered, data shall be provided deciphered.

EVENT_UNFORMATTED_SMS_PP_ENV

This event is generated when a Short Message Point to Point (Single or Concatenated) is received by Envelope SMS-PP download APDU(s) and is unformatted.

The registered toolkit applets will be triggered by this event and get the data transmitted in the Envelope APDU(s).~~APDU envelope SMS_DATADOWNLOAD.~~

But only the first toolkit applet triggered will be able to send back a response as defined by the rules in chapter 6.6.

EVENT_FORMATTED_SMS_PP_UPD

This event is generated when a Short Message Point to Point (Single or Concatenated) is received by Update Record EFsms APDU(s) and is formatted according to TS 23.048[4].

~~This event is triggered by Update Record EFsms with an SMS TP-UD field formatted according to TS 23.048[4].~~

The SIM Toolkit Framework ~~shall :~~shall:

- update the EFsms file with the data received, it is then up to the receiving toolkit applet to change the SMS stored in the file (i.e. the toolkit applet need to have access to the EFsms file)
- verify the ~~TS 23.048[4]~~ security of the Short Message as per TS 23.048[4] ~~SMS TPDU~~ ;
- convert the Update Record EFsms in the EnvelopeHandler a ~~TLV List, an EnvelopeHandler~~ ;
- trigger the toolkit applet registered with the corresponding TAR defined at applet loading;

When the toolkit applet is triggered, data shall be provided deciphered.

The Update Record EFsms APDU shall be converted in a TLV list as defined ~~below :~~below:

UPDATE RECORD APDU	Nb bytes	Handler TLV LIST	size
CLA, INS	2	~~Specific event~~	1
P1,P2	2	device Identity Absolute Record Number ~~rec-number~~	1
P3 = 176	1		1
Status	1	device Identity Record Status~~rec-status~~	1
TS-SCA (RP-OA)	<= 12	Address	Y
SMS TPDU	Var	SMS TPDU	Y
Padding bytes	Var		Y

The EnvelopeHandler provided to the applet shall:
- return *BTAG_SMS_PP_DOWNLOAD* to the *getEnvelopeTag()* method call;
- return the Simple TLV list length to the *getLength( )* method call ~~method call~~;
- contain the Simple TLV list :

EnvelopeHandler TLV List
Device identities
Address
SMS TPDU

The applet should use the *findTLV()* methods to get each Simple TLV.

The Device Identity Simple TLV is used to store the information about the absolute record number in the EFsms file and the value of the EFsms record status byte, and formatted as defined below:

Device identities Simple TLV
Device identities tag
Length = 02
Absolute Record Number
Record Status

With the absolute record number the toolkit applet can update EFsms in absolute mode to change the received SMS in a readable text.

For Concatenated Short Message the Absolute Record Number and the Record Status will correspond to the last Update Record EFsms APDU received.

EVENT_UNFORMATTED_SMS_PP_UPD

This event is generated when a Short Message Point to Point (Single or Concatenated) is received by Update Record EFsms APDU(s) and is unformatted.

The SIM Toolkit Framework will first update the EFsms file, convert the received APDU as described above, and then trigger all the registered toolkit applets. All of them may modify the content of EFsms (i.e. the toolkit applets need to have access to the EFsms file).

6.8 Usage of ViewHandler and EditHandler

The ViewHandler and EditHandler classes have been defined to group the properties of the system handler, and may be used in the future to provide a simple mechanism to the toolkit applet to handle TLV lists. The length of simple TLV present in a Handler TLV List shall be coded according to ISO/IEC 7816-6 [12] (e.g. coded onto 1,or 2 or 3 bytes).

Annex A (normative): Java Card SIM API

The attached files "Annex_A_java.zip" and "Annex_A_HTML.zip" contains source files for the Java Card SIM API.

[The HTML and JAVA source files will be included]

Annex B (normative): Java Card SIM API identifiers

The attached file "Annex_B_Export_files.zip" contains source files for the Java Card SIM API identifiers.

[The export files will be included]

NOTE: The export files in this annex have been generated with the following steps and tools :

- Compilation from the API java source file (.java) to the API class files (.class) with the Java compiler from the Java Development Kit version 1.2.2.
- Convertion from the API class files (.class) to the API export files (.exp) with the Java Card 2.1.2 Class File Converter (version 1.2) and the Java Development Kit 1.2.2.

List of changes to the API html and java source files

Class sim.toolkit.EnvelopeHandler

Modify the methods:

```
/**
    * Looks for the TP-UDL field in the first TPDU TLV element in the Envelope
    * data field. This method can be used on the events EVENT_FORMATTED_SMS_PP_ENV,
    * EVENT_FORMATTED_SMS_PP_UPD, EVENT_UNFORMATTED_SMS_PP_ENV,
EVENT_UNFORMATTED_SMS_PP_UPD.
    * If the element is available it becomes the TLV selected.
    *
    * @return TPUDL offset in the first TPDU TLV element if TPUDL exists.
    *     The value retrieved by using getValueByte is meaningless when the message is
    Concatenated.
    *     To recover the TP-User-Data-Length the method getUserDataLength() shall be used.~~The
    TPUD length can be recovered by using the getValueByte method in Handler class~~.
    *
    * @exception ToolkitException with the following reason codes: <ul>
    *     <li><code>UNAVAILABLE_ELEMENT</code> in case of unavailable TPDU TLV element or if
the TPUDL field does not exist</ul>
    */
   public short getTPUDLOffset() throws ToolkitException {
    return 0;
   }

   /**
    * Looks for the Secured Data from the Command Packet in the first SMS TPDU
    * or Cell Broadcast Page Simple TLV contained in the Envelope handler. This can
    * be used on the events:
    * - EVENT_FORMATTED_SMS_PP_ENV, EVENT_FORMATTED_SMS_PP_UPD, if the SMS TP-
UD is formatted
    * according to GSM03.48 Single or Concatenated Short Message.
    * - EVENT_FORMATTED_SMS_CB, if the Cell Broadcast Page is formatted according to GSM
03.48.
    * If the element is available it becomes the TLV selected.
    *
    * @return the offset of the Secured Data first byte in the first SMS TPDU or Cell Broadcast Page
TLV element. If the Secured Data length is zero the value returned shall be the SMS TPDU TLV length.
    *
    * @exception ToolkitException with the following reason codes: <ul>
    *     <li><code>UNAVAILABLE_ELEMENT</code> in case of unavailable SMS TPDU or Cell
Broadcast Page TLV element or wrong data format </ul>
    */

    public short getSecuredDataOffset() throws ToolkitException {
    return 0;
   }

   /**
    * Looks for the length of the Secured Data from the Command Packet in the first SMS TPDU
    * or Cell Broadcast Page Simple TLV contained in the Envelope handler. This can be used
    * on the events:
    * - EVENT_FORMATTED_SMS_PP_ENV, EVENT_FORMATTED_SMS_PP_UPD, if the SMS TP-
UD
    * is formatted according to GSM03.48 Single or Concatenated Short Message.
    * - EVENT_FORMATTED_SMS_CB, if the Cell Broadcast Page is formatted according to GSM
03.48.
    * If the element is available it becomes the TLV selected.
    *
```

```
        * @return the length of the Secured Data contained in the first SMS TPDU or Cell Broadcast Page
TLV element (without padding bytes). If the Secured Data length is zero, no exception shall be thrown.
        *
        * @exception ToolkitException with the following reason codes: <ul>
        *      <li><code>UNAVAILABLE_ELEMENT</code> in case of unavailable SMS TPDU or Cell
Broadcast Page TLV element or wrong data format </ul>
        */

        public short getSecuredDataLength() throws ToolkitException {
        return 0;
       }
```

Add the method:

- getUserDataLength()

```
/**
    * Looks for the TP-User Data field in the first TPDU TLV element contained in the Envelope
Handler
    * This method can be used on the events EVENT_FORMATTED_SMS_PP_ENV,
    * EVENT_FORMATTED_SMS_PP_UPD, EVENT_UNFORMATTED_SMS_PP_ENV,
EVENT_UNFORMATTED_SMS_PP_UPD.
    * If the element is available it becomes the TLV selected.
    *
    * @return the length of the User Data contained in the first SMS TPDU TLV element.
    *
    * @exception ToolkitException with the following reason codes: <ul>
    *      <li><code>UNAVAILABLE_ELEMENT</code> in case of unavailable TPDU TLV element or
wrong data format </ul>
    */

public short getUserDataLength() throws ToolkitException {
    return 0;
}
```

3GPP T3 Meeting #22
Marbella, Spain, 22 - 25 Jan 2002

Tdoc T3-020087

CR-Form-v3

CHANGE REQUEST

⌘ **43.019** **CR** **014** ⌘ rev **-** ⌘ Current version: **5.1.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM [X] ME/UE [ ] Radio Access Network [ ] Core Network [ ]

Title:	⌘	Change in the EnvelopResponseHandler behavior
Source:	⌘	T3
Work item code:	⌘	USAT1-API-JAVA
Date:	⌘	23/01/02
Category:	⌘	**B**
Release:	⌘	REL-5

Use one of the following categories:
F *(correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
REL-4	*(Release 4)*
REL-5	*(Release 5)*

Reason for change:	⌘	Improve flexibility on the response management.
Summary of change:	⌘	The chapter 6.2 with the event *EVENT_UNFORMATTED_SMS_PP_ENV* is restricting the availability of the EnvelopeResponseHandler defined in chapter 6.6 This limitation is removed in the *EVENT_UNFORMATTED_SMS_PP_ENV* description and add a note in the *EVENT_UNRECOGNIZED_ENVELOPE* and *EVENT_UNFORMATTED_SMS_PP_ENV* description.
Consequences if not approved:	⌘	

Clauses affected:	⌘	§ 6.2		
Other specs Affected:	⌘	[ ] Other core specifications [ ] Test specifications [ ] O&M Specifications	⌘	
Other comments:	⌘			

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be

downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

EVENT_UNFORMATTED_SMS_PP_ENV

The registered toolkit applets will be triggered by this event and get the data transmitted in the APDU envelope SMS_DATADOWNLOAD.

~~But only the first toolkit applet triggered will be able to send back a response as defined by the rules in chapter 6.6.~~

Note : As a consequence of the EnvelopeResponseHandler availability rules specified in clause 6.6, only the first triggered toolkit applet is guaranteed to be able to send back a response.

[…]

EVENT_UNRECOGNIZED_ENVELOPE

The applet registered to this event shall be triggered by the framework if the BER-TLV tag contained in the ENVELOPE APDU is not defined in the associated release of TS 11.14 [3] and if no corresponding constant is defined in the list of the ToolkitConstants interface. The unrecognized Envelope event will allow a toolkit applet to handle the evolution of the TS 11.14 specification.

Note : As a consequence of the EnvelopeResponseHandler availability rules specified in clause 6.6, only the first triggered toolkit applet is guaranteed to be able to send back a response.

3GPP T3 Meeting #22
Marbella, Spain, 22 - 25 Jan 2002

Tdoc T3-020101

CR-Form-v3

CHANGE REQUEST

⌘ **43.019** **CR** **015** ⌘ rev **-** ⌘ Current version: **5.1.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM [X] ME/UE [ ] Radio Access Network [ ] Core Network [ ]

Title:	⌘	Handler availability
Source:	⌘	T3
Work item code:	⌘	USAT1-API-JAVA
Date:	⌘	23/01/02
Category:	⌘	**C**
Release:	⌘	REL-5

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
REL-4	*(Release 4)*
REL-5	*(Release 5)*

Reason for change:	⌘	The minimum availability of the handlers is not clear when proactive session is ongoing.
Summary of change:	⌘	- Re-order the chapter 6.6 for clarity and understanding. - Modify in table 1 the availability of ProactiveHandler and ProactiveResponseHandler to allow applets triggerring if a proactive session is ongoing. - Remove the limitation:" The reception of the SMS by the toolkit applet cannot be guaranteed for the Update Record EFsms instruction"
Consequences if not approved:	⌘	

Clauses affected:	⌘	$6.2, $ 6.6		
Other specs Affected:	⌘	[ ] Other core specifications [ ] Test specifications [ ] O&M Specifications	⌘	
Other comments:	⌘			

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

6.2 Applet Triggering

[…..]

EVENT_FORMATTED_SMS_PP_ENV, EVENT_UNFORMATTED_SMS_PP_ENV, EVENT_FORMATTED_SMS_PP_UPD, EVENT_UNFORMATTED_SMS_PP_UPD

A toolkit applet can be activated upon the reception of a short message.

There are two ways for a card to receive an SMS : via the Envelope SMS-PP Data Download or the Update Record EFsms instruction.

~~The reception of the SMS by the toolkit applet cannot be guaranteed for the Update Record EFsms instruction.~~

The received SMS may be :
- formatted according to TS 03.48[4] or an other protocol to identify explicitly the toolkit applet for which the message is sent ;
- unformatted or using a toolkit applet specific protocol the SIM Toolkit Framework will pass this data to all registered toolkit applets.

6.6 Handler availability

The system handlers : ProactiveHandler, ProactiveResponseHandler, EnvelopeHandler and EnvelopeResponseHandler are Temporary JCRE Entry Point Object as defined in the Java Card Runtime Environment Specification [8].

The following rules define the availability of the system handlers and the lifetime of their content. They are generic rules and may vary with the event that triggers the toolkit applet.

ProactiveHandler:

- The ProactiveHandler is valid from the invocation to the termination of the processToolkit method.
- If a proactive command is pending the ProactiveHandler may not be available.
- At the processToolkit method invocation the TLV-List is cleared.
- At the call of it's init method the content is cleared and then initialised.
- After a call to ProactiveHandler.send method the handler will remain unchanged (i.e. previously send proactive command) until the ProactiveHandler.init or appendTLV methods are called.

ProactiveResponseHandler:

- The ProactiveResponseHandler may not be available before the first call to ProactiveHandler.send method, if available the content is cleared.
- The ProactiveResponseHandler is available after the first call to the ProactiveHandler.send method to the termination of the processToolkit method.
- If a proactive command is pending the ProactiveResponseHandler may not be available.
- The ProactiveResponseHandler content is changed after the call to ProactiveHandler.send method and remains unchanged until next call to the ProactiveHandler.send method.

EnvelopeHandler:

- The EnvelopeHandler and its content are available for all triggered toolkit applets (see Table1), from the invocation to the termination of their processToolkit method.
- The SIM Toolkit Framework guarantees that all registered toolkit applet are triggered and receive the data.

EnvelopeResponseHandler:

- The EnvelopeResponseHandler is available for all triggered toolkit applets, until a toolkit applet has posted an envelope response or sent a proactive command. After a call to the post method the handler is no longer available.
- The EnvelopeResponseHandler content must be posted before the first invocation of a ProactiveHandler.send method or before the termination of the processToolkit, so that the GSM applet can offer these data to the ME (eg 9Fxx/9Exx). After the first invocation of the ProactiveHandler.send method the EnvelopeResponseHandler is no more available.

The following diagram illustrates these rules.

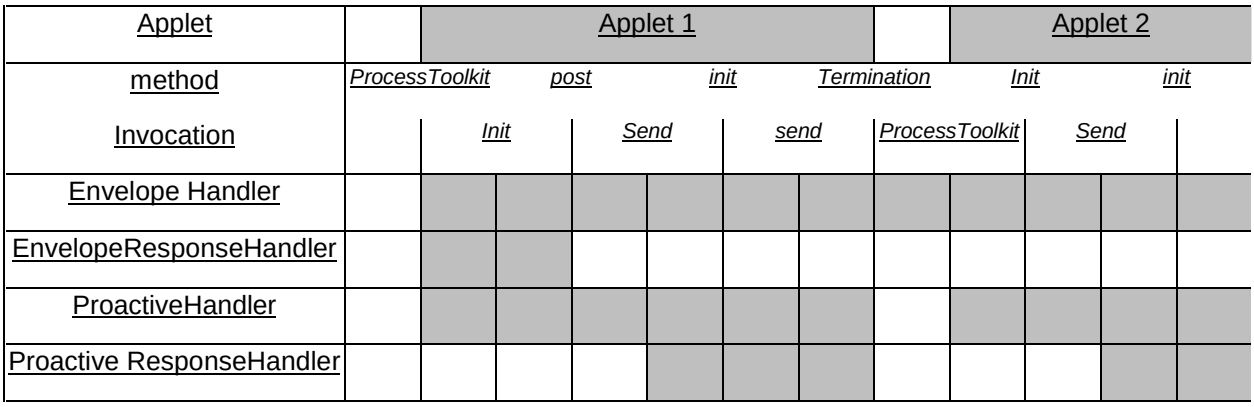


Figure 5: Typical handler availability for toolkit applets (see Table 1 for detail)

The following table describes the minimum availability of the handlers for all the events at the invocation of the processToolkit method of the toolkit applet.

Table 1: Handler availability for each event

EVENT_	**Reply busy allowed**	**~~ProactiveHandler~~ ~~ProactiveResponseHandler~~**	**Envelope Handler**	**EnvelopeResponseHandler**	**Nb of triggered / registrered Applet**
_FORMATTED_SMS_PP_ENV	Y	~~Y/N (see Note 2)Y~~	Y	Y	1 / n (per TAR)
_FORMATTED_SMS_PP_UPD	N	~~Y/N (see Note 2)Y~~	Y	N	1 / n (per TAR)
_UNFORMATTED_SMS_PP_ENV	Y	~~Y/N (see Note 2)Y~~	Y	Y	n / n
_UNFORMATTED_SMS_PP_UPD	N	~~Y/N (see Note 2)Y~~	Y	N	n / n
_FORMATTED_SMS_CB	Y	~~Y/N (see Note 2)Y~~	Y	N	1 / n (per TAR)
_UNFORMATTED_SMS_CB	Y	~~Y/N (see Note 2)Y~~	Y	N	n / n
_MENU_SELECTION	Y	~~Y/N (see Note 2)Y~~	Y	N	1 / n (per Item Id)
_MENU_SELECTION_HELP_REQUEST	Y	~~Y/N (see Note 2)Y~~	Y	N	1 / n (per Item Id)
_CALL_CONTROL	N	~~Y/N (see Note 2)~~	Y	Y	1 / 1
_SMS_MO_CONTROL	N	~~Y/N (see Note 2)~~	Y	Y	1 / 1
_TIMER_EXPIRATION	Y	~~Y/N (see Note 2)Y~~	Y	N	1/ 8 (per timer) (see Note 1)
_EVENT_DOWNLOAD					
_MT_CALL	Y	~~Y/N (see Note 2)Y~~	Y	N	n / n
_CALL_CONNECTED	Y	~~Y/N (see Note 2)Y~~	Y	N	n / n
_CALL_DISCONNECTED	Y	~~Y/N (see Note 2)Y~~	Y	N	n / n
_LOCATION_STATUS	Y	~~Y/N (see Note 2)Y~~	Y	N	n / n
_USER_ACTIVITY	Y	~~Y/N (see Note 2)Y~~	Y	N	n / n
_IDLE_SCREEN_AVAILABLE	Y	~~Y/N (see Note 2)Y~~	Y	N	n / n
_LANGUAGE_SELECTION	Y	~~Y/N (see Note 2)Y~~	Y	N	n / n
_BROWSER_TERMINATION	Y	~~Y/N (see Note 2)Y~~	Y	N	n / n
_CARD_READER_STATUS	Y	~~Y/N (see Note 2)Y~~	Y	N	n / n
_UNRECOGNIZED_ENVELOPE	Y	~~Y/N (see Note 2)Y~~	Y	Y	n / n
_STATUS_COMMAND	N	~~Y/N (see Note 2)~~	N	N	n / n
_PROFILE_DOWNLOAD	N	~~Y/N (see Note 2)~~	N	N	n / n
_EVENT_COMMAND_AFTER_SELECT	N	~~N~~	N	N	n/n

NOTE 1: One toolkit applet can register to several timers, but a timer can only be allocated to one toolkit applet.

~~NOTE 2: Y/N means that handlers may / may not be available depending whether a proactive session is ongoing.~~

~~The following rules define the minimum requirement for the availability of the system handlers and the lifetime of their content.~~

ProactiveHandler:

- The ProactiveHandler is valid from the invocation to the termination of the processToolkit method.
- If a proactive command is pending the ProactiveHandler may not be available.
- At the processToolkit method invocation the TLV-List is cleared.
- At the call of it's init method the content is cleared and then initialised.
- After a call to ProactiveHandler.send method the handler will remain unchanged (i.e. previously send proactive command) until the ProactiveHandler.init or appendTLV methods are called.

ProactiveResponseHandler:

- The ProactiveResponseHandler may not be available before the first call to ProactiveHandler.send method, if available the content is cleared.
- The ProactiveResponseHandler is available after the first call to the ProactiveHandler.send method to the termination of the processToolkit method.
- If a proactive command is pending the ProactiveResponseHandler may not be available.
- The ProactiveResponseHandler content is changed after the call to ProactiveHandler.send method and remains unchanged until next call to the ProactiveHandler.send method.

EnvelopeHandler:

- The EnvelopeHandler and its content are available for all triggered toolkit applets (see Table1), from the invocation to the termination of their processToolkit method.
- The SIM Toolkit Framework guarantees that all registered toolkit applet are triggered and receive the data.

EnvelopeResponseHandler:

- The EnvelopeResponseHandler is available for all triggered toolkit applets, until a toolkit applet has posted an envelope response or sent a proactive command. After a call to the post method the handler is no longer available.
- The EnvelopeResponseHandler content must be posted before the first invocation of a ProactiveHandler.send method or before the termination of the processToolkit, so that the GSM applet can offer these data to the ME (eg 9Fxx/9Exx). After the first invocation of the ProactiveHandler.send method the EnvelopeResponseHandler is no more available.

The following diagram illustrates these rules.

Applet		Applet 1							Applet 2			
Method Invocation	*ProcessToolkit*	*Post* *Init*			*Termination*	*init*		*init*				
		Init		*Send*		*send*		*ProcessToolkit*		*Send*		
Envelope Handler												
EnvelopeResponseHandler												
ProactiveHandler												
Proactive ResponseHandler												

Figure 5: Typical handler availability for toolkit applets (see Table 1 for detail)