## 5.5 CAT Runtime Environment

### 5.5.1 Minimum Handler Availability

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

#### 5.5.1.1 ProactiveHandler

Test Area Reference: Cre\_Mha\_Pahd.

##### 5.5.1.1.1 Conformance requirement

5.5.1.1.1.1 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\_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\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION

EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION

EVENT\_UNRECOGNIZED\_ENVELOPE

EVENT\_STATUS\_COMMAND

EVENT\_CALL\_CONTROL\_BY\_NAA

EVENT\_PROFILE\_DOWNLOAD

EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE

EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS

EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE

EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED

EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION

EVENT\_PROACTIVE\_HANDLER\_AVAILABLE

EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE

EVENT\_EVENT\_BROWSING\_STATUS

EVENT\_EXTERNAL\_FILE\_UPDATE

* CRRN2: A ProactiveHandler is considered available when no HANDLER\_NOT\_AVAILABLE *ToolkitException* is thrown when the corresponding *getTheHandler()* method is called or a method of the handler is called.
* CRRN3: When available the *ProactiveHandler* shall remain available until the termination of the *processToolkit()* method.
* CRRN4: If a proactive command is pending the *ProactiveHandler* may not be available.

5.5.1.1.1.2 Parameter errors

No requirements.

5.5.1.1.1.3 Context errors

* CRRC1: The ProactiveHandler 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\_FIRST\_COMMAND\_AFTER\_ATR

EVENT\_APPLICATION\_DESELECT

* CRRC2: The ProactiveHandler shall not be available if the Terminal Profile command has not yet been processed by the CAT Runtime Environment
* CRRC3: The ProactiveHandler shall not be available if the *getTheHandler()* method is not called, directly or indirectly, from the applet's *processToolkit*() method.

##### 5.5.1.1.2 Test area files

Test Source: Test\_Cre\_Mha\_Pahd.java.

Test Applet: Cre\_Mha\_Pahd\_1.java.

Cre\_Mha\_Pahd\_2.java.

Cre\_Mha\_Pahd\_3.java.

Cap File: Cre\_Mha\_Pahd.cap.

##### 5.5.1.1.3 Test coverage

| **CRR Number** | **Test Case Number** |
| --- | --- |
| CRRN1 | 2 to 23, 45, 46 |
| CRRN2 | 1 to 22,45,46 |
| CRRN3 | 2 to 22, 45, 46 |
| CRRN4 | Not testable |
| CRRC1 | 1, 24 |
| CRRC2 | 25 to 44 or also tested in TestCases 25 to 44 in Cre\_Mha\_Prhd |
| CRRC3 | 47 |

##### 5.5.1.1.4 Test procedure

| Id | Description | API /Framework Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applets registration to all events and Proactive Handler availability with EVENT\_FIRST\_COMMAND\_AFTER\_ATR  Applet1 is registered to all events defined in TS 102 241 [9] except to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE.  Using the methods initMenuEntry() for EVENT\_MENU\_SELECTION, requestPollInterval() for EVENT\_STATUS\_COMMAND, allocateTimer() for EVENT\_TIMER\_EXPIRATION, allocateServiceIdentifier() for EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION and setEventList() for the rest of the events.  Applet1 is registered on EVENT\_EXTERNAL\_FILE\_UPDATE on update of EFTARU of the UICC file system  Applet2 is registered to all events defined in TS 102 241 [9], except to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE and EVENT\_CALL\_CONTROL\_BY\_NAA.  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 on EVENT\_EXTERNAL\_FILE\_UPDATE on update of EFTARU of the UICC file system  The priority of Applet1 is higher than priority of Applet2  1- Select MF  2- Applet1 gets the Proactive Handler.  Applet1 is deregistered from EVENT\_FIRST\_COMMAND\_AFTER\_ATR.  3- Applet2 gets the Proactive Handler  Applet2 is deregistered to EVENT\_FIRST\_COMMAND\_AFTER\_ATR. | 1- Applet1 is triggered by EVENT\_FIRST\_COMMAND\_AFTER\_ATR  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet1 finalizes  Applet2 is triggered by EVENT\_FIRST\_COMMAND\_AFTER\_ATR  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet2 finalizes |  |
| 2 | Proactive Handler availability with EVENT\_PROFILE\_DOWNLOAD  1- Terminal Profile command is sent to the UICC without the facility of SET UP 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 by EVENT\_PROFILE\_DOWNLOAD  2- No exception is thrown  Applet1 finalizes.  Applet2 is triggered by EVENT\_PROFILE\_DOWNLOAD  3- No exception is thrown  Applet2 finalizes |  |
| 3 | Proactive Handler availability with EVENT\_MENU\_SELECTION\_HELP\_REQUEST  Perform UICC initialization with all the facilities supported, without facility SET\_UP\_EVENT\_LIST  1- Envelope menu selection with help request is sent to the UICC  2- Applet1 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes |  |
| 4 | Proactive Handler availability with EVENT\_MENU\_SELECTION  1- Envelope menu selection is sent to the UICC  2- Applet1 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes |  |
| 5 | Proactive Handler availability with EVENT\_TIMER\_EXPIRATION  1- Timer Id =1  Envelope Timer Expiration is sent to the UICC  2- Applet1 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes |  |
| 6 | Proactive Handler availability with EVENT\_CALL\_CONTROL\_BY\_NAA  1- Envelope call control by NAA is sent to the UICC  2- Applet1 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes |  |
| 7 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  1- Envelope event download mt call is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3-No exception is thrown  Applet2 finalizes |  |
| 8 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_CALL\_CONNECTED  1- Envelope event download call connected is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 9 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_CALL\_DISCONNECTED  1- Envelope event download call disconnected is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 10 | Applets triggering with EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS  1- Envelope event download location status is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 11 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY  1- Envelope event download user activity is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 12 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_IDLE\_SCREEN\_AVAILABLE  1- Envelope event download idle screen available is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 13 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_CARD\_READER\_STATUS  1- Envelope event download card reader status is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 14 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION  1- Envelope event download language selection is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 15 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION  1- Envelope event download browser termination is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 16 | Proactive Handler availability with EVENT\_STATUS\_COMMAND  1- Status command is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 17 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE  1- Applet1 builds a proactive command OPEN CHANNEL proactiveHandler.send() method is called.  2- An Envelope Event Download Data Available is sent to the UICC, with channelId=01.  3- Applet1 gets the Proactive Handler | 2- Applet1 is triggered  3- No exception is thrown  Applet1 finalizes | 1- OPEN CHANNEL proactive Command is fetched  TERMINAL RESPONSE is issued with Channel Id = 01 |
| 18 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS  1- An Envelope Event Download Channel Status is sent to the UICC, with ChannelId=01  2- Applet1 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes |  |
| 19 | Proactive Handler availability with UNRECOGNIZED\_ENVELOPE  1- An unrecognized Envelope (BER TLV Tag unrecognized) is sent to the UICC  2- Applet1 gets the Proactive Handler  3-Applet2 gets the Proactive Handler | 1- Applet1 is triggered    2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 20 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE  1- An envelope event download access technology change is sent to the UICC  2- Applet1 gets the Proactive Handler  3-Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 21 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED  1- An envelope event download display parameter changed is sent to the UICC  2- Applet1 gets the Proactive Handler  4-Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  3- Applet1 sends a DECLARE SERVICE (add) proactive command with its service identifier  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  Applet2 finalizes | 3- DECLARE SERVICE (add) proactive command is fetched  Successful TERMINAL RESPONSE is issued |
| 22 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION  1- An envelope event download local connection is sent to the UICC, with the  allocated service identifier  2- Applet1 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes |  |
| 23 | Proactive Handler availability with EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  1- Envelope event download call connected is sent to the UICC  2-The display Text proactive command is fetch and the terminal response is sent  3-Applet1 gets the Proactive Handler  4-Applet2 gets the Proactive Handler | 1- Applet1 is triggered and registers to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  Applet1 finalizes  Applet2 is triggered, registers to  EVENT\_PROACTIVE\_HANDLER\_AVAILABLE and sends a Display Test proactive command  2- Applet1 is triggered by EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered by EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  4- No exception is thrown  Applet2 finalizes |  |
| 24 | Proactive Handler availability with EVENT\_APPLICATION\_DESELECT  1- Select for activation ADF1  2- Select for termination ADF1  3- Applet1 gets the Proactive Handler  Applet1 deregisters to EVENT\_APPLICATION\_DESELECT.  4-Applet2 gets the Proactive Handler  Applet2 deregisters to EVENT\_APPLICATION\_DESELECT | 2- Applet1 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 25 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_MENU SELECTION\_HELP\_REQUEST  1- Reset the card without sending the Terminal Profile  2- Envelope menu selection with help request is sent to the UICC  3- Applet1 gets the Proactive Handler | 2- Applet1 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 26 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_MENU SELECTION  1- Envelope menu selection is sent to the UICC  2- Applet1 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 27 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_TIMER\_EXPIRATION  1- Timer Id =1  Envelope Timer Expiration is sent to the UICC  2- Applet1 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 28 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_CALL\_CONTROL\_BY\_NAA  1- Envelope call control by NAA is sent to the UICC  2- Applet1 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet1 finalizes |  |
| 29 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  1- Envelope event download mt call is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 30 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_CALL\_CONNECTED  1- Envelope event download call connected is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 31 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_CALL\_DISCONNECTED  1- Envelope event download call disconnected is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 32 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS  1- Envelope event download location status is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 33 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY  1- Envelope event download user activity is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 34 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_IDLE\_SCREEN\_AVAILABLE  1- Envelope event download idle screen available is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 35 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_CARD\_READER\_STATUS  1- Envelope event download card reader status is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 36 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION  1- Envelope event download language selection is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 37 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION  1- Envelope event download browser termination is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 38 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_STATUS\_COMMAND  1- Status command is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 39 | The ProactiveHandler is not available before the Terminal Profile with UNRECOGNIZED\_ENVELOPE  1- An unrecognized Envelope (BER TLV Tag unrecognized) is sent to the UICC  2- Applet1 gets the Proactive Handler  3-Applet2 gets the Proactive Handler | 1- Applet1 is triggered    2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 40 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE  1- An envelope event download access technology change is sent to the UICC  2- Applet1 gets the Proactive Handler  3-Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 41 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED  1- An envelope event download display parameter changed is sent to the UICC  2- Applet1 gets the Proactive Handler  3-Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 registers to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE then finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 registers to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE then finalizes |  |
| 42 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_PROACTIVE\_HANDLER\_AVAILABLE | Applet1 and Applet2 are not triggered by EVENT\_PROACTIVE\_HANDLER\_AVAILABLE |  |
| 43 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE  1- Envelope event download network search mode change is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 deregisters from  EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 deregisters from  EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  Applet2 finalizes |  |
| 44 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_BROWSING STATUS  1- Envelope event download browsing status is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 45 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE  0- Reset card and sendTerminal Profile without facility SETUP\_EVENT\_LIST  1- Envelope event download network search mode change is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 46 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_BROWSING STATUS  1- Envelope event download browsing status is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 47 | The ProactiveHandler is not available outside the processToolkit() method  1- Install Applet3. In its install method, Applet3 gets the ProactiveHandler in a Try/Catch session  2- Select Applet3  3- Applet3 gets the ProactiveHandler | 1- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  2- Applet3 is triggered by its process() method  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown |  |

#### 5.5.1.2 ProactiveResponseHandler

Test Area Reference: Cre\_Mha\_Prhd.

##### 5.5.1.2.1 Conformance requirement

5.5.1.2.1.1 Normal execution

* CRRN1: The ProactiveResponseHandler is available as soon as the ProactiveHandler is available and remains available untill the termination of the processToolkit method for the following events:

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\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION

EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION

EVENT\_UNRECOGNIZED\_ENVELOPE

EVENT\_STATUS\_COMMAND

EVENT\_CALL\_CONTROL\_BY\_NAA

EVENT\_PROFILE\_DOWNLOAD

EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE

EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS

EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE

EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED

EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION

EVENT\_PROACTIVE\_HANDLER\_AVAILABLE

EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE

EVENT\_EVENT\_BROWSING\_STATUS

EVENT\_EXTERNAL\_FILE\_UPDATE

* CRRN2: A ProactiveResponseHandler is considered available when no HANDLER\_NOT\_AVAILABLE ToolkitException is thrown when the corresponding getTheHandler() method is called or a method of the handler is called.

5.5.1.2.1.2 Parameter errors

No requirements.

5.5.1.2.1.3 Context errors

* CRRC1: The ProactiveResponseHandler 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\_FIRST\_COMMAND\_AFTER\_ATR

EVENT\_APPLICATION\_DESELECT

* CRRC2: The ProactiveResponseHandler shall not be available if the ProactiveHandler is not available.
* CRRC3: The ProactiveResponseHandler shall not be available if the *getTheHandler()* method is not called, directly or indirectly, from the applet's *processToolkit*() method.

##### 5.5.1.2.2 Test area files

Test Source: Test\_Cre\_Mha\_Prhd.java.

Test Applet: Cre\_Mha\_Prhd\_1.java.

Cre\_Mha\_Prhd\_2.java.

Cre\_Mha\_Prhd\_3.java.

Cap File: Cre\_Mha\_Prhd.cap.

##### 5.5.1.2.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 2 to 23, 45, 46 |
| CRRN2 | 1 to 22,45,46 |
| CRRC1 | 1, 24 |
| CRRC2 | 25 to 44 |
| CRRC3 | 47 |

##### 5.5.1.2.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applets registration to all events and Proactive Handler availability with EVENT\_FIRST\_COMMAND\_AFTER\_ATR  Applet1 is registered to all events defined in TS 102 241 [9] except to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE.  Using the methods initMenuEntry() for EVENT\_MENU\_SELECTION, requestPollInterval() for EVENT\_STATUS\_COMMAND, allocateTimer() for EVENT\_TIMER\_EXPIRATION, allocateServiceIdentifier() for EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION and setEventList() for the rest of the events.  Applet1 is registered on EVENT\_EXTERNAL\_FILE\_UPDATE on update of EFTARU of the UICC file system  Applet2 is registered to all events defined in TS 102 241 [9], except to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE and EVENT\_CALL\_CONTROL\_BY\_NAA.  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 on EVENT\_EXTERNAL\_FILE\_UPDATE on update of EFTARU of the UICC file system  The priority of Applet1 is higher than priority of Applet2  1- Select MF  2- Applet1 gets the Proactive Handler.  3- Applet2 gets the Proactive Response Handler  Applet1 is deregistered from EVENT\_FIRST\_COMMAND\_AFTER\_ATR.  4- Applet2 get the Proactive Handler  5- Applet2 gets the Proactive Response Handler  Applet2 is deregistered to EVENT\_FIRST\_COMMAND\_AFTER\_ATR. | 1- Applet1 is triggered by EVENT\_FIRST\_COMMAND\_AFTER\_ATR  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered by EVENT\_FIRST\_COMMAND\_AFTER\_ATR  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  5- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet2 finalizes |  |
| 2 | Proactive Handler availability with EVENT\_PROFILE\_DOWNLOAD  1- Terminal Profile command is sent to the UICC without the facility of SET UP EVENT LIST, POLL INTERVAL,SET UP IDLE MODE TEXT and SET UP MENU.  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  Applet1 is deregistered to EVENT\_PROFILE\_DOWNLOAD  4- Applet2 gets the Proactive Handler  5-Applet2 gets the Proactive Respones Handler  Applet2 is deregistered to EVENT\_PROFILE\_DOWNLOAD | 1- Applet1 is triggered by EVENT\_PROFILE\_DOWNLOAD  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes.  Applet2 is triggered by EVENT\_PROFILE\_DOWNLOAD  4- No exception is thrown  5- No exception is thrown  Applet2 finalizes |  |
| 3 | Proactive Handler availability with EVENT\_MENU\_SELECTION\_HELP\_REQUEST  Perform UICC initialization with all the facilities supported, without facility SET\_UP\_EVENT\_LIST  1- Envelope menu selection with help request is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes |  |
| 4 | Proactive Handler availability with EVENT\_MENU\_SELECTION  1- Envelope menu selection is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes |  |
| 5 | Proactive Handler availability with EVENT\_TIMER\_EXPIRATION  1- Timer Id =1  Envelope Timer Expiration is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes |  |
| 6 | Proactive Handler availability with EVENT\_CALL\_CONTROL\_BY\_NAA  1- Envelope call control by NAA is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes |  |
| 7 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  1- Envelope event download mt call is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4-No exception is thrown  5-No exception is thrown  Applet2 finalizes |  |
| 8 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_CALL\_CONNECTED  1- Envelope event download call connected is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  5. No exception is thrown  Applet2 finalizes |  |
| 9 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_CALL\_DISCONNECTED  1- Envelope event download call disconnected is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  5- No exception is thrown  Applet2 finalizes |  |
| 10 | Applets triggering with EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS  1- Envelope event download location status is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  5- No exception is thrown  Applet2 finalizes |  |
| 11 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY  1- Envelope event download user activity is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  5- No exception is thrown  Applet2 finalizes |  |
| 12 | Proactive Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_CARD\_READER\_STATUS  1-Envelope event download card reader status is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  5- No exception is thrown  Applet2 finalizes |  |
| 13 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_CARD\_READER\_STATUS  1- Envelope event download card reader status is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  5- No exception is thrown  Applet2 finalizes |  |
| 14 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION  1- Envelope event download language selection is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  5- No exception is thrown  Applet2 finalizes |  |
| 15 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION  1- Envelope event download browser termination is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  5- No exception is thrown  Applet2 finalizes |  |
| 16 | Proactive Handler availability with EVENT\_STATUS\_COMMAND  1- Status command is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  5- No exception is thrown  Applet2 finalizes |  |
| 17 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE  1- Applet1 builds a proactive command OPEN CHANNEL proactiveHandler.send() method is called.  2- An Envelope Event Download Data Available is sent to the UICC, with channelId=01.  3- Applet1 gets the Proactive Handler  4- Applet1 gets the Proactive Response Handler | 2- Applet1 is triggered  3- No exception is thrown  4- No exception is thrown  Applet1 finalizes | 1- OPEN CHANNEL proactive command is fetched  TERMINAL RESPONSE is issued with Channel Id = 01 |
| 18 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS  1- An Envelope Event Download Channel Status is sent to the UICC, with ChannelId=01  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes |  |
| 19 | Proactive Handler availability with UNRECOGNIZED\_ENVELOPE  1- An unrecognized Envelope (BER TLV Tag unrecognized) is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered    2- No exception is thrown  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  5- No exception is thrown  Applet2 finalizes |  |
| 20 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE  1- An envelope event download access technology change is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  5- No exception is thrown  Applet2 finalizes |  |
| 21 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED  1- An envelope event download display parameter changed is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  5- Applet2 gets the Proactive Handler  6- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  4- Applet1 sends a DECLARE SERVICE (add) proactive command with its service identifier  Applet1 finalizes  Applet2 is triggered  5- No exception is thrown  6- No exception is thrown  Applet2 finalizes | 4- DECLARE SERVICE (add) proactive command is fetched  Successful TERMINAL RESPONSE is issued |
| 22 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION  1- An envelope event download local connection is sent to the UICC, with the  allocated service identifier  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes |  |
| 23 | Proactive Handler availability with EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  1- Envelope event download call connected is sent to the UICC  2- The display Text proactive command is fetch and the terminal response is sent  3- Applet1 gets the Proactive Handler  4- Applet1 gets the Proactive Response Handler  5- Applet2 gets the Proactive Handler  6- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered and registers to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  Applet1 finalizes  Applet2 is triggered, registers to  EVENT\_PROACTIVE\_HANDLER\_AVAILABLE and sends a Display Test proactive command  2- Applet1 is triggered by EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  3- No exception is thrown  4- No exception is thrown  Applet1 finalizes  Applet2 is triggered by EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  4- No exception is thrown  5- No exception is thrown  Applet2 finalizes |  |
| 24 | Proactive Handler availability with EVENT\_APPLICATION\_DESELECT  1- Select for activation ADF1  2- Select for termination ADF1  3- Applet1 gets the Proactive Handler  4- Applet1 gets the Proactive Response Handler  Applet1 deregisters to EVENT\_APPLICATION\_DESELECT.  5-Applet2 gets the Proactive Handler  6-Applet2 gets the Proactive Response Handler  Applet2 deregisters to EVENT\_APPLICATION\_DESELECT. | 2- Applet1 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  5- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  6- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 25 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_MENU SELECTION\_HELP\_REQUEST  1- Reset the card without sending the Terminal Profile  2- Envelope menu selection with help request is sent to the UICC  3- Applet1 gets the Proactive Handler  4- Applet1 gets the Proactive Response Handler | 2- Applet1 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 26 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_MENU SELECTION  1- Envelope menu selection is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 27 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_TIMER\_EXPIRATION  1- Timer Id =1  Envelope Timer Expiration is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 28 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_CALL\_CONTROL\_BY\_NAA  1- Envelope call control by NAA is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 29 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  1- Envelope event download mt call is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  5-A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 30 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_CALL\_CONNECTED  1- Envelope event download call connected is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  5-A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 31 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_CALL\_DISCONNECTED  1- Envelope event download call disconnected is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  5-A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 32 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS  1- Envelope event download location status is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  5- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 33 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY  1- Envelope event download user activity is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  5-A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 34 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_IDLE\_SCREEN\_AVAILABLE  1- Envelope event download idle screen available is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  5- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet2 finalizes |  |
| 35 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_CARD\_READER\_STATUS  1- Envelope event download card reader status is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3-A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  5- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet2 finalizes |  |
| 36 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION  1- Envelope event download language selection is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3--A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  5-A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet2 finalizes |  |
| 37 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION  1- Envelope event download browser termination is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet2 gets the Proactive Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 38 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_STATUS\_COMMAND  1- Status command is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3-A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  5A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet2 finalizes |  |
| 39 | The ProactiveHandler is not available before the Terminal Profile with UNRECOGNIZED\_ENVELOPE  1- An unrecognized Envelope (BER TLV Tag unrecognized) is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered    2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  5- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet2 finalizes |  |
| 40 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE  1- An envelope event download access technology change is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3-A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  5-A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet2 finalizes |  |
| 41 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED  1- An envelope event download display parameter changed is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3-A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet1 registers to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE then finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  5- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet2 registers to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE then finalizes |  |
| 42 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_PROACTIVE\_HANDLER\_AVAILABLE | Applet1 and Applet2 are not triggered by EVENT\_PROACTIVE\_HANDLER\_AVAILABLE |  |
| 43 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE  1- Envelope event download network search mode change is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet1 deregisters from  EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  5- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet2 deregisters from  EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  Applet2 finalizes |  |
| 44 | The ProactiveHandler is not available before the Terminal Profile with EVENT\_EVENT\_DOWNLOAD\_BROWSING STATUS  1- Envelope event download browsing status is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  5- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet2 finalizes |  |
| 45 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE  0- Reset card and sendTerminal Profile without facility SETUP\_EVENT\_LIST  1- Envelope event download network search mode change is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  5- No exception is thrown  Applet2 finalizes |  |
| 46 | Proactive Handler availability with EVENT\_EVENT\_DOWNLOAD\_BROWSING STATUS  1- Envelope event download browsing status is sent to the UICC  2- Applet1 gets the Proactive Handler  3- Applet1 gets the Proactive Response Handler  4- Applet2 gets the Proactive Handler  5- Applet2 gets the Proactive Response Handler | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  5- No exception is thrown  Applet2 finalizes |  |
| 47 | The ProactiveHandler is not available outside the processToolkit() method  1- Install Applet3. In its install method, Applet3 gets the Proactive Handler and the Proactive Response Handler in a Try/Catch session  2- Select Applet3  3- Applet3 gets the ProactiveHandler and the Proactive Response Handler. | 1- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  2- Applet3 is triggered by its process() method  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown |  |

#### 5.5.1.3 EnvelopeHandler

Test Area Reference: Cre\_Mha\_Enhd.

##### 5.5.1.3.1 Conformance requirement

5.5.1.3.1.1 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\_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\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION

EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION

EVENT\_UNRECOGNIZED\_ENVELOPE

EVENT\_CALL\_CONTROL\_BY\_NAA

EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE

EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS

EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE

EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED

EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION

EVENT\_APPLICATION\_DESELECT

EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE

EVENT\_EVENT\_BROWSING\_STATUS

EVENT\_EXTERNAL\_FILE\_UPDATE

* CRRN2: An EnvelopeHandler is considered available when no HANDLER\_NOT\_AVAILABLE ToolkitException is thrown when the corresponding getTheHandler() method is called or a method of the handler is called.

5.5.1.3.1.2 Parameter errors

No requirements.

5.5.1.3.1.3 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

EVENT\_FIRST\_COMMAND\_AFTER\_ATR

EVENT\_PROACTIVE\_HANDLER\_AVAILABLE

* CRRC2: The EnvelopeHandler shall not be available if the *getTheHandler()* method is not called, directly or indirectly, from the applet's *processToolkit*() method.

##### 5.5.1.3.2 Test area files

Test Source: Test\_Cre\_Mha\_Enhd.java.

Test Applet: Cre\_Mha\_Enhd\_1.java.

Cre\_Mha\_Enhd\_2.java.

Cre\_Mha\_Enhd\_3.java.

Cap File: Cre\_Mha\_Enhd.cap.

##### 5.5.1.3.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 17, 18, 19, 20, 22, 24, 26, 27, 28, 29 |
| CRRN2 | 1 to 24, 26, 27, 28, 29 |
| CRRC1 | 1, 2, 16, 23 |
| CRRC2 | 25 |

##### 5.5.1.3.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet1 and Applet2 registration and Envelope Handler availability with EVENT\_FIRST\_COMMAND\_AFTER\_ATR  1- Applet1 is registered to all events defined TS 102 241 [9] except to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE.  The registration is done using the methods initMenuEntry() for EVENT\_MENU\_SELECTION, requestPollInterval() for EVENT\_STATUS\_COMMAND, allocateTimer() for EVENT\_TIMER\_EXPIRATION, allocateServiceIdentifier() for EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION and setEventList() for the rest of the events.  Applet1 is registered on EVENT\_EXTERNAL\_FILE\_UPDATE on update of EFTARU of the UICC file system  Applet2 is registered to all events defined TS 102 241 [9] except to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE and EVENT\_CALL\_CONTROL\_BY\_NAA .  The registration is done 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 on EVENT\_EXTERNAL\_FILE\_UPDATE on update of EFTARU of the UICC file system  2- Select MF.  3- EnvelopeHandler.getTheHandler() method is called by Applet1  Applet1 is deregistered from EVENT\_FIRST\_COMMAND\_AFTER\_ATR.  4- EnvelopeHandler.getTheHandler() method is called by Applet2  Applet2 is deregistered to EVENT\_FIRST\_COMMAND\_AFTER\_ATR. | 1- No exception is thrown  2- Applet1 is triggered by EVENT\_FIRST\_COMMAND\_AFTER\_ATR  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 2 | Handler availability with EVENT\_PROFILE\_DOWNLOAD  1- Terminal Profile command is sent to the UICC without the facility of SET\_EVENT\_LIST, SETUP\_IDLE\_MODE\_TEXT, POLL\_INTERVAL and SETUP MENU  2- EnvelopeHandler.getTheHandler() method is called by Applet1  Applet1 is deregistered to EVENT\_PROFILE\_DOWNLOAD  3- EnvelopeHandler.getTheHandler() method is called by Applet2  Applet2 is deregistered to EVENT\_PROFILE\_DOWNLOAD | 1- Applet1 is triggered by EVENT\_PROFILE\_DOWNLOAD  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered by EVENT\_PROFILE\_DOWNLOAD  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 3 | Envelope Handler availability with EVENT\_MENU\_SELECTION\_HELP\_REQUEST  Perform UICC initialization with all the facilities supported  1- Envelope menu selection with help request is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes |  |
| 4 | Envelope Handler availability with EVENT\_MENU\_SELECTION  1- Envelope menu selection is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes |  |
| 5 | Envelope Handler availability with EVENT\_TIMER\_EXPIRATION  Timer id=1  1- Envelope Timer Expiration is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes |  |
| 6 | Envelope Handler availability with EVENT\_CALL\_CONTROL\_BY\_NAA  1- Envelope call control by NAA is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes |  |
| 7 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  1- Envelope event download mt call is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 8 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_CALL\_CONNECTED  1- Envelope event download call connected is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 9 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_CALL\_DISCONECTTED  1- Envelope event download call disconnected is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered.  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 10 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS  1- Envelope event download location status is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 11 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY  1- Envelope event download user activity is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 12 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_IDLE\_SCREEN\_AVAILABLE  1- Envelope event download idle screen available is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 13 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_CARD\_READER\_STATUS  1- Envelope event download card reader status is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 14 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_  SELECTION  1- Envelope event download language selection is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 15 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_BROWSER\_  TERMINATION  1- Envelope event download browser termination is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 16 | Envelope Handler availability with EVENT\_STATUS\_COMMAND  1- Status command is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 17 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE  1- Applet1 builds a proactive command OPEN CHANNEL.  proactiveHandler.send() method is called  2- Envelope event download data available is sent to the UICC with ChannelId=01.  3- EnvelopeHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is registered to EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE and EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS  2- Applet1 is triggered  3- No exception is thrown  Applet1 finalizes | 1- OPEN CHANNEL proactive command is fetched  TERMINAL RESPONSE is issued with Channel Id = 01 |
| 18 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS  1- Envelope event download channel status is sent to the UICC with ChannelId=01.  2- EnvelopeHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes |  |
| 19 | Envelope Handler availability with EVENT\_ UNRECOGNIZED\_ENVELOPE  1- An unrecognized Envelope is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  4- EnvelopeHandler.getTheHandler() method  is called by Applet2 | 1- Applet1 is triggered  2- No exception is thrown  3- Applet1 sends a DECLARE SERVICE (add) proactive command with its service identifier  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  Applet2 finalizes | 3- DECLARE SERVICE (add) proactive command is fetched  Successful TERMINAL RESPONSE is issued |
| 20 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION  1- Envelope event download local connection is sent to the UICC with the allocated service Id of Applet1  2- EnvelopeHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes |  |
| 21 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE  1- Envelope event download access technology change is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 22 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED  1- Envelope event display parameter changed is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered and registers to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered and registers to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  3- No exception is thrown  Applet2 finalizes |  |
| 23 | Envelope Handler availability with EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered by EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes  Applet2 is triggered by EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet2 finalizes |  |
| 24 | Envelope Handler availability with EVENT\_APPLICATION\_DESELECT  1- Select for activation ADF1  2- Select for termination ADF1  3- EnvelopeHandler.getTheHandler() method is called by Applet1.  Applet1 deregisters to EVENT\_APPLICATION\_DESELECT.  4- EnvelopeHandler.getTheHandler() method is called by Applet2.  Applet1 deregisters to EVENT\_APPLICATION\_DESELECT. | 2- Applet1 is triggered  3- No exception is thrown  Applet1 finalizes  Applet2 is triggered  4- No exception is thrown  Applet2 finalizes |  |
| 25 | Envelope Handler availability with EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE  1- Envelope event download network search mode change is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 26 | Envelope Handler availability with EVENT\_EVENT\_BROWSING\_STATUS  1- Envelope event download browsing status is sent to the UICC  2- EnvelopeHandler.getTheHandler() method is called by Applet1  3- EnvelopeHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No exception is thrown  Applet2 finalizes |  |
| 27 | The EnvelopeHandler is not available outside the processToolkit() method  1- Install Applet3. In its install method, Applet3 gets the EnvelopeHandler in a Try/Catch session  2- Select Applet3  3- Applet3 gets the EnvelopeHandler | 1- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  2- Applet3 is triggered by its process() method  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown |  |

#### 5.5.1.4 EnvelopeResponseHandler

Test Area Reference: Cre\_Mha\_Erhd

##### 5.5.1.4.1 Conformance requirement

5.5.1.4.1.1 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\_CALL\_CONTROL\_BY\_NAA

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.
* CRRN4: An EnvelopeResponseHandler is considered available when no HANDLER\_NOT\_AVAILABLE ToolkitException is thrown when the corresponding getTheHandler() method is called or a method of the handler is called.

5.5.1.4.1.2 Parameter errors

No requirements.

5.5.1.4.1.3 Context errors

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

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\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION

EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION

EVENT\_STATUS\_COMMAND

EVENT\_PROFILE\_DOWNLOAD

EVENT\_FIRST\_COMMAND\_AFTER\_ATR

EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE

EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS

EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE

EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED

EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION

EVENT\_APPLICATION\_DESELECT

EVENT\_PROACTIVE\_HANDLER\_AVAILABLE

EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE

EVENT\_EVENT\_BROWSING\_STATUS

EVENT\_EXTERNAL\_FILE\_UPDATE

* CRRC2: The EnvelopeResponseHandler shall not be available if the *getTheHandler()* method is not called, directly or indirectly, from the applet's *processToolkit*() method.

##### 5.5.1.4.2 Test area files

Test Source: Test\_Cre\_Mha\_Erhd.java.

Test Applet: Cre\_Mha\_Erhd\_1.java.

Cre\_Mha\_Erhd\_2.java.

Cre\_Mha\_Erhd\_3.java.

Cap File: Cre\_Mha\_Erhd.cap.

##### 5.5.1.4.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 18, 19, 20, 21 |
| CRRN2 | 18, 19 |
| CRRN3 | 18, 19 |
| CRRN4 | 1 to 29 |
| CRRC1 | 1 to 17 and 22 to 25, 26, 27, 28 |
| CRRC2 | 29 |

##### 5.5.1.4.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Toolkit Applet1 and Toolkit Applet2 registration and Envelope Response Handler availability with EVENT\_FIRST\_COMMAND\_AFTER\_ATR  1- Applet1 is registered to all events defined in TS 102 241 [9] except EVENT\_PROACTIVE\_HANDLER\_AVAILABLE.  Using the methods initMenuEntry() for EVENT\_MENU\_SELECTION, requestPollInterval() for EVENT\_STATUS\_COMMAND, allocateTimer() for EVENT\_TIMER\_EXPIRATION, allocateServiceIdentifier() for EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION and setEventList() for the rest of the events.  Applet1 is registered on EVENT\_EXTERNAL\_FILE\_UPDATE on update of EFTARU of the UICC file system  Applet2 is registered to  EVENT\_UNRECOGNIZED\_ENVELOPE.  2- Select MF.  3- EnvelopeResponseHandler.getTheHandler() method is called by Applet1  Applet1 is deregistered to EVENT\_FIRST\_COMMAND\_AFTER\_ATR. | 1- No exception is thrown  2- Applet1 is triggered by EVENT\_FIRST\_COMMAND\_AFTER\_ATR  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 2 | Handler availability with EVENT\_PROFILE\_DOWNLOAD  1- Terminal Profile command is sent to the UICC without the facility of SET\_EVENT\_LIST, SETUP\_IDLE\_MODE\_TEXT, SETUP\_MENU and POLL\_INTERVAL.  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1  Applet1 is deregistered to EVENT\_PROFILE\_DOWNLOAD | 1- Applet1 Is Triggered By EVENT\_PROFILE\_DOWNLOAD  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 3 | Envelope Response Handler availability with EVENT\_MENU\_SELECTION\_HELP\_REQUEST  Perform UICC initialization with all the facilities supported, except facility SET UP EVENT LIST  1- Envelope menu selection with help request is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 4 | Envelope Response Handler availability with EVENT\_MENU\_SELECTION  1- A envelope menu selection is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 5 | Envelope Response Handler availability with EVENT\_TIMER\_EXPIRATION  1- Envelope Timer Expiration is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown |  |
| 6 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  1- Envelope event download mt call is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 7 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_CALL\_CONNECTED  1- Envelope event download call connected is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 8 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_CALL\_DISCONNECTED  1- Envelope event download call disconnected is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 9 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS  1- Envelope event download location status is sent to the UICC  2- Applet1 obtains the Envelope Response Handler | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 10 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY  1- Envelope event download user activity is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 11 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_IDLE\_SCREEN\_AVAILABLE  1- Envelope event download idle screen available is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 12 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_CARD\_READER\_STATUS  1- Envelope event download card reader status is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 13 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_  SELECTION  1- Envelope event download language selection is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 14 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_BROWSER\_  TERMINATION  1- Envelope event download browser termination is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 15 | Envelope Response Handler availability with EVENT\_STATUS\_COMMAND  1- Status command is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 16 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE  1- Applet1 initializes a proactive command OPEN CHANNEL and calls the send() method.  2- Envelope event download data avalaible is sent to the UICC with channelId=01  3- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 2- Applet1 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes | 1- The OPEN CHANNEL command is fetched  TERMINAL RESPONSE IS SENT TO THE UICC with channel Id=01 |
| 17 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS  1- Envelope event download channel status is sent to the UICC with channelId=01  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 18 | Envelope Response Handler availability with EVENT\_CALL\_CONTROL\_BY\_NAA  1- Envelope call control by NAA is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1  3- Applet1 builds the envelope response and it calls the postAsBERTLV() method  4- Applet1 calls all methods of the Envelope Response Handler (including the inherited method)  5- Envelope call control by NAA is sent to the UICC  6- EnvelopeResponseHandler.getTheHandler() method is called by Applet1  7- Applet1 builds a proactive command and it calls the send() method  8- Applet1 calls all methods of the Envelope Response Handler (including the inherited method) | 1- Applet1 is triggered  2- No exception is thrown  4- ToolkitException HANDLER\_NOT\_AVAILABLE is thrown for each method  Applet1 finalizes  5- Applet1 is triggered  6- No Exception is thrown  8- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown for each method  Applet1 finalizes | 3- The envelope response is sent  7- The proactive command is fetched and the Terminal response is issued |
| 19 | Envelope Response Handler availability with EVENT\_UNRECOGNIZED\_ENVELOPE  1- An unrecognized Envelope is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1  3- Applet1 builds the envelope response and it calls the postAsBERTLV() or post method  4-Applet1 calls all methods of Envelope Response Handler (including the inherited method)  5- EnvelopeResponseHandler.getTheHandler() method is called  6- An unrecognized Envelope is sent to the UICC  7- EnvelopeResponseHandler.getTheHandler() method is called  8- Applet1 builds a proactive command and it calls the send() method  9- Applet1 calls all methods of the Envelope Response Handler (including the inherited method)  10- EnvelopeResponseHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered  2- No exception is thrown  4- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown for each method  Applet1 finalizes  Applet2 is triggered  5- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet2 finalizes  6- Applet1 is triggered  7- No exception is thrown  9- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown for each method  Applet1 finalizes  Applet2 is triggered  10- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown.  Applet2 finalizes | 3- The envelope response is sent  9- The proactive command is fetched and the Terminal response is issued |
| 20 | The envelope response is sent when a proactive session is ongoing  1- An unrecognized envelope is sent to the UICC.  2- Proactive command DISPLAY TEXT is built and it calls the send() method.  3- A call control by NAA envelope is sent to the UICC.  4- EnvelopeResponseHandler.getTheHandler() method is called by Applet1  5- Applet1 builds the envelope response and it calls the postAsBERTLV() | 1- Applet1 is triggered.  3- Applet1 is triggered  4- No exception is thrown | 2- 91 XX  5- The envelope response is checked  Expected SW = 91 XX  Fetch DISPLAY TEXT  Terminal Response DISPLAY TEXT |
| 21 | Envelope Response Handler availability with EVENT\_UNRECOGNIZED\_ENVELOPE in case of multi-triggering  1- An unrecognized Envelope is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1  3- EnvelopeResponseHandler.getTheHandler() method is called by Applet2 | 1- Applet1 is triggered  2- No exception is thrown  Applet1 finalizes  Applet2 is triggered  3- No Exception is thrown  Applet2 finalizes |  |
| 22 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE  1- Envelope event download access technology change is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 23 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED  1- Envelope event download display parameter changed is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered and registers to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 24 | Envelope Response Handler availability with EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered by EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  3- Applet1 sends a DECLARE SERVICE (add) proactive command with its service identifier  Applet1 finalizes | 3- DECLARE SERVICE (add) proactive command is fetched  Successful TERMINAL RESPONSE is issued |
| 25 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNACTION  1- Envelope event download local connection is sent to the UICC with the allocated service Id of Applet1  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 26 | Envelope Response Handler availability with EVENT\_APPLICATION\_DESELECT  1- Select for activation ADF1  2- Select for termination ADF1  3- EnvelopeResponseHandler.getTheHandler() method is called by Applet1  Applet1 deregisters to EVENT\_APPLICATION\_DESELECT | 2- Applet1 is triggered  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 27 | Envelope Response Handler availability with EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE  1- Envelope event download network search mode change is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 28 | Envelope Response Handler availability with EVENT\_EVENT\_BROWSING\_STATUS  1- Envelope event download browsing status is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler() method is called by Applet1 | 1- Applet1 is triggered  2- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  Applet1 finalizes |  |
| 29 | The EnvelopeResponseHandler is not available outside the processToolkit() method  1- Install Applet3. In its install method, Applet3 gets the EnvelopeResponseHandler in a Try/Catch session  2- Select Applet3  3- Applet3 gets the EnvelopeResponseHandler | 1- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown  2- Applet3 is triggered by its process() method  3- A ToolkitException HANDLER\_NOT\_AVAILABLE is thrown |  |

### 5.5.2 Handler Integrity

#### 5.5.2.1 ProactiveHandler

Test Area Reference: Cre\_Hin\_Pahd.

##### 5.5.2.1.1 Conformance requirement

5.5.2.1.1.1 Normal execution

* CRRN1: At the processToolkit invocation the TLV-List is cleared.
* CRRN2: After a call to *ProactiveHandler.send()* method the content of the handler shall not be modified by the CAT Runtime Environment.
* CRRN3: At the call of its init method the content is cleared and then initialized.

5.5.2.1.1.2 Parameter errors

No requirements.

5.5.2.1.1.3 Context errors

No requirements.

##### 5.5.2.1.2 Test area files

Test Source: Test\_Cre\_Hin\_Pahd.java.

Test Applet: Cre\_Hin\_Pahd\_1.java.

Cre\_Hin\_Pahd\_2.java.

Cap File: Cre\_Hin\_Pahd.cap.

##### 5.5.2.1.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2, 3, 4 |
| CRRN2 | 3 |
| CRRN3 | 5 |

##### 5.5.2.1.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | At the processToolkit invocation the TLV-List is cleared  Applet1 and Applet2 are registered to EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY.  1-An envelope containing an event download user activity is sent to the UICC  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 is checked that the content is the same than before the calling to send method using ProactiveHandler.copy and Util.arrayCompare methods | 2- The return value is 9, and its contents is the same than before the calling to send method  Applet1 finalizes | 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 |  |
| 5 | At the call of its init method the content is cleared and then initialized  1- proactiveHandler.init() method is called by Applet2  2- proactiveHandler.initCloseChannel() method is called by Applet2  3- proactiveHandler.initDisplayText() method is called by Applet2  4- proactiveHandler.initGetInkey() method is called by Applet2  5- proactiveHandler.initGetInput() method is called by Applet2  6- ProactiveHandler.initMoreTime() method is called by Applet2  7- proactiveHandler.init() method is called by Applet2 | 1- Check that the content of the proactive handler corresponds to the command defined by the parameters of the init() method  2- Check that the content of the proactive handler corresponds to the close channel command  3- Check that the content of the proactive handler corresponds to the display text command  4- Check that the content of the proactive handler corresponds to the get inkey command  5- Check that the content of the proactive handler corresponds to the get input command  6- Check that the content of the proactive handler corresponds to the More Time command  7- Check that the content of the proactive handler corresponds to the command defined by the parameters of the init() method |  |

#### 5.5.2.2 ProactiveResponseHandler

Test Area Reference: Cre\_Hin\_Prhd.

##### 5.5.2.2.1 Conformance requirement

5.5.2.2.1.1 Normal execution

* CRRN1: The *ProactiveResponseHandler* content shall be updated after each successful call to *ProactiveHandler.send()* method and shall remain unchanged until the next successful call to the *ProactiveHandler.send()* method.
* CRRN2: The ProactiveResponseHandler TLV list is filled with the comprehension TLV data objects of the last TERMINAL RESPONSE APDU command.
* CRRN3: The comprehension TLV data objects shall be provided in the order given in the TERMINAL RESPONSE command data.
* CRRN4: The ProactiveResponseHandler TLV list shall be empty before the first call to the ProactiveHandler.send() method.

5.5.2.2.1.2 Parameter errors

No requirements.

5.5.2.2.1.3 Context errors

No requirements.

##### 5.5.2.2.2 Test area files

Test Source: Test\_Cre\_Hin\_Prhd.java.

Test Applet: Cre\_Hin\_Prhd\_1.java.

Cap File: Cre\_Hin\_Prhd.cap.

##### 5.5.2.2.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |
| CRRN3 | 2 |
| CRRN4 | 1 |

##### 5.5.2.2.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration and ProactiveResponseHandler obtaining  1- Applet is registered to all events defined in ETSI TS 102 241[13] except to EVENT\_FIRST\_COMMAND\_AFTER\_ATR, EVENT\_PROACTIVE\_HANDLER\_AVAILABLE and EVENT\_APPLICATION\_DESELECT.  Using the methods initMenuEntry for EVENT\_MENU\_SELECTION and EVENT\_MENU\_SELECTION\_HELP\_REQUEST requestPollInterval() for EVENT\_STATUS\_COMMAND, allocateTimer() for EVENT\_TIMER\_EXPIRATION, allocateServiceIdentifier() for EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION and setEventList() for the rest of the events.  Applet is registered on EVENT\_EXTERNAL\_FILE\_UPDATE on update of EFTARU of the UICC file system  Terminal Profile command is sent to the UICC without the facilities of SETUP\_EVENT\_LIST, SETUP\_IDLE\_MODE\_TEXT, SETUP\_MENU and POLL\_INTERVAL.  2- For each event/triggering:  3- ProactiveResponseHandler.getTheHandler() is called  4- ProactiveResponseHandler.getLength() is called | 1- No exception is thrown  2- Applet is triggered.  For the first triggering, the applet registers to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  3- No exception is thrown  4- The return value is 0 |  |
| 2 | The ProactiveResponseHandler remains unchanged after send() method invocation until next send() method invocation  1- Applet builds a proactive command then the ProactiveHandler.send() method is called  2- proactiveResponseHandler.getTheHandler() method is called  3-ProactiveResponseHandler.getLength() method is called  4-ProactiveHandler.init() method is called  5-ProactiveHandler.send() method is called  6- proactiveResponseHandler.getTheHandler() method is called  7-ProactiveResponseHandler.getLength() method is called | 2- The ProactiveResponseHandler contains the terminal response. Comprehension TLV are provided in the order of the Terminal Response APDU  3- The return value is 12  4- No exception is thrown and the Proactive Response Handler remains unchanged  6- The ProactiveResponseHandler contains the terminal response of the second proactive command. Comprehension TLV are provided in the order of the Terminal Response APDU  7- The return value is 15 | 1- A proactive command is fetched  The terminal response is sent with length 12  5- A proactive command is fetched  The terminal response is sent with length 15 |

#### 5.5.2.3 EnvelopeHandler

Test Area Reference: Cre\_Hin\_Enhd.

##### 5.5.2.3.1 Conformance requirement

5.5.2.3.1.1 Normal execution

* CRRN1: When available, the *EnvelopeHandler* shall remain available and its content shall remain unchanged from the invocation to the termination of the *processToolkit()* method.
* CRRN2: The EnvelopeHandler TLV list is filled with the Comprehension TLV data objects of the ENVELOPE APDU command. The Comprehension TLV data objects shall be provided in the order given in the ENVELOPE command data.
* CRRN3: When an applet is triggered by the *EVENT\_EXTERNAL\_FILE\_UPDATE* event, the system Envelope Handler shall be available
* CRRN4: When an applet is triggered by the *EVENT\_EXTERNAL\_FILE\_UPDATE* event, the Envelope Handler shall contains the following COMPREHENSION TLVs (the order of the TLVs given in the system EnvelopeHandler is not specified):
* Device Identity with source set to terminal and destination set to UICC, as defined in ETSI TS 102 223 [6];
* File List, as defined in ETSI TS 102 223 [6]. The number of files shall be set to one. If a SFI referencing is used in the APDU Command, it shall be converted to its File Identifier;
* AID of the ADF, as defined in ETSI TS 102 223 [6], if the updated file belongs to an ADF. In this case, the path '3F007FFF' given in the File List indicates the ADF of the UICC application given through the AID. If the updated file belongs to the UICC shared file system, the AID TLV object is not present;
* File Update Information object.

|  |  |  |
| --- | --- | --- |
| Byte(s) | Description | Length |
| 1 | File Update Information tag | 1 |
| 2 | Length = 4 | 1 |
| 3 to 4 | Position | 2 |
| 5 to 6 | Number of bytes updated | 2 |

Position depends on the file type:

In case of transparent file, Position = Offset

In case of record file, Position = Absolute Record number

For the INCREASE APDU, the number of bytes updated is the record length.

* CRRN5: The value returned upon a *getTag()* method invocation is equal to the BER-TLV tag for intra-UICC communication, as defined in ETSI TS 101 220 [4].

5.5.2.3.1.2 Parameter errors

No requirements.

5.5.2.3.1.3 Context errors

No requirements.

##### 5.5.2.3.2 Test area files

Test Source: Test\_Cre\_Hin\_Enhd.java.

Test Applet: Cre\_Hin\_Enhd\_1.java.

Cap File: Cre\_Hin\_Enhd.cap.

##### 5.5.2.3.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1 to 21 |
| CRRN2 | 1 to 21 |
| CRRN3 | 22, 23 |
| CRRN4 | 22, 23 |
| CRRN5 | 1 |

##### 5.5.2.3.4 Test procedure

| Id | Description | API/CAT RE 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 TS 102 241 [9] except EVENT\_PROFILE\_DOWNLOAD, EVENT\_PROACTIVE\_HANDLER\_AVAILABLE and EVENT\_STATUS\_COMMAND.  Using the methods initMenuEntry() for EVENT\_MENU\_SELECTION and EVENT\_MENU\_SELECTION\_HELP\_REQUEST, allocateTimer()for EVENT\_TIMER\_EXPIRATION, allocateServiceIdentifier() for EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION and setEventList() for the rest of the events.  Perform UICC initialization with all the facilities supported  Applet is registered on EVENT\_EXTERNAL\_FILE\_UPDATE on update of EFTARU of the UICC file system and on update of EFLARU of the ADF file system  2- Envelope menu selection with help request is sent to the UICC  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- EnvelopeHandler.getTag() method is called  6- A proactive command DISPLAY TEXT is sent  7- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  8- It is 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.  9- Check that the TAG\_HELP\_REQUEST is the TLV selected  10- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare() | 1-No exception is thrown  2- Applet is triggered  3- No exception is thrown.  4- No exception is thrown  5- 0xD3 is returned  7- Applet is triggered  8- No exception is thrown and the handler contains the envelope call control by NAA  10- The contents of the envelope handler shall be the same as stored in buffer1 | 6- 91 xx.  A proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 2 | Envelope Handler integrity checks with EVENT\_MENU\_SELECTION  1- An envelope menu selection is sent to UICC  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 NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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  7- It is checked that the TAG\_ITEM\_IDENTIFIER is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 3 | Envelope Handler integrity checks with EVENT\_TIMER\_EXPIRATION  1- A timer expiration envelope is sent to UICC  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 NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_TIMER\_ID is the TLV selected  8- 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 NAA  7- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 4 | Envelope Handler integrity checks with EVENT\_CALL\_CONTROL\_BY\_NAA  1- A call control envelope is sent to UICC  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 NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_ADDRESS is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 5 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  1- A event download mt call envelope is sent to UICC  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\_EVENT\_LIST  4- A proactive command DISPLAY TEXT is sent  5- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_EVENT\_LIST is the TLV selected  8- 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 NAA  8- 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 UICC |
| 6 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_CALL\_CONNECTED  1- A event download call connected envelope is sent to UICC  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\_EVENT\_LIST  4- A proactive command DISPLAY TEXT is sent  5- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_EVENT\_LIST is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 7 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_CALL\_DISCONNECTED  1- A event download call disconnected envelope is sent to UICC  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\_EVENT\_LIST  4- A proactive command DISPLAY TEXT is sent  5- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_EVENT\_LIST is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 8 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS  1- A event download location status envelope is sent to UICC  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 NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_LOCATION\_STATUS is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4-91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 9 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY  1- A event download user activity envelope is sent to UICC  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 NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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-It is checked that the TAG\_DEVICE\_IDENTITIES is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 10 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_IDLE\_SCREEN\_AVAILABLE  1- A event download idle screen available envelope is sent to UICC  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 NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_DEVICE\_IDENTITIES is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 11 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_CARD\_READER\_STATUS  1- A event download card reader status envelope is sent to UICC  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 NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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  7- It is checked that the TAG\_CARD\_READER\_STATUS is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 12 | Envelope Handler integrity checks with UNRECOGNIZED\_ENVELOPE  1- A unrecognized envelope is sent to UICC  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 NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  The EnvelopeHandler.getValueLength() is called  6- It is 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 NAA  7- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 13 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION  1- A event download language selection envelope is sent to UICC  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\_EVENT\_LIST  4- A proactive command DISPLAY TEXT is sent  5- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_EVENT\_LIST is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4-91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 14 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION  1- A event download browser termination envelope is sent to UICC  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\_EVENT\_LIST  4- A proactive command DISPLAY TEXT is sent  5- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_EVENT\_LIST is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4-91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 15 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE  1- The applet builds a proactive command OPEN CHANNEL.  proactiveHandler.send() method is called  2- A event download data available envelope is sent to UICC  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\_CHANNEL\_STATUS  5- A proactive command DISPLAY TEXT is sent  6- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  7- It is 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  8- It is checked that the TAG\_CHANNEL\_STATUS is the TLV selected  9- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare() | 1- The applet is registered to EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE and EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS  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 control by NAA  9- The contents of the envelope handler shall be the same as stored in buffer1 | 1- OPEN CHANNEL proactive command is fetched  TERMINAL RESPONSE is issued with Channel Id = 01  5-91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 16 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS  1- A event download channel status envelope is sent to UICC  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\_CHANNEL\_STATUS  4- A proactive command DISPLAY TEXT is sent  5- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_CHANNEL\_STATUS is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4-91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 17 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE  1- A event download access technology change envelope is sent to UICC  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\_ACCESS\_TECHNOLOGY  4- A proactive command DISPLAY TEXT is sent  5- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_ACCESS\_TECHNOLOGY is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 18 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETER\_CHANGED  1- A event download display parameter changed envelope is sent to UICC  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\_DISPLAY\_PARAMETER  4- A proactive command DISPLAY TEXT is sent  5- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_DISPLAY\_PARAMETER is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1  9- Applet sends a DECLARE SERVICE (add) proactive command with its service identifier | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC  9- DECLARE SERVICE (add) proactive command is fetched  Successful TERMINAL RESPONSE is issued |
| 19 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION  1- A event download local connection envelope is sent to UICC with the allocated service Id of Applet  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\_SERVICE\_RECORD  4- A proactive command DISPLAY TEXT is sent  5- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_SERVICE\_RECORD is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 20 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE  1- A event download network search mode change envelope is sent to UICC  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\_NETWORK\_SEARCH\_MODE  4- A proactive command DISPLAY TEXT is sent  5- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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-It is checked that the TAG\_NETWORK\_SEARCH\_MODE is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 21 | Envelope Handler integrity checks with EVENT\_EVENT\_DOWNLOAD\_BROWSING STATUS  1- A event download browsing status envelope is sent to UICC  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\_BROWSING\_STATUS  4- A proactive command DISPLAY TEXT is sent  5- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_BROWSING\_STATUS is the TLV selected  8- 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 22 | Envelope Handler integrity checks with EVENT\_EXTERNAL\_FILE\_UPDATE under MF  1- EFTARU of the UICC file system is updated  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\_FILE\_LIST  4- A proactive command DISPLAY TEXT is sent  5- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_FILE\_LIST is the TLV selected  8- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()  9- It is checked that the EnvelopeHandler contains the comprehension TLVs: Device Identity, File List and File Update Information with the correct value.  The EnvelopeHandler does not contain the comprehension TLV AID | 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |
| 23 | Envelope Handler integrity checks with EVENT\_EXTERNAL\_FILE\_UPDATE under ADF  1- EFLARU of the ADF file system is updated  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\_FILE\_LIST  4- A proactive command DISPLAY TEXT is sent  5- Envelope call control by NAA is sent to UICC  EnvelopeHandler.getTheHandler() method is called  6- It is 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- It is checked that the TAG\_FILE\_LIST is the TLV selected  8- The contents of EnvelopeHandler are compared with buffer1 using Util.arrayCompare()  9- It is checked that the EnvelopeHandler contains the comprehension TLVs: Device Identity, File List, AID and File Update Information with the correct value. | 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 NAA  8- The contents of the envelope handler shall be the same as stored in buffer1 | 4- 91 XX  Proactive command Display Text is fetched  The terminal Response of DISPLAY TEXT is sent to the UICC |

#### 5.5.2.4 EnvelopeResponseHandler

Test Area Reference: Cre\_Hin\_Erhd.

##### 5.5.2.4.1 Conformance requirement

5.5.2.4.1.1 Normal execution

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

5.5.2.4.1.2 Parameter errors

No requirements.

5.5.2.4.1.3 Context errors

No requirements.

##### 5.5.2.4.2 Test area files

Test Source: Test\_Cre\_Hin\_Erhd.java.

Test Applet: Cre\_Hin\_Erhd\_1.java.

Cap\_File: Cre\_Hin\_Erhd.cap.

##### 5.5.2.4.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1 |

##### 5.5.2.4.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | At the processToolkit invocation the TLV-List is cleared  Applet1 is registered to EVENT\_UNRECOGNIZED\_ENVELOPE.  1-An unrecognized envelope is sent to the UICC  2- EnvelopeResponseHandler.getTheHandler()is called by the Applet1.  3- EnvelopeResponseHandler.getLength() method is called by Applet1 | 1- Applet1 is triggered  3- The return value shall be 0 |  |

### 5.5.3 Applet Triggering

#### 5.5.3.1 General behaviour

Test Area Reference: Cre\_Apt\_Genb.

##### 5.5.3.1.1 Conformance requirement

5.5.3.1.1.1 Normal execution

* CRRN1: When a first level application is the selected application and when a Toolkit Applet is triggered the *select*() method of the Toolkit Applet shall not be launched since the Toolkit Applet itself is not selected.
* CRRN2: The CAT Runtime Environment shall only trigger a Toolkit Applet if it is in the selectable state as defined in ETSI TS 102 226 [8].
* CRRN3: When the CAT Runtime Environment has to trigger several applets on the same event, the next applet is triggered on the return of the *processToolkit()* method of the previous Toolkit Applet.

5.5.3.1.1.2 Parameter errors

No requirements.

5.5.3.1.1.3 Context errors

No requirements.

##### 5.5.3.1.2 Test area files

Test Source: Test\_Cre\_Apt\_Genb.java.

Test Applet: Cre\_Apt\_Genb\_1.java.

Cap File: cre\_apt\_genb.cap.

##### 5.5.3.1.3 Test coverage

| CR Number | Test Case Number |
| --- | --- |
| CRRN1 | Covered by all other tests of this clause (5.5.3) |
| CRRN2 | 1, 2, 3, 4 |
| CRRN3 | Covered by all other tests of this clause (5.5.3) |

##### 5.5.3.1.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Triggering an applet in the installed state  When installed, the applet is registered to EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY  1- Install the applet without making it selectable  2- Select the Applet by AID  3- An envelope event download user activity is sent to the UICC | 2- Applet is not triggered | 1- SW = 90 00  2- SW ≠ 90 00 |
| 2 | Triggering an applet in the make selectable state  1- Install the applet to make it selectable  2- Select the Applet by AID3- An envelope event download user activity is sent to the UICC | 2- Applet is triggered | 1- SW = 90 00  2- SW = 90 00 |
| 3 | Triggering an applet in the lock state  1- Set the applet in the lock state  2- Select the Applet by AID 3- An envelope event download user activity is sent to the UICC | 2- Applet is not triggered | 1- SW = 90 00  2- SW ≠ 90 00 |
| 4 | Triggering an applet in the make selectable state  1- Set the applet in the make selectable state  2- Select the Applet by AID 3- An envelope event download user activity is sent to the UICC | 2- Applet is triggered | 1- SW = 90 00  2- SW = 90 00 |

#### 5.5.3.2 EVENT\_PROFILE\_DOWNLOAD

Test Area Reference: Cre\_Apt\_Epdw.

##### 5.5.3.2.1 Conformance requirement

5.5.3.2.1.1 Normal execution

* CRRN1: Upon reception of a TERMINAL PROFILE APDU command the CAT Runtime Environment shall store the terminal profile and trigger all the Toolkit Applet(s) registered to this event.
* CRRN2: The applet is not triggered by the EVENT\_PROFILE\_DOWNLOAD once it has deregistered from this event.
* CRRN3: The CAT Runtime Environment shall not reply busy to a Terminal Profile command.

5.5.3.2.1.2 Parameter errors

No requirements.

5.5.3.2.1.3 Context errors

No requirements.

##### 5.5.3.2.2 Test area files

Test Source: Test\_Cre\_Apt\_Epdw.java.

Test Applet: Cre\_Apt\_Epdw\_1.java.

Cre\_Apt\_Epdw\_2.java.

Cre\_Apt\_Epdw\_3.java.

Cap\_File: Cre\_apt\_epdw.cap.

##### 5.5.3.2.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1 |
| CRRN2 | 3 |
| CRRN3 | 2 |

##### 5.5.3.2.4 Test procedure

| Id | Description | API/CAT RE 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\_MENU\_SELECTION.  1-Terminal Profile command is sent to UICC | 1- Applet1 is triggered  Applet1 finalizes  2- Applet2 is triggered  Applet2 finalizes  3- Applet3 is not triggered |  |
| 2 | The CAT Runtime Environment shall not reply busy to a Terminal Profile command  1-Envelope menu selection is sent to the UICC  Applet3 builds a REFRESH proactive command in UICC initialization mode  2-ProactiveHandler.send() method is called by applet3  3-Terminal Profile command is sent to the UICC  Applet1 calls ToolkitRegistry.clearEvent( EVENT\_PROFILE\_DOWNLOAD)  4-Applet2 calls ToolkitRegistry.clearEvent( EVENT\_PROFILE\_DOWNLOAD)  6-Applet3 calls ToolkitRegistry.setEvent(EVENT\_PROFILE\_DOWNLOAD) | 1- Applet3 is triggered by the EVENT\_MENU\_SELECTION  Applet3 is suspended until the terminal response  3- Applet1 is triggered by EVENT\_PROFILE\_DOWNLOAD  Applet1 finalizes  Applet2 is triggered by EVENT\_PROFILE\_DOWNLOAD  Applet2 finalizes  Applet3 finalizes | 2- A proactive command is sent  5- The terminal response of the proactive command is sent |
| 3 | Deregistered applets are not triggered  Terminal Profile command is sent to the UICC | Applet3 is triggered  (Applet1 and Applet2 are not triggered) |  |

#### 5.5.3.3 EVENT\_MENU\_SELECTION

Test Area Reference: Cre\_Apt\_Emse.

##### 5.5.3.3.1 Conformance requirement

5.5.3.3.1.1 Normal execution

* CRRN1: Upon reception of an ENVELOPE (MENU SELECTION) APDU command, the CAT Runtime Environment shall only trigger the Toolkit Applet registered to the corresponding event with the associated menu identifier.

5.5.3.3.1.2 Parameter errors

No requirements.

5.5.3.3.1.3 Context errors

No requirements.

##### 5.5.3.3.2 Test area files

Test Source: Test\_Cre\_Apt\_Emse.java.

Test Applet: Cre\_Apt\_Emse\_1.java.

Cre\_Apt\_Emse\_2.java.

Cap File: Cre\_apt\_emse.cap.

##### 5.5.3.3.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1 |

##### 5.5.3.3.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_MENU\_SELECTION and triggering  ToolkitRegistry.initMenuEntry() method is called at the installation 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=true  IconQualifier=0  IconIdentifier=0  event= EVENT\_MENU\_SELECTION  1-ToolkitRegistry.isEventSet() is called at installation.  Perform UICC initialization with the facility SET UP MENU and without the facilities SET EVENT LIST and POLL INTERVAL features  2-Item Identifier = 1  Event Menu Selection envelope is sent to the UICC with the item identifier of a menu entry of Applet1  3-Item Identifier = 2  Event Menu Selection envelope is sent to the UICC with the item identifier of a menu entry of Applet2 | 1- The method shall return true  2- Applet1 is triggered and Applet2 is not triggered  Applet1 finalizes  3- Applet2 is triggered and Applet1 is not triggered |  |

#### 5.5.3.4 EVENT\_MENU\_SELECTION\_HELP\_REQUEST

Test Area Reference: Cre\_Apt\_Emsh.

##### 5.5.3.4.1 Conformance requirement

5.5.3.4.1.1 Normal execution

* CRRN1: If an ENVELOPE (MENU\_SELECTION\_HELP\_SUPPORTED) command is received for one entry supporting help, then CAT Runtime Environment shall trigger the corresponding applet.
* CCRN2: A toolkit applet shall be triggered by the EVENT\_MENU\_SELECTION\_HELP\_REQUEST event only if the Menu Id corresponding to the Envelope Menu Selection Help Request received by the CAT Runtime Environment was registered with the helpSupported value set to true.
* CCRN3: If at least one menuId of a Toolkit Applet registers to EVENT\_MENU\_SELECTION\_HELP\_REQUEST, the SET UP MENU proactive command sent by the CAT Runtime Environment shall indicate to the ME that help information is available unless all the menus entries that support help are disabled.

5.5.3.4.1.2 Parameter errors

No requirements.

5.5.3.4.1.3 Context errors

No requirements.

##### 5.5.3.4.2 Test area files

Test Source: Test\_Cre\_Apt\_Emsh.java.

Test Applet: Cre\_Apt\_Emsh\_1.java.

Cre\_Apt\_Emsh\_2.java.

Cre\_Apt\_Emsh\_3.java.

Cap File: Cre\_apt\_emsh.cap.

##### 5.5.3.4.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1 |
| CRRN2 | 1 |
| CRRN3 | 2 |

##### 5.5.3.4.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_MENU\_SELECTION\_HELP\_REQUEST and triggering  Applet1 and Applet2 are installed  ToolkitRegistry.InitMenuEntry() method is called at the installation of Applet1 and Applet2  For Applet1 (item id 1):  MenuEntry="Applet1A"  Offset=0  Length=menuEntry.length  HelpSupported=true  IconQualifier=0  IconIdentifier=0  For Applet1 (item id 2):  MenuEntry="Applet1B"  Offset=0  Length=menuEntry.length  HelpSupported=false  IconQualifier=0  IconIdentifier=0  event= EVENT\_MENU\_SELECTION\_HELP\_REQUEST  1- ToolkitRegistry.isEventSet() is called at the installation  For Applet2 (item id 3):  MenuEntry="Applet2A"  Offset=0  Length=menuEntry.length  HelpSupported=true  IconQualifier=0  IconIdentifier=0  For Applet2 (item id 4):  MenuEntry="Applet2B"  Offset=0  Length=menuEntry.length  HelpSupported=false  IconQualifier=0  IconIdentifier=0  event= EVENT\_MENU\_SELECTION\_HELP\_REQUEST  2- ToolkitRegistry.isEventSet() is called at installation  Perform UICC initialization with the facility SET UP MENU and without the facilities SET EVENT LIST and POLL INTERVAL  3-Item identifier = 1  Menu Selection Help Request envelope is sent to the UICC with item identifier 1 belonging to Applet1  4-Item identifier = 2  Menu Selection Help Request envelope is sent to the UICC with item identifier 2 belonging to Applet1  5-Item identifier = 3  Menu Selection Help Request envelope is sent to the UICC with item identifier 3 belonging to Applet2  6-Item identifier = 4  Menu Selection Help Request envelope is sent to the UICC with item identifier 4 belonging to Applet2 | 1- The command shall return true  2- The command shall return true  3- Applet1 is triggered and Applet2 is not triggered  4 Applet1 and Applet2 are not triggered  5- Applet2 is triggered and Applet1 is not triggered  6- Applet2 and Applet1 are not triggered |  |
| 2 | Applet deregistration to EVENT\_MENU\_SELECTION\_HELP\_REQUEST  Applet1 and Applet2 are deleted  Applet3 is installed  ToolkitRegistry.InitMenuEntry() method is called at the installation of Applet3  For Applet3 (item id 5):  MenuEntry="Applet3A"  Offset=0  Length=menuEntry.length  HelpSupported=true  IconQualifier=0  IconIdentifier=0  For Applet3 (item id 6):  MenuEntry="Applet3B"  Offset=0  Length=menuEntry.length  HelpSupported=true  IconQualifier=0  IconIdentifier=0  For Applet3 (item id 7):  MenuEntry="Applet3C"  Offset=0  Length=menuEntry.length  HelpSupported=false  IconQualifier=0  IconIdentifier=0  1- Perform UICC initialization with the facility SET UP MENU and without the facilities SET EVENT LIST and POLL INTERVAL  2- Menu Selection Help Request envelope is sent to the UICC with item identifier 5 belonging to applet3  3- ToolkitRegistry.disableMenuEntry() method for item id 5 is called by the Menu Selection Help Request Envelope.  4- Menu Selection Help Request envelope is sent to the UICC with item identifier 6 belonging to applet3  5- ToolkitRegistry.disableMenuEntry() method for item id 6 is called by the Menu Selection Help Request Envelope | 2- Applet3 is triggered by EVENT\_MENU\_SELECTION\_HELP\_REQUEST  4- Applet3 is triggered by EVENT\_MENU\_SELECTION\_HELP\_REQUEST | 1- The UICC shall issue a SET UP MENU proactive command with Menu Entry ID entry '05', '06' and '07', and Help supported set to true.  3- The UICC shall issue a SET UP MENU proactive command with Menu Entry ID entry '06' and '07', and Help supported set to true.  5- The UICC shall issue a SET UP MENU proactive command with Menu Entry ID entry '07', and Help supported set to false |

#### 5.5.3.5 EVENT\_CALL\_CONTROL\_BY\_NAA

Test Area Reference: Cre\_Apt\_Eccn.

##### 5.5.3.5.1 Conformance requirement

5.5.3.5.1.1 Normal execution

* CRRN1: The applet is triggered by the EVENT\_CALL\_CONTROL\_BY\_NAA once it has registered to this event and an Envelope Call Control by NAA is received.
* CRRN2: The applet is not triggered by the EVENT\_CALL\_CONTROL\_BY\_NAA once it has deregistered from this event.
* CRRN3: Regardless of the Toolkit Applet state the CAT Runtime Environment shall not allow more than one Toolkit Applet to be registered to this event at a time.

5.5.3.5.1.2 Parameter errors

No requirements.

5.5.3.5.1.3 Context errors

No requirements.

##### 5.5.3.5.2 Test area files

Test Source: Test\_Cre\_APT\_ECCN.java.

Test Applet: Cre\_Apt\_Eccn\_1.java.

Cre\_Apt\_Eccn\_2.java.

Cap File: Cre\_apt\_eccn.cap.

##### 5.5.3.5.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2, 3 |
| CRRN2 | 3 |
| CRRN3 | See API\_2\_TKR\_SEVTB |

##### 5.5.3.5.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applets registration to EVENT\_CALL\_CONTROL\_BY\_NAA and triggering  Applet1 is registered to EVENT\_CALL\_CONTROL\_BY\_NAA.  Applet2 is registered to EVENT\_MENU\_SELECTION  1- An Envelope Call control by NAA is sent to the UICC | 1- Applet1 is triggered |  |
| 2 | Applet deregistration and registration of the second applet to EVENT\_CALL\_CONTROL\_BY\_NAA.  1- An Envelope menu selection is sent to the UICC  Applet2 contructs a Display Text proactive command.  2- ProactiveHandler.send() method is called  3- An Envelope Call control by NAA envelope is sent to the UICC  ToolkitRegistry.clearEvent() is called for EVENT\_CALL\_CONTROL\_BY\_NAA  4- TERMINAL RESPONSE of Display Text is sent to the UICC  ToolkitRegistry.setEvent() method is called for EVENT\_CALL\_CONTROL\_BY\_NAA | 1- Applet2 is triggered by EVENT\_MENU\_SELECTION.  3- Applet1 is triggered  Applet1 finalizes.  4- Applet2 is resumed  Applet2 finalizes | 2- A proactive command Display Text is sent and  applet is suspended until the terminal response |
| 3 | Applet triggering  An Envelope Call control by NAA envelope is sent to the UICC | Applet2 is triggered.  (Applet1 is not triggered) |  |

#### 5.5.3.6 EVENT\_TIMER\_EXPIRATION

Test Area Reference: Cre\_Apt\_Etex.

##### 5.5.3.6.1 Conformance requirement

5.5.3.6.1.1 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.

5.5.3.6.1.2 Parameter errors

No requirements.

5.5.3.6.1.3 Context errors

No requirements.

##### 5.5.3.6.2 Test area files

Test Source: Test\_Cre\_Apt\_Etex.java.

Test Applet: Cre\_Apt\_Etex\_1.java.

Cap File: Cre\_apt\_etex.cap.

##### 5.5.3.6.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.6.4 Test procedure

| Id | Description | API/CAT RE 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\_MENU\_SELECTION.  event= EVENT\_TIMER\_EXPIRATION  1- ToolkitRegistry.isEventSet() method is called.  2- An Envelope TIMER\_EXPIRATION is sent to the UICC. | 1- The method returns true  2- Applet is triggered. |  |
| 2 | Applet deregistration  Timer id=1  ToolkitRegistry.ReleaseTimer() method is called  1- An Envelope timer expiration is sent to the UICC  An Envelope Menu selection is sent to the UICC  ToolkitRegistry.AllocateTimer() method is called  2- An Envelope TIMER\_EXPIRATION is sent to the UICC | 1- Applet is not triggered  2- Applet is triggered |  |

#### 5.5.3.7 EVENT\_EVENT\_DOWNLOAD\_MT\_CALL

Test Area Reference: Cre\_Apt\_Edmc.

##### 5.5.3.7.1 Conformance requirement

5.5.3.7.1.1 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.

5.5.3.7.1.2 Parameter errors

No requirements.

5.5.3.7.1.3 Context errors

No requirements.

##### 5.5.3.7.2 Test area files

Test Source: Test\_Cre\_Apt\_Edmc.java.

Test Applet: Cre\_Apt\_Edmc\_1.java.

Cap File: cre\_apt\_edmc.cap.

##### 5.5.3.7.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.7.4 Test procedure

| Id | Description | API/CAT RE 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\_MENU\_SELECTION  event= EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  1-ToolkitRegistry.isEventSet() method is called  2-An Envelope EVENT\_DOWNLOAD\_MT\_CALL is sent to the UICC | 1- The method returns true  2- Applet is triggered |  |
| 2 | Applet deregistration  event= EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  ToolkitRegistry.clearEvent()method is called  1-An Envelope EVENT\_DOWNLOAD\_MT\_CALL is sent to the UICC  An Envelope menu selestion is sent to the UICC  event= EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  ToolkitRegistry.setEvent() method is called  2-An Envelope EVENT\_DOWNLOAD\_MT\_CALL is sent to the UICC | 1- Applet is not triggered  2- Applet is triggered |  |

#### 5.5.3.8 EVENT\_EVENT\_DOWNLOAD\_CALL\_CONNECTED

Test Area Reference: Cre\_Apt\_Edcc.

##### 5.5.3.8.1 Conformance requirement

5.5.3.8.1.1 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.

5.5.3.8.1.2 Parameter errors

No requirements.

5.5.3.8.1.3 Context errors

No requirements.

##### 5.5.3.8.2 Test area files

Test Source: Test\_Cre\_Apt\_Edcc.java.

Test Applet: Cre\_Apt\_Edcc\_1.java.

Cap File: Cre\_apt\_edcc.cap.

##### 5.5.3.8.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.8.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_EVENT\_DOWNLOAD\_CALL\_CONNECTED and triggering  Applet is registered to the EVENT\_EVENT\_DOWNLOAD\_CALL\_CONNECTED and to EVENT\_MENU\_SELECTION  event= EVENT\_EVENT\_DOWNLOAD\_CALL\_CONNECTED  1-ToolkitRegistry.isEventSet() method is called  2-An Envelope EVENT\_DOWNLOAD\_CALL\_CONNECTED is sent to the UICC | 1- Method returns true  2- Applet is triggered |  |
| 2 | Applet deregistration  event=EVENT\_EVENT\_DOWNLOAD\_CALL\_CONNECTED  ToolkitRegistry.clearEvent()method is called  1-A call connected event dowload is sent to the UICC  An Envelope menu selection is sent to the UICC  Event= EVENT\_EVENT\_DOWNLOAD\_CALL\_CONNECTED  ToolkitRegistry.setEvent() method is called  2-An Envelope EVENT\_DOWNLOAD\_CALL\_CONNECTED is sent to the UICC | 1- Applet is not triggered  2- Applet is triggered |  |

#### 5.5.3.9 EVENT\_EVENT\_DOWNLOAD\_CALL\_DISCONNECTED

Test Area Reference: Cre\_Apt\_Edcd.

##### 5.5.3.9.1 Conformance requirement

5.5.3.9.1.1 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.

5.5.3.9.1.2 Parameter errors

No requirements.

5.5.3.9.1.3 Context errors

No requirements.

##### 5.5.3.9.2 Test area files

Test Source: Test\_Cre\_Apt\_Edcd.java.

Test Applet: Cre\_Apt\_Edcd\_1.java.

Cap File: Cre\_apt\_edcd.cap.

##### 5.5.3.9.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.9.4 Test procedure

| Id | Description | API/CAT RE 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\_MENU\_SELECTION  Event=EVENT\_EVENT\_DOWNLOAD\_CALL\_DISCONNECTED  1-ToolkitRegistry.isEventSet() method is called  2-An Envelope EVENT\_EVENT\_DOWNLOAD\_CALL\_DISCONNECTED is sent to the UICC | 1- Method returns true  2- Applet is triggered |  |
| 2 | Applet deregistration  Event= EVENT\_EVENT\_DOWNLOAD\_CALL\_DISCONNECTED  ToolkitRegistry.clearEvent()method is called  1-An Envelope EVENT\_DOWNLOAD\_CALL\_DISCONNECTED is sent to the UICC  An envelope menu selection is sent to the UICC  Event= EVENT\_EVENT\_DOWNLOAD\_CALL\_DISCONNECTED  ToolkitRegistry.setEvent() method is called  2-An Envelope EVENT\_EVENT\_DOWNLOAD\_CALL\_DISCONNECTED is sent to the UICC | 1- Applet is not triggered  2- Applet is triggered |  |

#### 5.5.3.10 EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS

Test Area Reference: Cre\_Apt\_Edls.

##### 5.5.3.10.1 Conformance requirement

5.5.3.10.1.1 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.

5.5.3.10.1.2 Parameter errors

No requirements.

5.5.3.10.1.3 Context errors

No requirements.

##### 5.5.3.10.2 Test area files

Test Source: Test\_Cre\_Apt\_Edls.java.

Test Applet: Cre\_Apt\_Edls\_1.java.

Cap File: Cre\_apt\_edls.cap.

##### 5.5.3.10.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.10.4 Test procedure

| Id | Description | API/CAT RE 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\_MENU\_SELECTION  Event=EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS  1-ToolkitRegistry.isEventSet() method is called  2-An Envelope EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS is sent to the UICC | 1- Method returns true  2- Applet is triggered |  |
| 2 | Applet deregistration  Event=EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS  ToolkitRegistry.clearEvent()method is called  1-An Envelope EVENT\_DOWNLOAD\_LOCATION\_STATUS is sent to the UICC  An Envelope menu selection is sent to the UICC  Event= EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS  ToolkitRegistry.setEvent() method is called  2-An Envelope EVENT\_DOWNLOAD\_LOCATION\_STATUS is sent to the UICC | 1- Applet is not triggered  2- Applet is triggered |  |

#### 5.5.3.11 EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY

Test Area Reference: Cre\_Apt\_Edua

##### 5.5.3.11.1 Conformance requirement

5.5.3.11.1.1 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.

5.5.3.11.1.2 Parameter errors

No requirements.

5.5.3.11.1.3 Context errors

No requirements.

##### 5.5.3.11.2 Test area files

Test Source: Test\_Cre\_Apt\_Edua.java.

Test Applet: Cre\_Apt\_Edua\_1.java.

Cap File: Cre\_apt\_edua.cap.

##### 5.5.3.11.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.11.4 Test procedure

| Id | Description | API/CAT RE 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\_MENU\_SELECTION  Event= EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY  1-ToolkitRegistry.isEventSet() method is called  2-An Envelope EVENT\_DOWNLOAD\_USER\_ACTIVITY is sent to the UICC | 1- Method returns true  2- Applet is triggered |  |
| 2 | Applet deregistration  Event= EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY  ToolkitRegistry.clearEvent()method is called  1-An Envelope EVENT\_DOWNLOAD\_USER\_ACTIVITY is sent to the UICC  An Envelope menu selection is sent to the UICC  Event= EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY  ToolkitRegistry.setEvent() method is called  2-An Envelope EVENT\_DOWNLOAD\_USER\_ACTIVITY is sent to the UICC | 1- Applet is not triggered  2- Applet is triggered |  |

#### 5.5.3.12 EVENT\_EVENT\_DOWNLOAD\_IDLE\_SCREEN\_AVAILABLE

Test Area Reference: Cre\_Apt\_Edis.

##### 5.5.3.12.1 Conformance requirement

5.5.3.12.1.1 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.

5.5.3.12.1.2 Parameter errors

No requirements.

5.5.3.12.1.3 Context errors

No requirements.

##### 5.5.3.12.2 Test area files

Test Source: Test\_Cre\_Apt\_Edis.java.

Test Applet: Cre\_Apt\_Edis\_1.java.

Cap File: Cre\_apt\_edis.cap.

##### 5.5.3.12.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.12.4 Test procedure

| Id | Description | API/CAT RE 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\_MENU\_SELECTION  Event=  EVENT\_EVENT\_DOWNLOAD\_IDLE\_SCREEN\_AVAILABLE  1-ToolkitRegistry.isEventSet() method is called  2- An Envelope EVENT\_DOWNLOAD\_IDLE\_SCREEN\_AVAILABLE is sent to the UICC | 1- Method returns true  2- Applet is triggered |  |
| 2 | Applet deregistration  Event=EVENT\_EVENT\_DOWNLOAD\_IDLE\_SCREEN\_AVAILABLE  ToolkitRegistry.clearEvent()method is called  1- An Envelope EVENT\_DOWNLOAD\_IDLE\_SCREEN\_AVAILABLE is sent to the UICC  An Envelope menu selection is sent to the UICC  Event= EVENT\_EVENT\_DOWNLOAD\_IDLE\_SCREEN\_AVAILABLE  ToolkitRegistry.setEvent() method is called  2-An Envelope EVENT\_DOWNLOAD\_IDLE\_SCREEN\_AVAILABLE is sent to the UICC | 1- Applet is not triggered  2- Applet is triggered |  |

#### 5.5.3.13 EVENT\_EVENT\_DOWNLOAD\_CARD\_READER\_STATUS

Test Area Reference: Cre\_Apt\_Edcr.

##### 5.5.3.13.1 Conformance requirement

5.5.3.13.1.1 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.

5.5.3.13.1.2 Parameter errors

No requirements.

5.5.3.13.1.3 Context errors

No requirements.

##### 5.5.3.13.2 Test area files

Test Source: Test\_Cre\_Apt\_Edcr.java.

Test Applet: Cre\_Apt\_Edcr\_1.java.

Cap File: Cre\_apt\_edcr.cap.

##### 5.5.3.13.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.13.4 Test procedure

| Id | Description | API/CAT RE 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\_MENU\_SELECTION  Event=EVENT\_EVENT\_DOWNLOAD\_CARD\_READER\_STATUS  1-ToolkitRegistry.isEventSet() method is called.  2- An Envelope EVENT\_DOWNLOAD\_CARD\_READER\_STATUS is sent to the UICC | 1- Method returns true  2- Applet is triggered |  |
| 2 | Applet deregistration  Event= EVENT\_EVENT\_DOWNLOAD\_CARD\_READER\_STATUS  ToolkitRegistry.clearEvent()method is called  1-An Envelope EVENT\_DOWNLOAD\_CARD\_READER\_STATUS is sent to the UICC  An Envelope menu selection is sent to the UICC  Event= EVENT\_EVENT\_DOWNLOAD\_CARD\_READER\_STATUS  ToolkitRegistry.setEvent() method is called  2-An Envelope EVENT\_DOWNLOAD\_CARD\_READER\_STATUS is sent to the UICC | 1- Applet is not triggered  2- Applet is triggered |  |

#### 5.5.3.14 EVENT\_UNRECOGNIZED\_ENVELOPE

Test Area Reference: Cre\_Apt\_Euev.

##### 5.5.3.14.1 Conformance requirement

5.5.3.14.1.1 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.

5.5.3.14.1.2 Parameter errors

No requirements.

5.5.3.14.1.3 Context errors

No requirements.

##### 5.5.3.14.2 Test area files

Test Source: Test\_Cre\_Apt\_Euen.java.

Test Applet: Cre\_Apt\_Euen\_1.java.

Cap File: Cre\_apt\_euen.cap.

##### 5.5.3.14.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.14.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_UNRECOGNIZED\_ENVELOPE and triggering  Applet is registered to the  EVENT\_UNRECOGNIZED\_ENVELOPE  Event= EVENT\_UNRECOGNIZED\_ENVELOPE  1-ToolkitRegistry.isEventSet() method is called  2-An Envelope UNRECOGNIZED\_ENVELOPE is sent to the UICC | 1- Method returns true  2- Applet is triggered |  |
| 2 | Applet deregistration  Event= EVENT\_UNRECOGNIZED\_ENVELOPE  ToolkitRegistry.clearEvent()method is called  1-An Envelope UNRECOGNIZED\_ENVELOPE is sent to the UICC  An Envelope menu selection is sent to the UICC  Event= EVENT\_UNRECOGNIZED\_ENVELOPE  ToolkitRegistry.setEvent() method is called  2-An Envelope UNRECOGNIZED\_ENVELOPE is sent to the UICC | 1- Applet is not triggered  2- Applet is triggered |  |

#### 5.5.3.15 EVENT\_STATUS\_COMMAND

Test Area Reference: Cre\_Apt\_Estc.

##### 5.5.3.15.1 Conformance requirement

5.5.3.15.1.1 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.

5.5.3.15.1.2 Parameter errors

No requirements.

5.5.3.15.1.3 Context errors

No requirements.

##### 5.5.3.15.2 Test area files

Test Source: Test\_Cre\_Apt\_Estc.java.

Test Applet: Cre\_Apt\_Estc\_1.java.

Cre\_Apt\_Estc\_2.java.

Cre\_Apt\_Estc\_3.java.

Cap File: Cre\_apt\_estc.cap.

##### 5.5.3.15.3 Test coverage

| CR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2, 3 |
| CRRN2 | 3 |

##### 5.5.3.15.4 Test procedure

| Id | Description | API/CAT RE 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\_MENU\_SELECTION.  1-A status command is sent to UICC | 1- Applet1 is triggered.  Applet1 finalizes  2- Applet2 is triggered.  Applet2 finalizes  3- Applet3 is not triggered |  |
| 2 | Applet deregistration and registration of the third applet to EVENT\_STATUS\_COMMAND.  1- An Envelope menu selection is sent to UICC  Applet3 builds a Display Text.  2- ProactiveHandler.send() is called  3- A status command is sent to UICC  4- requestPollInterval() method with POLL\_NO\_DURATION is called  5- requestPollInterval() method with POLL\_NO\_DURATION is called  6- TERMINAL RESPONSE of Display Text is sent to the UICC  7- requestPollInterval() method is called | 1- Applet3 is triggered.  3- Applet1 is triggered.  4- Applet1 is deregistered to EVENT\_STATUS\_COMMAND  Applet1 finalizes  Applet2 is triggered.  5- Applet2 is deregistered to EVENT\_STATUS\_COMMAND  Applet2 finalizes  6- Applet3 is resumed  7- Applet3 is registered to EVENT\_STATUS\_COMMAND  Applet3 finalizes | 2- A proactive command Display Text is sent and  applet is suspended until the terminal response |
| 3 | Applet3 triggering  Status command is sent to UICC | Applet3 is triggered.  (Applet1 and Applet2 are not triggered) |  |

#### 5.5.3.16 EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION

Test Area Reference: Cre\_Apt\_Edlg.

##### 5.5.3.16.1 Conformance requirement

5.5.3.16.1.1 Normal execution

* CRRN1: The applet is triggered by the EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION once it has registered to this event and an Envelope Event DownLoad Language Selection is received.
* CRRN2: The applet is not triggered by the EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION once it has deregistered from this event.

5.5.3.16.1.2 Parameter errors

No requirements.

5.5.3.16.1.3 Context errors

No requirements.

##### 5.5.3.16.2 Test area files

Test Source: Test\_Cre\_Apt\_Edlg.java.

Test Applet: Cre\_Apt\_Edlg\_1.java.

Cap File: Cre\_apt\_edlg.cap.

##### 5.5.3.16.3 Test coverage

| CR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.16.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION and triggering  Applet is registered to the EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION and to EVENT\_MENU\_SELECTION.  Event= EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION  1-ToolkitRegistry.isEventSet() method is called  2-An Envelope EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION is sent to the UICC | 1-Method returns true  2- Applet is triggered |  |
| 2 | Applet deregistration  Event= EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION  ToolkitRegistry.clearEvent()method is called  1-An Envelope EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION is sent to the UICC  An Envelope menu selection is sent to the UICC  Event= EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION  ToolkitRegistry.setEvent() method is called  2-An Envelope EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION is sent to the UICC | 1- Applet is not triggered  2- Applet is triggered |  |

#### 5.5.3.17 EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION

Test Area Reference: Cre\_Apt\_Edbt.

##### 5.5.3.17.1 Conformance requirement

5.5.3.17.1.1 Normal execution

* CRRN1: The applet is triggered by the EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION once it has registered to this event and an Envelope Event DownLoad Browser Termination is received.
* CRRN2: The applet is not triggered by the EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION once it has deregistered from this event.

5.5.3.17.1.2 Parameter errors

No requirements.

5.5.3.17.1.3 Context errors

No requirements.

##### 5.5.3.17.2 Test area files

Test Source: Test\_Cre\_Apt\_Edbt.java.

Test Applet: Cre\_Apt\_Edbt\_1.java.

Cap File: Cre\_apt\_edbt.cap.

##### 5.5.3.17.3 Test coverage

| CR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.17.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_EVENT\_DOWNLOAD\_ BROWSER\_TERMINATION and triggering  Applet is registered to the EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION and to EVENT\_MENU\_SELECTION  Event=  EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION  1-ToolkitRegistry.isEventSet() method is called  2-An Envelope EVENT\_DOWNLOAD\_BROWSER\_TERMINATION is sent to the UICC | 1-Method returns true  2- Applet is triggered |  |
| 2 | Applet deregistration  Event= EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION  ToolkitRegistry.clearEvent()method is called  1-An Envelope EVENT\_DOWNLOAD\_BROWSER\_TERMINATION is sent to the UICC  An Envelope menu selection is sent to the UICC  Event= EVENT\_EVENT\_DOWNLOAD\_BROWSER\_TERMINATION  ToolkitRegistry.setEvent() method is called  2-An Envelope EVENT\_DOWNLOAD\_BROWSER\_TERMINATION is sent to the UICC | 1- Applet is not triggered  2- Applet is triggered |  |

#### 5.5.3.18 EVENT\_FIRST\_COMMAND\_AFTER\_ATR

Test Area Reference: Cre\_Apt\_Efca.

##### 5.5.3.18.1 Conformance requirement

5.5.3.18.1.1 Normal execution

* CRRN1: The applet is triggered by the EVENT\_FIRST\_COMMAND\_AFTER\_ATR once it has registered to this event and upon the reception of the first APDU after the ATR, before the Status Word of the processed command has been sent back by the UICC.
* CRRN2: The applet is not triggered by the EVENT\_FIRST\_COMMAND\_AFTER\_ATR once it has deregistered from this event.
* CRRN3: If the first APDU received is a toolkit applet triggering APDU (e.g. TERMINAL PROFILE), the toolkit applets registered to the EVENT\_FIRST\_COMMAND\_AFTER\_ATR event shall be triggered first.

5.5.3.18.1.2 Parameter errors

No requirements.

5.5.3.18.1.3 Context errors

No requirements.

##### 5.5.3.18.2 Test area files

Test Source: Test\_Cre\_Apt\_Efca.java.

Test Applet: Cre\_Apt\_Efca\_1.java.

Cre\_Apt\_Efca\_2.java.

Cre\_Apt\_Efca\_3.java.

Cre\_Apt\_Efca\_4.java.

Cre\_Apt\_Efca\_5.java.

Cap File: Cre\_apt\_efca.cap.

##### 5.5.3.18.3 Test coverage

| CR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2, 3, 4 |
| CRRN2 | 3 |
| CRRN3 | 1, 4 |

##### 5.5.3.18.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applets registration to EVENT\_FIRST\_COMMAND\_AFTER\_ATR and triggering  Applet1 is registered to the EVENT\_FIRST\_COMMAND\_AFTER\_ATR  Applet2 is registered to the EVENT\_PROFILE\_DOWNLOAD  Applet3 is registered to EVENT\_MENU\_SELECTION  1- Terminal Profile command is sent to the UICC  2- Applet1 deregisters from EVENT\_FIRST\_COMMAND\_AFTER\_ATR  3- Applet2 deregisters from EVENT\_PROFILE\_DOWNLOAD  4- An Envelope menu selection is sent to the UICC  5- Applet3 calls setEvent() on event EVENT\_FIRST\_COMMAND\_AFTER\_ATR | 1- Applet1 is triggered by EVENT\_FIRST\_COMMAND\_AFTER\_ATR  Applet1 finalizes  Applet2 is triggered by EVENT\_PROFILE\_DOWNLOAD  Applet2 finalizes  Applet3 is not triggered  4- Applet3 is triggered. |  |
| 2 | Deregistered applets are not triggered  1-Reset then Terminal Profile command is sent to the UICC  2-Applet3 calls setEvent() on EVENT\_PROFILE\_DOWNLOAD | 1-Applet3 is triggered  Applet1 and Applet2 are not triggered  2-Applet3 finalizes |  |
| 3 | Install a 4th applet registered to EVENT\_FIRST\_COMMAND\_AFTER\_ATR and EVENT\_PROFILE\_DOWNLOAD  Applet4 is installed, with the same priority level as Applet3  1-Reset then Terminal Profile command is sent to the UICC  2- Delete all applets | 1- Applet4 is triggered by EVENT\_FIRST\_COMMAND\_AFTER\_ATR  Applet4 finalizes  Applet3 is triggered by EVENT\_FIRST\_COMMAND\_AFTER\_ATR  Applet3 finalizes  Applet4 is triggered by EVENT\_PROFILE DOWNLOAD  Applet4 finalizes  Applet3 is triggered by EVENT\_PROFILE\_DOWNLOAD  Applet3 finalizes |  |
| 4 | Check that the applet is triggered before the first status word is sent  1- Install Applet5  Applet5 is registered with two entries in the menu entries list  Applet5 is also registered to EVENT\_FIRST\_COMMAND\_AFTER\_ATR  2- Reset then Terminal Profile command is sent to the UICC  3- Applet5 disables a menu entry | 2- Applet5 is triggered | 3-The SETUP MENU proactive command is fetched.  There is only one item for Applet5 |

#### 5.5.3.19 EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE

Test Area Reference: Cre\_Apt\_Edda.

##### 5.5.3.19.1 Conformance requirement

5.5.3.19.1.1 Normal execution

* CRRN1: For EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE, the framework shall only trigger the applet registered to this event with the appropriate channel identifier.
* CRRN2: The registration to the EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE is effective once the toolkit applet has issued a successful OPEN CHANNEL proactive command, and valid till the first successful CLOSE CHANNEL or the end of card session.
* CRRN3: When a Toolkit Applet has sent an OPEN CHANNEL proactive command and received a successful TERMINAL RESPONSE with General Result ="0x0X", the framework shall register the received channel identifier for the calling Toolkit Applet.
* CRRN4: When a Toolkit Applet has sent a CLOSE CHANNEL proactive command and received a successful TERMINAL RESPONSE with General Result ="0x0X", the framework shall release the channel identifier contained in the command.

5.5.3.19.1.2 Parameter errors

No requirements.

5.5.3.19.1.3 Context errors

No requirements.

##### 5.5.3.19.2 Test area files

Test Applet: Test\_Cre\_Apt\_Edda.java.

Test Applet: Cre\_Apt\_Edda\_1.java.

Cap File: Cre\_apt\_edda.cap.

##### 5.5.3.19.3 Test coverage

| CR Number | Test Case Number |
| --- | --- |
| CRRN1 | 2 |
| CRRN2 | 1, 4, 5 |
| CRRN3 | 1 |
| CRRN4 | 3 |

##### 5.5.3.19.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE  Applet1 is registered to Event Menu selection.  1- An Envelope menu selection is sent to the UICC  2- Applet calls setEvent() with the event EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE  3- An envelope Event Download Data Available is sent to the UICC  Channel Status = 81 00  4- An Envelope menu selection is sent to the UICC  5- Applet1 builds a proactive command OPEN CHANNEL calling ProactiveHandler.init() method  6- send() method is called to register to this event  8- An envelope Event Download Data Available is sent to the UICC with Channel Status = 01 00  9- An Envelope menu selection is sent to the UICC  10- Applet1 builds a proactive command OPEN CHANNEL calling ProactiveHandler.init() method  11- send() method is called to register to this event | 1- Applet1 is triggered by the envelope menu selection  2- Applet1 finalizes  3- Applet1 is not triggered  4- Applet1 is triggered by the envelope menu selection  7- Applet1 finalizes  8- Applet1 is not triggered  9- Applet1 is triggered by the envelope menu selection  12- Applet1 finalizes | 6- OPEN CHANNEL proactive command is fetched  Unsuccessful TERMINAL RESPONSE of OPEN CHANNEL is sent to the UICC with General Result = 0x10  11- OPEN CHANNEL proactive command is fetched.  Successful TERMINAL RESPONSE of OPEN CHANNEL is sent to the UICC with Channel Id = 01 with General Result = 0x00 |
| 2 | Applet triggering to EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE  1- An envelope Event Download Data Available is sent to the UICC  Channel Status = 81 00 | 1- Applet1 is triggered |  |
| 3 | Applet deregistration to EVENT\_EVENT\_ DOWNLOAD\_DATA\_ AVAILABLE  1- An Envelope menu selection is sent to the UICC  2- Applet1 initializes and sends an OPEN CHANNEL proactive command  3- Applet1 builds a CLOSE CHANNEL Proactive Command calling ProactiveHandler.initCloseChannel() and ProactiveHandler.send() methods  4- An envelope Event Download Data Available is sent to the UICC  Channel Status = 82 00  5- Applet1 builds a CLOSE CHANNEL Proactive Command calling ProactiveHandler.initCloseChannel() and ProactiveHandler.send() methods. | 1- Applet1 is triggered  4- Applet1 is triggered  6- Applet1 finalize | 2- OPEN CHANNEL proactive command is fetched  Successful terminal response is sent, with channelId=02 with General Result = 0x01  3- CLOSE CHANNEL proactive command is fetched  Unsuccessful TERMINAL RESPONSE of CLOSE CHANNEL is sent to the UICC with General Result = 0X20  5- CLOSE CHANNEL proactive command is fetched  Successful TERMINAL RESPONSE of CLOSE CHANNEL is sent to the UICC with Channel Id = 02 with General Result = 0X02 |
| 4 | Applet triggering to EVENT\_EVENT\_DOWNLOAD\_DATA\_AVAILABLE  1- An envelope Event Download Data Available is sent to the UICC  Channel Status = 82 00 | 1- Applet1 is not triggered |  |
| 5 | Applet1 not triggered after a reset  1- Applet1 is triggered by an envelope menu selection  2- Applet1 builds a proactive command OPEN CHANNEL calling ProactiveHandler.init() method  3- send() method is called to register to this event  4- isEventSet() method is called  5- Reset the card  6- An envelope Event Download Data Available is sent to the UICC  Channel Status = 82 00 | 4- returns true  6- Applet1 is not triggered | 3- OPEN CHANNEL proactive command is fetched  Successful TERMINAL RESPONSE of OPEN CHANNEL is sent to the UICC with Channel Id = 02 with General Result = 0X00 |

#### 5.5.3.20 EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS

Test Area Reference: Cre\_Apt\_Edcs.

##### 5.5.3.20.1 Conformance requirement

5.5.3.20.1.1 Normal execution

* CRRN1: For EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS, the framework shall only trigger the applet registered to this event with the appropriate channel identifier.
* CRRN2: The registration to the EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS is effective once the toolkit applet has issued a successful OPEN CHANNEL proactive command, and valid till the first successful CLOSE CHANNEL or the end of the card session.
* CRRN3: When a Toolkit Applet has sent an OPEN CHANNEL proactive command and received a successful TERMINAL RESPONSE, the framework shall register the received channel identifier for the calling Toolkit Applet.
* CRRN4: When a Toolkit Applet has sent a CLOSE CHANNEL proactive command and received a successful TERMINAL RESPONSE, the framework shall release the channel identifier contained in the command. A successful TERMINAL RESPONSE means that the result of the proactive command execution belongs to command performed category (i.e. General Result ='0x').

5.5.3.20.1.2 Parameter errors

No requirements.

5.5.3.20.1.3 Context errors

No requirements.

##### 5.5.3.20.2 Test area files

Test Source: Test\_Cre\_Apt\_Edcs.java.

Test Applet: Cre\_Apt\_Edcs\_1.java.

Cap File: Cre\_apt\_edcs.cap.

##### 5.5.3.20.3 Test coverage

| CR Number | Test Case Number |
| --- | --- |
| CRRN1 | 2 |
| CRRN2 | 1, 4, 5 |
| CRRN3 | 1 |
| CRRN4 | 3 |

##### 5.5.3.20.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS  Applet1 is registered to Event Menu Selection  1- An Envelope menu selection is sent to the UICC  2-The applet calls setEvent() with EVENT\_EVENT\_DOWNLOAD\_CHANNEL\_STATUS  3- An envelope Event Download Channel Status is sent to the UICC  Channel Status = 81 00  4- An Envelope menu selection is sent to the UICC  5- Applet1 builds a proactive command OPEN CHANNEL calling ProactiveHandler.init() method  6- send() method is called to register to this event  8- An envelope Event Download Data Available is sent to the UICC with Channel Status = 01 00  9- An Envelope menu selection is sent to the UICC  10- Applet1 builds a proactive command OPEN CHANNEL calling ProactiveHandler.init() method  11- send() method is called to register to this event a second time | 1- Applet1 is triggered by the envelope menu selection  Applet1 finalizes  3- Applet1 is not triggered  4- Applet1 is triggered by the envelope menu selection  7- Applet finalizes  8- Applet1 is not triggered  9- Applet1 is triggered by the envelope menu selection  12- Applet1 finalizes | 6- OPEN CHANNEL proactive command is fetched  Unsuccessful TERMINAL RESPONSE of OPEN CHANNEL is sent to the UICC with General Result = 0x10  11- OPEN CHANNEL proactive command is fetched  Successful TERMINAL RESPONSE of OPEN CHANNEL is sent to the UICC with Channel Id = 01 with General Result = 0x00 |
| 2 | Applet triggering to EVENT\_EVENT\_DOWNLOAD\_CHANNEL STATUS  1- An envelope Event Download Channel Status is sent to the UICC  Channel Status = 81 00 | 1- Applet1 is triggered |  |
| 3 | Applet deregistration to EVENT\_EVENT\_ DOWNLOAD\_CHANNEL STATUS  1- An Envelope menu selection is sent to the UICC  2-Applet1 initializes and sends an OPEN CHANNEL proactive command  3- Applet1 builds a CLOSE CHANNEL Proactive Command calling ProactiveHandler.initCloseChannel() and ProactiveHandler.send() methods  4-An envelope Event Download Channel Status is sent to the UICC  Channel Status = 82 00  5- Applet1 builds a Close Channel Proactive Command calling ProactiveHandler.initCloseChannel() and ProactiveHandler.send() methods | 1- Applet1 is triggered  4- The applet is triggered  6- Applet1 finalizes | 2- OPEN CHANNEL proactive command is fetched  Successful terminal response is sent, with channel Id=02 with General Result = 0x01  3-CLOSE CHANNEL proactive command is fetched  Unsuccessful TERMINAL RESPONSE of CLOSE CHANNEL is sent to the UICC with General Result = 0X20  5- CLOSE CHANNEL proactive command is fetched  Successful TERMINAL RESPONSE of CLOSE CHANNEL is sent to the UICC with Channel Id = 02  with General Result = 0X02 |
| 4 | Applet triggering to EVENT\_EVENT\_DOWNLOAD\_CHANNEL STATUS  1- An envelope Event Download Channel Status is sent to the UICC  Channel Status = 82 00 | 1- Applet1 is not triggered |  |
| 5 | Applet1 not triggered after a reset  1- Applet1 is triggered by an envelope menu selection  2- Applet1 builds a proactive command OPEN CHANNEL calling ProactiveHandler.init() method  3- send() method is called to register to this event  4- isEventSet() method is called  5- Reset the card  6- An envelope Event Download Channel Status is sent to the UICC  Channel Status = 82 00 | 4- returns true  6- Applet1 is not triggered | 3- OPEN CHANNEL proactive command is fetched  Successful TERMINAL RESPONSE of OPEN CHANNEL is sent to the UICC with Channel Id = 02 with General Result = 0X00 |

#### 5.5.3.21 EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE

Test Area Reference: Cre\_Apt\_Edat.

##### 5.5.3.21.1 Conformance requirement

5.5.3.21.1.1 Normal execution

* CRRN1: The applet is triggered by the EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE once it has registered to this event and an Envelope Event DownLoad Access Technology Change is received.
* CRRN2: The applet is not triggered by the EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE once it has deregistered from this event.

5.5.3.21.1.2 Parameter errors

No requirements.

5.5.3.21.1.3 Context errors

No requirements.

##### 5.5.3.21.2 Test area files

Test Source: Test\_Cre\_Apt\_Edat.java.

Test Applet: Cre\_Apt\_Edat\_1.java.

Cap File: Cre\_apt\_edat.cap.

##### 5.5.3.21.3 Test coverage

| CR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.21.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_EVENT\_DOWNLOAD ACCESS\_TECHNOLOGY\_CHANGE and triggering  Applet is registered to the EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE and to EVENT\_MENU\_SELECTION  Event=  EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE  1- ToolkitRegistry.isEventSet() method is called  2- An Envelope EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE is sent to the UICC | 1- Method returns true  2- Applet is triggered |  |
| 2 | Applet deregistration  Event= EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE  ToolkitRegistry.clearEvent()method is called  1- An Envelope EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE is sent to the UICC  2- An Envelope menu selection is sent to the UICC  Event= EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE  ToolkitRegistry.setEvent() method is called  3- An Envelope EVENT\_EVENT\_DOWNLOAD\_ACCESS\_TECHNOLOGY\_CHANGE is sent to the UICC | 1- Applet is not triggered  2- Applet is triggered  3- Applet is triggered |  |

#### 5.5.3.22 EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED

Test Area Reference: Cre\_Apt\_Eddp.

##### 5.5.3.22.1 Conformance requirement

5.5.3.22.1.1 Normal execution

* CRRN1: The applet is triggered by the EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED once it has registered to this event and an Envelope Event DownLoad Display Parameters Changed is received.
* CRRN2: The applet is not triggered by the EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED once it has deregistered from this event.

5.5.3.22.1.2 Parameter errors

No requirements.

5.5.3.22.1.3 Context errors

No requirements.

##### 5.5.3.22.2 Test area files

Test Source: Test\_Cre\_Apt\_Eddp.java.

Test Applet: Cre\_Apt\_Eddp\_1.java.

Cap File: Cre\_apt\_eddp.cap.

##### 5.5.3.22.3 Test coverage

| CR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.22.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_EVENT\_DOWNLOAD DISPLAY\_PARAMETERS\_CHANGED and triggering  Applet is registered to the EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED and to EVENT\_MENU\_SELECTION  Event=  EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED  1- ToolkitRegistry.isEventSet() method is called  2- An Envelope EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED is sent to the UICC | 1- Method returns true  2- Applet is triggered |  |
| 2 | Applet deregistration  Event= EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED  ToolkitRegistry.clearEvent()method is called  1- An Envelope EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED is sent to the UICC  2- An Envelope menu selection is sent to the UICC  Event= EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED  ToolkitRegistry.setEvent() method is called  3- An Envelope EVENT\_EVENT\_DOWNLOAD\_DISPLAY\_PARAMETERS\_CHANGED is sent to the UICC | 1- Applet is not triggered  2- Applet is triggered  3- Applet is triggered |  |

#### 5.5.3.23 EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION

Test Area Reference: Cre\_Apt\_Edlc.

##### 5.5.3.23.1 Conformance requirement

5.5.3.23.1.1 Normal execution

* CRRN1: For EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION, the framework shall only trigger the applet registered to this event with the associated service identifier.
* CRRN2: The registration to the EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION is effective once the toolkit applet has issued a successful DECLARE SERVICE (add) proactive command, and valid till the first successful DECLARE SERVICE (delete) with the corresponding service identifier or the end of the card session.

5.5.3.23.1.2 Parameter errors

No requirements.

5.5.3.23.1.3 Context errors

No requirements.

##### 5.5.3.23.2 Test area files

Test Source: Test\_Cre\_Apt\_Edlc.java.

Test Applet: Cre\_Apt\_Edlc\_1.java.

Cap File: Cre\_apt\_edlc.cap.

##### 5.5.3.23.3 Test coverage

| CR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 3, 4, 5 |

##### 5.5.3.23.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_EVENT\_DOWNLOAD LOCAL\_CONNECTION  Applet is registered to the EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION using the allocateServiceIdentifier() method and to EVENT\_MENU\_SELECTION using the initMenuEntry() method.  1- An envelope menu selection is sent to the UICC  Event=  EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION  2- ToolkitRegistry.isEventSet() method is called  3- An Envelope EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION is sent to the UICC with the service identifier of Applet  4- An envelope menu selection is sent to the UICC  5- Applet builds and sends a DECLARE SERVICE (add) proactive command with its service ID to register to this event.  6- An Envelope EVENT\_EVENT\_DOWNLOAD\_LOCAL\_CONNECTION is sent to the UICC with the service identifier of Applet  7- An envelope menu selection is sent to the UICC  8- Applet builds and sends a DECLARE SERVICE (add) proactive command with its service ID to register to this event. | 1- Applet is triggered  2- Method returns true  3- Applet is not triggered  4- Applet is triggered by the envelope menu selection  5- Applet finalizes  6- Applet is not triggered  7- Applet is triggered by the envelope menu selection  9- Applet finalizes | 5- DECLARE SERVICE (add) proactive command is fetched  Unsuccessful TERMINAL RESPONSE is sent to the UICC with General Result = 0x20  8- DECLARE SERVICE proactive command is fetched  Successful TERMINAL RESPONSE is sent to the UICC with General Result = 0x00 |
| 2 | Applet triggering to EVENT\_EVENT\_DOWNLOAD LOCAL\_CONNECTION  1- An envelope Event Download local connection is sent to the UICC  with the service identifier of the Applet | 1- Applet is triggered |  |
| 3 | Applet deregistration to EVENT\_EVENT\_DOWNLOAD LOCAL\_CONNECTION with proactive command  1- An Envelope menu selection is sent to the UICC  2- Applet initializes and sends a DECLARE SERVICE (delete) proactive command with the service identifier of Applet  3- An envelope Event Download local connection is sent to the UICC  with the service identifier of the Applet  4-Applet initializes and sends a DECLARE SERVICE (delete) proactive command with the service identifier of Applet | 1- Applet1 is triggered  Applet finalizes  3- Applet is triggered  5- Applet finalizes | 2- DECLARE SERVICE (delete) proactive command is fetched  Unsuccessful TERMINAL RESPONSE is sent to the UICC with General Result = 0X20  4- DECLARE SERVICE (delete) proactive command is fetched  Successful TERMINAL RESPONSE is sent to the UICC with General Result = 0X00 |
| 4 | Applet triggering to EVENT\_EVENT\_DOWNLOAD LOCAL\_CONNECTION  1- An envelope Event Download local connection is sent to the UICC  with the service identifier of the Applet | 1- Applet is not triggered |  |
| 5 | Applet1 not triggered after a reset  1- Applet1 is triggered by an envelope menu selection  2- Applet builds and sends a DECLARE SERVICE (add) proactive command with its service ID to register to this event  3- An envelope Event Download local connection is sent to the UICC  with the service identifier of the Applet  4- Reset the card  5- An envelope Event Download local connection is sent to the UICC  with the service identifier of the Applet | 3- Applet is triggered  5- Applet is not triggered | 2- DECLARE SERVICE (add) proactive command is fetched  Successful TERMINAL RESPONSE is sent to the UICC with General Result = 0x00 |

#### 5.5.3.24 EVENT\_APPLICATION\_DESELECT

Test Area Reference: Cre\_Apt\_Eade.

##### 5.5.3.24.1 Conformance requirement

5.5.3.24.1.1 Normal execution

* CRRN1: The applet is triggered by the EVENT\_APPLICATION\_DESELECT once it has registered to this event and once an application session is terminated (as described in ETSI TS 102 221 [5]).
* CRRN2: The applet is not triggered by the EVENT\_APPLICATION\_DESELECT once it has deregistered from this event.
* CRRN3: The AID of the deselected application is available to the Toolkit Applet in the *EnvelopeHandler*, as an AID Comprehension TLV data object as defined in the ETSI TS 102 223 [6].

5.5.3.24.1.2 Parameter errors

No requirements.

5.5.3.24.1.3 Context errors

No requirements.

##### 5.5.3.24.2 Test area files

Test Source: Test\_Cre\_Apt\_Eade.java.

Test Applet: Cre\_Apt\_Eade\_1.java.

Cap File: Cre\_apt\_eade.cap.

##### 5.5.3.24.3 Test coverage

| CR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |
| CRRN3 | 1, 2 |

##### 5.5.3.24.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_APPLICATION\_DESELECT and triggering  Applet is registered to the EVENT\_APPLICATION\_DESELECT and to EVENT\_MENU\_SELECTION  Event=  EVENT\_APPLICATION\_DESELECT  1- ToolkitRegistry.isEventSet() method is called  2- Select for activation ADF1  3- Select for activation ADF2  4- Select for termination ADF2 | 1- Method returns true  2- Applet is not triggered  3- Applet is triggered  The envelope handler contains the AID of ADF1  4- Applet is triggered  The envelope handler contains the AID of ADF2 |  |
| 2 | Applet deregistration  Event= EVENT\_APPLICATION\_DESELECT  ToolkitRegistry.clearEvent() method is called  Perform UICC initialization  1- Select for activation ADF1  2- Select for activation ADF2  3- Select for termination ADF2  4- An Envelope menu selection is sent to the UICC  Event= EVENT\_APPLICATION\_DESELECT  ToolkitRegistry.setEvent() method is called  Perform UICC initialization  5- Select for activation ADF1  6- Select for activation ADF2  7- Select for termination ADF2 | 2- Applet is not triggered  3- Applet is not triggered  4- Applet is triggered  5- Applet is not triggered  6- Applet is triggered  The envelope handler contains the AID of ADF1  7- Applet is triggered  The envelope handler contains the AID of ADF2 |  |

#### 5.5.3.25 EVENT\_PROACTIVE\_HANDLER\_AVAILABLE

Test Area Reference: Cre\_Apt\_Epha.

##### 5.5.3.25.1 Conformance requirement

5.5.3.25.1.1 Normal execution

* CRRN1: The applet is triggered by the EVENT\_PROACTIVE\_HANDLER\_AVAILABLE once it has registered to this event and once the *ProactiveHandler* is available and all the Toolkit Applets registered to the previous event have been triggered and have returned from the *processToolkit()* invocation.
* CRRN2: The applet is not triggered by the EVENT\_PROACTIVE\_HANDLER\_AVAILABLE once it has deregistered from this event.
* CRRN3: When the Toolkit Applet is triggered it is automatically deregistered by the CAT Runtime Environment.
* CRRN4: If a CAT session ends prior to an Applet triggering, the Applet will be triggered at the next CAT session.

5.5.3.25.1.2 Parameter errors

No requirements.

5.5.3.25.1.3 Context errors

No requirements.

##### 5.5.3.25.2 Test area files

Test Source: Test\_Cre\_Apt\_Epha.java.

Test Applet: Cre\_Apt\_Epha\_1.java.

Cre\_Apt\_Epha\_2.java.

Cap File: Cre\_apt\_epha.cap.

##### 5.5.3.25.3 Test coverage

| CR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1 |
| CRRN2 | 1 |
| CRRN3 | 1 |
| CRRN4 | 2 |

##### 5.5.3.25.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE, triggering and automatic deregistration  Applet1 is registered to EVENT\_MENU\_SELECTION  Applet2 is registered to EVENT\_MENU\_SELECTION and EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY  1- Applet1 is triggered by an envelope menu selection  1.1- ToolkitRegistry.setEvent() method is called with Event = EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  1.2- ToolkitRegistry.setEvent() method is called with Event = EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY  1.3- ToolkitRegistry.isEventSet() method is called with Event = EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  Event= EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  3- ToolkitRegistry.isEventSet() method is called by Applet1  4- An envelope event download user activity is sent to the UICC  Event= EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  5- ToolkitRegistry.setEvent() method is called by Applet1  6- ToolkitRegistry.isEventSet() method is called by Applet1  Event=  EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  7- ToolkitRegistry.setEvent() method is called by Applet2  8- ToolkitRegistry.isEventSet() method is called by Applet2  Event=  EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  9- ToolkitRegistry.isEventSet() method is called by Applet1  Event=  EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  10- ToolkitRegistry.isEventSet() method is called by Applet1 | 1- Applet1 is triggered  1.1- No exception is thrown  1.2- No exception is thrown  1.3- Method returns TRUE  1.4-  2- Applet1 is triggered by EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  3- Method returns FALSE  4- Applet1 is triggered by the envelope  5- No exception is thrown  6- Method returns TRUE  Applet1 finalizes  Applet2 is triggered by the envelope  7- No exception is thrown  8- Method returns TRUE  Applet2 finalizes  Applet1 is triggered by EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  9- Method returns FALSE  Applet1 finalizes  Applet2 is triggered by EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  10- Method returns FALSE  Applet2 finalizes | 2- SW = 9000 is returned to the envelope |
| 2 | Applet triggering between 2 CAT sessions  1- Applet1 is triggered by an envelope menu selection  1.1- Applet1 prepares and sends a Display Text proactive command  2- Fetch the proactive command  3- Applet2 is triggered by an envelope menu selection  3.1 Applet2 registers to EVENT\_PROACTIVE\_HANDLER\_AVAILABLE then finalizes  4- Reset the card and send the profile download command | 1- Applet1 is triggered  1.1 Applet1 is suspended  3- Applet2 is triggered  3.1- No exception is thrown  4- Applet2 is triggered by event EVENT\_PROACTIVE\_HANDLER\_AVAILABLE |  |

#### 5.5.3.26 EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE

Test Area Reference: Cre\_Apt\_Edns.

##### 5.5.3.26.1 Conformance requirement

5.5.3.26.1.1 Normal execution

* CRRN1: The applet is triggered by the EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE once it has registered to this event and an Envelope Event Event Download Network Search Mode Change is received.
* CRRN2: The applet is not triggered by the EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE once it has deregistered from this event.

5.5.3.26.1.2 Parameter errors

No requirements.

5.5.3.26.1.3 Context errors

No requirements.

##### 5.5.3.26.2 Test area files

Test Source: Test\_Cre\_Apt\_Edns.java.

Test Applet: Cre\_Apt\_Edns\_1.java.

Cap File: Cre\_apt\_edns.cap.

##### 5.5.3.26.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.26.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE and triggering  Applet is registered to the EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE and to EVENT\_MENU\_SELECTION  event= EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE  1-ToolkitRegistry.isEventSet() method is called  2-An Envelope EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE is sent to the UICC | 1- Method returns true  2- Applet is triggered |  |
| 2 | Applet deregistration  event= EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE  ToolkitRegistry.clearEvent()method is called  1-A network search mode change event dowload is sent to the UICC  2-An Envelope menu selection is sent to the UICC  Event= EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE  ToolkitRegistry.setEvent() method is called  3-An Envelope EVENT\_EVENT\_DOWNLOAD\_NETWORK\_SEARCH\_MODE\_CHANGE is sent to the UICC | 1- Applet is not triggered  2- Applet is triggered  3- Applet is triggered |  |

#### 5.5.3.27 EVENT\_EVENT\_DOWNLOAD\_BROWSING\_STATUS

Test Area Reference: Cre\_Apt\_Edbs.

##### 5.5.3.27.1 Conformance requirement

5.5.3.27.1.1 Normal execution

* CRRN1: The applet is triggered by the EVENT\_EVENT\_DOWNLOAD\_BROWSING\_STATUS once it has registered to this event and an Envelope Event Event Download Browsing Status is received.
* CRRN2: The applet is not triggered by the EVENT\_EVENT\_DOWNLOAD\_BROWSING\_STATUS once it has deregistered from this event.

5.5.3.27.1.2 Parameter errors

No requirements.

5.5.3.27.1.3 Context errors

No requirements.

##### 5.5.3.27.2 Test area files

Test Source: Test\_Cre\_Apt\_Edbs.java.

Test Applet: Cre\_Apt\_Edbs\_1.java.

Cap File: Cre\_apt\_edbs.cap.

##### 5.5.3.27.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |

##### 5.5.3.27.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_EVENT\_DOWNLOAD\_BROWSING\_STATUS and triggering  Applet is registered to the EVENT\_EVENT\_DOWNLOAD\_BROWSING\_STATUS and to EVENT\_MENU\_SELECTION  event= EVENT\_EVENT\_DOWNLOAD\_BROWSING\_STATUS  1-ToolkitRegistry.isEventSet() method is called  2-An Envelope EVENT\_EVENT\_DOWNLOAD\_BROWSING\_STATUS is sent to the UICC | 1- Method returns true  2- Applet is triggered |  |
| 2 | Applet deregistration  event= EVENT\_EVENT\_DOWNLOAD\_BROWSING\_STATUS  ToolkitRegistry.clearEvent()method is called  1-A browsing status event dowload is sent to the UICC  An Envelope menu selection is sent to the UICC  Event= EVENT\_EVENT\_DOWNLOAD\_BROWSING\_STATUS  ToolkitRegistry.setEvent() method is called  2-An Envelope EVENT\_EVENT\_DOWNLOAD\_BROWSING\_STATUS is sent to the UICC | 1 – Applet is not triggered  2- Applet is triggered |  |

#### 5.5.3.28 EVENT\_EXTERNAL\_FILE\_UPDATE

Test Area Reference: Cre\_Apt\_Eefu

##### 5.5.3.28.1 Conformance requirement

5.5.3.28.1.1 Normal execution

* CRRN1: The applet is triggered by the EVENT\_EXTERNAL\_FILE\_UPDATE once it has registered to this event and a successful execution of an UPDATE BINARY or UPDATE RECORD or INCREASE APDU command (sent by the Terminal and received by the UICC on the I/O line) as defined in TS 102 221 [5] is performed on the associated updated file.
* CRRN2: An applet shall only be triggered once per command.
* CRRN3: The registration to this event is effective once the applet has successfully called a method registerFileEvent(…).
* CRRN4: The applet is not triggered by the EVENT\_EXTERNAL\_FILE\_UPDATE once it has deregistered from this event.
* CRRN5: The deregistration for a particular file to this event is effective once the Applet has successfully called the method deregisterFileEvent(…).
* CRRN6: A call to the method *clearEvent*(EVENT\_EXTERNAL\_FILE\_UPDATE) clears the event EVENT\_EXTERNAL\_FILE\_UPDATE from the ToolkitRegistry of the Applet i.e. the Applet is no longer triggered when a file is updated.

5.5.3.28.1.2 Parameter errors

No requirements.

5.5.3.28.1.3 Context errors

* CRRC1: The applet shall not be triggered if the UPDATE BINARY or UPDATE RECORD or INCREASE APDU command are not sent by the Terminal and received by the UICC on the I/O line.

##### 5.5.3.28.2 Test area files

Test Source: Test\_Cre\_Apt\_Eefu.java.

Test Applet: Cre\_Apt\_Eefu\_1.java.

Cap File: Cre\_apt\_eefu.cap.

##### 5.5.3.28.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 1, 2 |
| CRRN3 | 1, 2 |
| CRRN4 | 2 |
| CRRN5 | 2 |
| CRRN6 | 3 |
| CRRC1 | See ETSI TS 131 213 [12] |

##### 5.5.3.28.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet registration to EVENT\_EXTERNAL\_FILE\_UPDATE and triggering  Applet is registered to the EVENT\_MENU\_SELECTION  1- Send an envelope menu selection  Event=EVENT\_EXTERNAL\_FILE\_UPDATE  2-ToolkitRegistry.isEventSet() method is called  3- ToolkitRegistry.registerFileEvent() method with fileview is called to register to EFTARU  4- ToolkitRegistry.isEventSet() method is called  5- ToolkitRegistry.registerFileEvent() method with paths of EFLARU and EFCARU is called  6- Update EFTARU  7- Update EFLARU  8- Increase EFCARU | 1- Applet is triggered  2- Method returns false  3- No exception is thrown  4- Method returns true  5- No exception is thrown  Applet finalizes  6- Applet is triggered  7- Applet is triggered  8- Applet is triggered |  |
| 2 | Applet deregistration – case 1  1-Update EFTARU  2- ToolkitRegistry.deregisterFileEvent() method with fileview is called to deregister EFCARU  Event=EVENT\_EXTERNAL\_FILE\_UPDATE  3- ToolkitRegistry.isEventSet() method is called  4- Increase EFCARU  5- Update EFTARU  6- ToolkitRegistry.deregisterFileEvent() method with path of EFLARU is called  7- ToolkitRegistry.isEventSet() method is called  8- Update EFLARU  9- Update EFTARU  10- ToolkitRegistry.deregisterFileEvent() method with path of EFTARU is called  11- ToolkitRegistry.isEventSet() method is called  12- Update EFTARU | 1- Applet is triggered  2- No exception is thrown  3- Method returns true  4- Applet is not triggered  5- Applet is triggered  6- No exception is thrown  7- Method returns true  8- Applet is not triggered  9- Applet is triggered  10- No exception is thrown  11- Method returns false  12- Applet is not triggered |  |
| 3 | Applet deregistration – case 2  1- Send an envelope menu selection  2- ToolkitRegistry.registerFileEvent() method with fileview is called to register to EFTARU  3- ToolkitRegistry.registerFileEvent() method with paths EFLARU and EFCARU is called  4- Update EFTARU  5- Update EFLARU  6- Increase EFCARU  Event=EVENT\_EXTERNAL\_FILE\_UPDATE  7- ToolkitRegistry.clearEvent() method is called  8- ToolkitRegistry.isEventSet() method is called  9- Update EFTARU  10- Update EFLARU  11- Increase EFCARU  12- Restore EFCARU, EFTARU and EFLARU | 1- Applet is triggered  2- No exception is thrown  3- No exception is thrown  4- Applet is triggered  5- Applet is triggered  6- Applet is triggered  7- No exception is thrown  8- Method returns false  9- Applet is not triggered  10- Applet is not triggered  11- Applet is not triggered |  |

### 5.5.4 Proactive Command Sending by the CAT Runtime Environment

#### 5.5.4.1 System Proactive Commands

Test Area Reference: Cre\_Pcs\_Spco.

##### 5.5.4.1.1 Conformance requirement

5.5.4.1.1.1 Normal execution

* CRRN1: During a CAT session the CAT Runtime Environment shall send a SET UP MENU system proactive command whenever a menu entry is modified, added or removed.
* CRRN2: The CAT Runtime Environment shall use the data of the EFsume file when issuing the SET UP MENU proactive command.
* CRRN3: During a CAT session the CAT Runtime Environment shall send a SET UP MENU system proactive command whenever the EFSUME file under the DFTELECOM file is updated as defined in TS 102 222 [7].
* CRRN4: At the beginning of a CAT session, the CAT Runtime Environment shall send a SET UP MENU system proactive command, if at least one menu entry is registered and enabled by a selectable Toolkit Applet.
* CRRN5: At the beginning of a CAT session, the CAT Runtime Environment shall send a SET UP EVENT LIST system proactive command, if at least one event is registered by a selectable Toolkit Applet.
* CRRN6: During a CAT session the CAT Runtime Environment shall send a SET UP EVENT LIST system proactive command whenever the registered event list is changed.
* CRRN7: At the beginning of a CAT session, the CAT Runtime Environment shall send a POLL INTERVAL system proactive command, if at least one Toolkit Applet has requested a poll interval duration.
* CRRN8: During a CAT session the CAT Runtime Environment shall send a POLL INTERVAL or POLLING OFF system proactive command whenever the system poll interval duration is changed.
* CRRN9: The CAT Runtime Environment shall send its system proactive command(s) 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.
* CRRN10: The system proactive command shall only contain information from Toolkit Applets that are in the selectable state.
* CRRN11: If help is available for at least one Menu Entry inserted in the SET UP MENU system proactive command the CAT Runtime Environment shall indicate to the terminal that help information is available.
* CRRN12: If help is not available for all Menu Entries inserted in the SET UP MENU system proactive command the CAT Runtime Environment shall not indicate to the terminal that help information is available.
* CRRN13: If a text attribute different from the default format is provided for at least one Menu Entry, the SET UP MENU system proactive command shall contain the item text attribute list Comprehension TLV. The default format as defined in ETSI TS 123 040 [10] is '00 00 03 90'.

5.5.4.1.1.2 Parameter errors

No requirements.

5.5.4.1.1.3 Context errors

No requirements.

##### 5.5.4.1.2 Test area files

Test Source: Test\_Cre\_Pcs\_Spco.java.

Test Applet: Cre\_Pcs\_Spco\_1.java.

Cre\_Pcs\_Spco\_2.java.

Cre\_Pcs\_Spco\_3.java.

Cap File: Cre\_pcs\_spco.cap.

##### 5.5.4.1.3 Test coverage

|  |  |
| --- | --- |
| CRR number | Test case number |
| CRRN1 | see:  Api\_2\_Tkr\_Cmet, CRRN1,  Api\_2\_Tkr\_Dmet, CRRN3,  Api\_2\_Tkr\_Emet, CRRN3,  Api\_2\_Tkr\_Imet, CRRN1  Api\_2\_Tkr\_Smta, CRRN3 |
| CRRN2 | 1 |
| CRRN3 | 1 |
| CRRN4 | 2, 3, 4 |
| CRRN5 | 5, 6, 7 |
| CRRN6 | 8 |
| CRRN7 | 9, 10, 11 |
| CRRN8 | 12 |
| CRRN9 | 13 |
| CRRN10 | 2, 4, 6, 7, 10, 11 |
| CRRN11 | See Api\_2\_Tkr\_Cmet, CRRN6 |
| CRRN12 | See Api\_2\_Tkr\_Cmet, CRRN6 |
| CRRN13 | 14 |

##### 5.5.4.1.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Set Up Menu at the beginning of a CAT session  Install Applet2  Applet2 registers to EVENT\_UNRECOGNIZED\_ENVELOPE and have its access conditions set to “ALWAYS”  1- Perform UICC initialization with set up menu facilities supported  2- Select EFSUME under DFTELECOM with a select by path command  3- Update EFSUME with the text “TEST MENU” and Text Attribute “00 00 13 90” (Bold)  4- An unrecognized envelope is sent to trigger Applet2  5- An unrecognized envelope is sent to trigger Applet2  6- The EFSUME under DFTELECOM is updated with the text “UICC TEST” and restore the initial Text Attribute | 4- Applet2 selects EFSUME and updates its content with the text “TEST UICC” and no Text Attribute  5- Applet2 selects EFSUME and updates its content with the text “TEST UICC” and Text Attribute “00 00 13 90” | 1- SET UP MENU with main menu “UICC TEST”  2- SW = 9000  3- SET UP MENU with main menu “TEST MENU” and Text Attribute “00001390”  4- SET UP MENU with main menu “TEST UICC” and no Text Attribute  5- SET UP MENU with main menu “TEST UICC” and Text Attribute “00001390”  6- SET UP MENU with main menu “UICC TEST” |
| 2 | Set Up Menu at the beginning of a CAT session  1- Install Applet1  Applet1 registers to EVENT\_MENU\_SELECTION using the initMenuEntry(), to EVENT\_STATUS\_COMMAND using the requestPollIntervall() and to EVENT\_EVENT\_DOWNLOAD\_MT\_CALL, EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS |  | 1- SET UP MENU with the menu of Applet1 |
| 3 | Set Up Menu with applet in LOCK state  1- Lock Applet1 |  | 1- SET UP MENU with no menu of Applet1 |
| 4 | Set Up Menu with applet in SELECTABLE state  1- Make selectable Applet1  2- An envelope menu selection is sent to trigger Applet1  3- Applet1 disables its menu  4- An envelope event download MT call is sent to trigger Applet1  5- Applet1 enables its menu | 2- Applet1 is triggered  4- Applet1 is triggered | 1- SET UP MENU with the menu  3- SET UP MENU with no menu  5- SET UP MENU with the menu |
| 5 | Set Up Event List at the beginning of a CAT session  1- Perform UICC initialization with EVENT DOWNLOAD and set up event list facilities supported |  | 1- SET UP EVENT LIST proactive command  [Event list]= '19020003' or '99020003' |
| 6 | Setup Event List with applet in LOCK state  1- Lock Applet1 |  | 1- SET UP EVENT LIST Proactive command [CommandQualifier]= 00h |
| 7 | Setup Event List with applet in SELECTABLE state  1- Make selectable Applet1 |  | 1- SET UP EVENT LIST proactive command  [Event list]= '19020003' or '99020003' |
| 8 | Dynamic setup event list on registry modification  1- An envelope menu selection is sent to trigger Applet1  2- Applet1 deregisters to event EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  3- An unrecognized envelope is sent to trigger Applet2  4- Applet registers to event EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS and to EVENT\_EVENT\_DOWNLOAD\_LANGUAGE\_SELECTION  5- An envelope event download location status is sent to the UICC  6- Applet1 and Applet2 clear their events download  7- An envelope menu selection is sent to trigger Applet1  8- Applet1 registers to event EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  9- Delete Applet1  10- Install Applet1 (same registration as before, plus registration to EVENT\_UNRECOGNIZED\_ENVELOPE) | 1- Applet1 is triggered  3- Applet2 is triggered  5- Applet1 and Applet2 are triggered  7- Applet1 is triggered | 2- SET UP EVENT LIST proactive command  [Event list]= '190103' or '990103'  4- SET UP EVENT LIST proactive command  [Event list]= '19020307' or '99020307'  6- SET UP EVENT LIST proactive command  [CommandQualifier]= 00h    8- SET UP EVENT LIST proactive command  [Event list]= '190100' or '990100'  9- SET UP EVENT LIST proactive command  [CommandQualifier]= 00h    10- SET UP EVENT LIST proactive command  [Event list]= '19020003' or '99020003' |
| 9 | Poll Interval at the beginning of a CAT session  1- Perform UICC initialization with polling facilities supported |  | 1- POLL INTERVAL  proactive command |
| 10 | Poll Interval with applet in LOCK state  1- Lock Applet1 |  | 1- POLLING OFF proactive command |
| 11 | Poll Interval with applet in SELECTABLE state  1- Make selectable Applet1 |  | 1- POLL INTERVAL  proactive command |
| 12 | Dynamic Polling commands on registry modification  1- A status command is sent  2- Applet1 calls the method requestPollInterval() with POLL\_NO\_DURATION  3- An unrecognized envelope is sent  4- Applet1 calls the method requestPollInterval() with POLL\_SYSTEM\_DURATION  5- Delete Applet1  6- Install Applet1 (same registration as before, plus registration to EVENT\_UNRECOGNIZED\_ENVELOPE) | 1- Applet1 is triggered  3- Applet1 is triggered  Applet1 finalizes  Applet2 is triggered | 2- POLLING OFF proactive command  4- POLL INTERVAL proactive command  5- POLLING OFF proactive command  6- POLL INTERVAL proactive command |
| 13 | System Proactive Commands sending  1- Perform UICC initialization with system proactive commands facilities  2- An unrecognized envelope is sent  3- Applet1 deregisters to event EVENT\_EVENT\_DOWNLOAD\_MT\_CALL and  UNRECOGNIZED\_ENVELOPE, disables its menu entry, calls method requestPollIntervall() with POLL\_NO\_DURATION then builds and sends a Display Text Proactive command with text 'Text1'  4- Applet2 registers to event EVENT\_PROACTIVE\_HANDLER\_AVAILABLE, disable its menu entry then builds and sends a Display Text Proactive command with text 'Text21'  5- Applet2 builds and sends a Display Text Proactive command with text 'Text22'  7- Delete Applet1 | 2- Applet1 is triggered  Applet1 finalizes  Applet2 is triggered  Applet2 finalizes  Applet2 is triggered by event EVENT\_PROACTIVE\_HANDLER\_AVAILABLE  Applet2 finalizes | 3- Display Text with text 'text1' proactive command  4- Display Text with text 'text21' proactive command  5- Display Text with text 'text22' proactive command  6- SET UP MENU proactive command with no menu,  SET UP EVENT LIST  proactive command  [Event list]= '190103' or '990103' and  POLLING OFF proactive command |
| 14 | Text Attribute management in Set Up Menu  1- Install Applet3, Applet3 calls the initMenuEntry() method, then reinitialize the card.  2- Send an Unrecognized Envelope to trigger Applet2  3- Applet2 enable its menu entry  4-Send an envelope Menu Selection with the Item Id of Applet2  5- Applet2 calls setMenuEntryTextAttribute() method to set the attribute to “00 00 13 90” (Bold)  6-Send an envelope Menu Selection with the Item Id of Applet2  7- Applet2 calls disableMenuEntry()  8-Send an envelope Menu Selection with the Item Id of Applet3  9- Applet3 calls setMenuEntryTextAttribute() method to set the attribute to “00 00 23 90” (Italic)  10-Send an envelope Menu Selection with the Item Id of Applet2  11- Applet2 calls enableMenuEntry()  12-Send an envelope Menu Selection with the Item Id of Applet2  13- Applet2 calls setMenuEntryTextAttribute() method to set the attribute to “00 00 03 90” (default)  14- Lock Applet3  15- unlock Applet3  16- Send an envelope Menu Selection with the Item Id of Applet3  17- Applet3 calls setMenuEntryTextAttribute() method to set the attribute to “00 00 03 90” (default) | 2- Applet2 is triggered  Applet2 finalizes  4- Applet2 is triggered  Applet2 finalizes  6- Applet2 is triggered  Applet2 finalizes  8- Applet3 is triggered  Applet3 finalizes  10- Applet2 is triggered  Applet2 finalizes  12- Applet2 is triggered  Applet2 finalizes  16- Applet3 is triggered  Applet3 finalizes | 1- SET UP MENU proactive command with one menu and no Item Text Attribute List or the default Text Attribute List '00000390”  3- SET UP MENU proactive command two menus and no Item Text Attribute List or the default Text Attribute List '0000039000000390”  5- SET UP MENU proactive command two menus and the Item Text Attribute List “00001390 00000390”  7- SET UP MENU proactive command with one menu and no Item Text Attribute List or the default Text Attribute List '00000390”  9- SET UP MENU proactive command one menu and the Item Text Attribute List “00002390”  11- SET UP MENU proactive command two menus and the Item Text Attribute List “00001390 00002390”  13- SET UP MENU proactive command two menus and the Item Text Attribute List “00000390 00002390”  14- SET UP MENU proactive command with one menu and no Item Text Attribute List or the default Text Attribute List '00000390”  15- SET UP MENU proactive command two menus and the Item Text Attribute List “00000390 00002390”  17- SET UP MENU proactive command two menus and no Item Text Attribute List or the default Text Attribute List '0000039000000390” |

#### 5.5.4.2 Interaction with GSM commands

Test Area Reference: Cre\_Pcs\_Igco

##### 5.5.4.2.1 Conformance requirement

5.5.4.2.1.1 Normal execution

* CRRN1: The CAT Runtime Environment shall process a UICC command even when a proactive command is pending (before and after the FETCH command until the terminal response). The CAT Runtime Environment shall answer with the SW1 and SW2 described in ETSI TS 102 221 [5] and ETSI TS 102 223 [6].

5.5.4.2.1.2 Parameter errors

No requirements.

5.5.4.2.1.3 Context errors

No requirements.

##### 5.5.4.2.2 Test area files

Test Source: Test\_Cre\_Pcs\_Igco.java.

Test Applet: Cre\_Pcs\_Igco\_1.java.

Cap File: Cre\_pcs\_igco.cap.

##### 5.5.4.2.3 Test coverage

|  |  |
| --- | --- |
| CRR number | Test case number |
| CRRN1 | 1, 2, 3 |

##### 5.5.4.2.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Interaction with GSM Commands after TERMINAL PROFILE in connection with FETCH and TERMINAL RESPONSE  Applet is registered to Menu Selection  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  2- SELECT MF  3- Failed SELECT File  4- FETCH  5- SELECT MF  6- TERMINAL RESPONSE |  | 1- 91xx  2- Expected data = XX XX XX XX 3F 00  SW = 91XX  3- 6A82  4- Proactive Command: SETUP MENU  5- Expected data = XX XX XX XX 3F 00  SW = 9000  6- 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- Failed SELECT File  3- FETCH  4- SELECT MF  5- TERMINAL RESPONSE |  | 1- Expected data = XX XX XX XX 3F 00  SW = 91XX  2- 6A82  3- Proactive Command: Display Text  4- Expected data = XX XX XX XX 3F 00  SW = 9000  5- 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- FETCH  3- SELECT MF  4- Failed SELECT File  5- TERMINAL RESPONSE  6- SELECT MF  7-Failed SELECT File  8-FETCH  9-SELECT MF  10-TERMINAL RESPONSE |  | 1- Expected data = XX XX XX XX 3F 00  SW = 91XX  2- Proactive Command: Display Text  3- Expected data = XX XX XX XX 3F 00  SW = 9000  4- 6A82  5- 91XX  6- Expected data = XX XX XX XX 3F 00  SW = 91XX  7-6A82  8-Proactive Command: Display Text  9- Expected data = XX XX XX XX 3F 00  SW = 9000  10- 9000 |

#### 5.5.4.3 Proactive Command Control

Test Area Reference: Cre\_Pcs\_Pcco.

##### 5.5.4.3.1 Conformance requirement

5.5.4.3.1.1 Normal execution

* CRRN1: The CAT Runtime Environment shall prevent the toolkit applet to issue the following proactive commands: SET UP MENU, SET UP EVENT LIST, POLL INTERVAL, POLLING OFF. If an applet attempts to issue such a command, the CAT Runtime Environment shall throw an exception.
* CRRN2: The CAT Runtime Environment shall prevent a toolkit applet to issue a TIMER MANAGEMENT proactive command using a timer identifier, which is not allocated to it. If an applet attempts to issue such a command, the CAT Runtime Environment shall throw an exception.
* CRRN3: The CAT Runtime Environment shall prevent a toolkit applet to issue a SEND DATA, RECEIVE DATA and CLOSE CHANNEL proactive commands using a channel identifier, which is not allocated to it. If an applet attempts to issue such a command the CAT Runtime Environment shall throw an exception.
* CRRN4: The CAT Runtime Environment shall prevent a toolkit applet to issue an OPEN CHANNEL proactive command if it exceeds the maximum number of channel allocated to this applet. If an applet attempts to issue such a command the CAT Runtime Environment shall throw an exception.
* CRRN5: The CAT Runtime Environment shall prevent a Toolkit Applet to issue a DECLARE SERVICE (add, delete) proactive command using a service identifier, which is not allocated to it. If an applet attempts to send such a command, the CAT Runtime Environment shall throw an exception.
* CRRN6: All proactive commands shall be sent to the terminal as constructed by the Toolkit Applet without any check by the CAT Runtime Environment.
* CRRN7: The CAT Runtime Environment cannot guarantee that if the SET UP IDLE MODE TEXT proactive command is used by a Toolkit Applet, another Toolkit Applet will not overwrite this text at a later stage.

5.5.4.3.1.2 Parameter errors

No requirements.

5.5.4.3.1.3 Context errors

No requirements.

##### 5.5.4.3.2 Test area files

Test Source: Test\_Cre\_Pcs\_Pcco.java.

Test Applet : Cre\_Pcs\_Pcco\_1.java.

Cre\_Pcs\_Pcco\_2.java.

Cre\_Pcs\_Pcco\_3.java.

Cap File: Cre\_pcs\_pcco.cap.

##### 5.5.4.3.3 Test coverage

|  |  |
| --- | --- |
| CRR number | Test case number |
| CRRN1 | 1 |
| CRRN2 | 2 |
| CRRN3 | 3,4 |
| CRRN4 | 3,4 |
| CRRN5 | 5 |
| CRRN6 | 6 |
| CRRN7 | Not testable |

##### 5.5.4.3.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 0 | Applets installation  Applet1 is installed with 4 timers maximum, 0 channel maximum, 1 menu and 4 service identifiers maximum  Applet2 is installed with 8 timers maximum, 3 channels maximum, 1 menu and 8 service identifiers maximum  Applet3 is installed with 1 channel maximum, 1 menu and no service identifier |  |  |
| 1 | **STK Proactive Commands**  1- Send envelope menu selection with the item id of Applet1  2- Applet1 builds and sends a SET UP MENU proactive command  3- Applet1 builds and sends a SET UP EVENT LIST proactive command  4- Applet1 builds and sends a POLL INTERVAL proactive command  5- Applet1 builds and sends a POLLING OFF proactive command | 1- Applet1 is triggered  2- COMMAND\_NOT\_ALLOWED ToolkitException is thrown  3- COMMAND\_NOT\_ALLOWED ToolkitException is thrown  4- COMMAND\_NOT\_ALLOWED ToolkitException is thrown  5- COMMAND\_NOT\_ALLOWED ToolkitException is thrown | 1- 90 00 (no proactive command is sent) |
| 2 | **TIMER MANAGEMENT Proactive command**  1- Send envelope menu selection with the item id of Applet2  2- Applet2 allocates 8 timers by calling allocateTimer() method and release the 3 timers from id 1 to 3.  3- Send envelope menu selection with the item id of Applet1  4- Applet1 allocates 3 timers (Id 1 to 3) by calling allocateTimer() method 3 times  5- Send envelope menu selection with the item id of Applet2  6- Applet2 releases timers of Id 4 to 7  7- Send envelope menu selection with the item id of Applet1  8- For each of the 3 timers allocated by Applet1 (Id 1to 3) a TIMER MANAGEMENT proactive session is performed  9- For other timers (Id 4 to 8), Applet1 builds and sends a TIMER MANAGEMENT proactive command | 1- Applet2 is triggered  2- No exception is thrown  3- Applet1 is triggered  4- No exception is thrown  5- Applet2 is triggered  6- No exception is thrown  7- Applet1 is triggered  8- No exception is thrown  9- COMMAND\_NOT\_ALLOWED ToolkitException is thrown | 8- 3 TIMER MANAGEMENT proactive commands are fetched  9- The Status word of the last previous Terminal Response is 90 00 (no more proactive command is sent) |
| 3 | **No Channel allowed**  1 Send envelope menu selection with the item id of Applet1  2- Applet1 builds and sends a CSD OPEN CHANNEL proactive command  3- Applet1 builds and sends a GPRS OPEN CHANNEL proactive command  4- Applet1 builds and sends a SEND DATA proactive command  5- Applet1 builds and sends a RECEIVE DATA proactive command  6- Applet1 builds and sends a CLOSE CHANNEL proactive command | 1- Applet1 is triggered  2- COMMAND\_NOT\_ALLOWED ToolkitException is thrown  3- COMMAND\_NOT\_ALLOWED ToolkitException is thrown  4- COMMAND\_NOT\_ALLOWED ToolkitException is thrown  5- COMMAND\_NOT\_ALLOWED ToolkitException is thrown  6- COMMAND\_NOT\_ALLOWED ToolkitException is thrown | 1- 90 00 (no proactive command is sent) |
| 4 | **4 Channels allowed**  1- Send envelope menu selection with the item id of Applet3  2- Applet3 builds and sends a CSD OPEN CHANNEL proactive command  3- Send a Fetch and Terminal Response OK on channel 7  4- Send envelope menu selection with the item id of Applet2  5- Applet2 builds and sends a CSD OPEN CHANNEL proactive command  6- Send a Fetch and Terminal Response OK on channel 1  7- Applet2 builds and sends a GPRS OPEN CHANNEL proactive command  8- Send Fetch and Terminal Response OK on channel 2  9- For each channel id from 3 to 7, Applet2 builds and sends a SEND DATA proactive command  10- For each channel id from 3 to 7, Applet2 builds and sends a RECEIVE DATA proactive command  11- For each channel id from 3 to 7, Applet2 builds and sends a CLOSE CHANNEL proactive command  12- Applet2 builds and sends a CSD OPEN CHANNEL proactive command  13- Fetch and Terminal Response OK on channel 3  14- Applet2 builds and sends an OPEN CHANNEL proactive command | 1- Applet3 is triggered  2- No exception is thrown  4- Applet2 is triggered  5- No exception is thrown  7- No exception is thrown  9- COMMAND\_NOT\_ALLOWED ToolkitException is thrown  10- COMMAND\_NOT\_ALLOWED ToolkitException is thrown  11- COMMAND\_NOT\_ALLOWED ToolkitException is thrown  12- No exception is thrown  14- COMMAND\_NOT\_ALLOWED ToolkitException is thrown | 2- 91 1C  3- OPEN CHANNEL proactive  5- 91 1C  6- OPEN CHANNEL proactive command is fetched  7- 91 17  8- OPEN CHANNEL proactive command is fetched, SW = 91 1C on the Terminal Response  13- OPEN CHANNEL proactive command is fetched  14- 90 00 expected to the previous Terminal Response (no proactive command is sent) |
| 5 | **DECLARE SERVICE Proactive command**  1- Send envelope menu selection with the item id of Applet2  2- Applet2 allocates 8 services by calling allocateServiceIdentifier() method and release the 3 services from id 0 to 2 using method releaseServiceIdentifier().  3- Send envelope menu selection with the item id of Applet1  4- Applet1 allocates 3 services (Id 0 to 2) by calling allocateServiceIdentifier() method 3 times  5- Send envelope menu selection with the item id of Applet2  6- Applet2 releases services of Id 5 to 7  7- Send envelope menu selection with the item id of Applet1  8- For each of the 3 services allocated by Applet1 (Id 0 to 2) DECLARE SERVICE (add) proactive commands are sent  9- For other services (Id 3 to 8), Applet1 builds and sends a DECLARE SERVICE (add) proactive command  10- For each of the 3 services allocated by Applet1 (Id 0 to 2) DECLARE SERVICE (delete) proactive commands are sent  11- For other services (Id 3 to 8), Applet1 builds and sends a DECLARE SERVICE (delete) proactive command | 1- Applet2 is triggered  2- No exception is thrown  3- Applet1 is triggered  4- No exception is thrown  5- Applet2 is triggered  6- No exception is thrown  7- Applet1 is triggered  8- No exception is thrown  9- COMMAND\_NOT\_ALLOWED ToolkitException is thrown  10- No exception is thrown  11- COMMAND\_NOT\_ALLOWED ToolkitException is thrown | 8- 3 DECLARE SERVICE proactive commands are fetched  9- The Status word of the last previous Terminal Response is 91 1C on the Terminal Response  10- 3 DECLARE SERVICE proactive commands are fetched  11- The Status word of the last previous Terminal Response is 90 00 (no more proactive command is sent) |
| 6 | **Unknown proactive command**  1- Send an envelope menu selection with the item id of Applet1  2- Applet1 builds an unknown proactive command  3- Fetch and terminal response OK | 1- Applet1 is triggered | 2- 91 08  3- The unknown proactive command is fetched |

### 5.5.5 Exception Handling

#### 5.5.5.1 General Behaviour

Test Area Reference: Cre\_Exh\_Genb.

##### 5.5.5.1.1 Conformance requirement

5.5.5.1.1.1 Normal execution

* CRRN1: If more than one Applet shall be triggered by the currently processed event all Exceptions shall be caught by the CAT Runtime Environment and shall not be sent to the terminal. The CAT Runtime Environment shall proceed with the triggering.
* CRRN2: If only one Applet shall be triggered by the currently processed event and an ISOException with the reason code REPLY\_BUSY is thrown, it shall be sent to the terminal using the Status Word 0x9300.
* CRRN3: If only one Applet shall be triggered by the currently processed event other Exceptions than an ISOException with the reason code REPLY\_BUSY shall not be propagated to the terminal.

5.5.5.1.1.2 Parameter errors

No requirements.

5.5.5.1.1.3 Context errors

No requirements.

##### 5.5.5.1.2 Test area files

Test Source: Test\_Cre\_Exh\_Genb.java.

Test Applet: Cre\_Exh\_Genb\_1.java.

Cre\_Exh\_Genb\_2.java.

Cap File: Cre\_exh\_genb.cap.

##### 5.5.5.1.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1 |
| CRRN2 | 2 |
| CRRN3 | 3 |

##### 5.5.5.1.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 0 | Applet1 is installed and registers to EVENT\_MENU\_SELECTION and EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  Applet2 is installed and registers to EVENT\_EVENT\_DOWNLOAD\_MT\_CALL and EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY |  |  |
| 1 | ISOException REPLY\_BUSY is not sent to the terminal in multi triggering  1- Send an envelope Event Download MT Call (multi triggering event, multi registered applets)  4- Send an envelope Event Download MT Call (multi triggering event, multi registered applets) | 1- Applet1 is triggered  2- Applet1 sends a ISOException with the reason code REPLY\_BUSY then finalizes  Applet2 is triggered, does nothing and finalizes  4- Applet1 is triggered, does nothing and finalizes  Applet2 is triggered, sends a ISOException with the reason code REPLY\_BUSY then finalizes | 3- SW = 90 00  5- SW = 90 00 |
| 2 | ISOException REPLY\_BUSY is sent to the terminal in single triggering  1- Send an envelope Menu Selection to trigger Applet1 (single triggering event)  2- Send an envelope Event Download User Activity (multi triggering event, single registered applet) | 1- Applet1 is triggered, sends a ISOException with the reason code REPLY\_BUSY then finalizes  2- Applet2 is triggered, sends a ISOException with the reason code REPLY\_BUSY then finalizes | 1- SW = 93 00  2- SW = 93 00 |
| 3 | Other exception than ISOException REPLY\_BUSY are not sent to the terminal  1- Send an envelope Menu Selection to trigger Applet1 (single triggering event)  2- Send an envelope Menu Selection to trigger Applet1 (single triggering event) | 1- Applet1 is triggered, sends a ISOException with reason code different to REPLY\_BUSY then finalizes  2- Applet1 is triggered, sends a ToolkitException then finalizes | 1- SW = 90 00  2- SW = 90 00 |

#### 5.5.5.2 Interaction with Multiple Triggering

Test Area Reference: Cre\_Exh\_Imtg.

##### 5.5.5.2.1 Conformance requirement

5.5.5.2.1.1 Normal execution

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

5.5.5.2.1.2 Parameter errors

No requirements.

5.5.5.2.1.3 Context errors

No requirements.

##### 5.5.5.2.2 Test area files

Test Source: Test\_Cre\_Exh\_Imtg.java.

Test Applet: Cre\_Exh\_Imtg\_1.java.

Cre\_Exh\_Imtg\_2.java.

Cap File: Cre\_exh\_imtg.cap.

##### 5.5.5.2.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2, 3, 4 |

##### 5.5.5.2.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 0 | Load/install 2 toolkit applets registered to EVENT\_STATUS\_COMMAND, EVENT\_PROFILE\_DOWNLOAD, EVENT\_UNRECOGNIZED\_ENVELOPE, EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  Applet1: Priority= 0x01,  Applet2: Priority= 0x02,  (i.e. Applet1 is triggered before Applet2) |  |  |
| 1 | Profile\_Download is sent | 1- Applet1 is triggered  2- NullPointerException is thrown  3- Applet2 is triggered |  |
| 2 | Status\_Command is sent | 1- Applet1 is triggered    2- NullPointerException is thrown  3- Applet2 is triggered |  |
| 3 | UNRECOGNIZED\_Envelope is sent | 1- Applet1 is triggered  2- NullPointerException is thrown  3- Applet2 is triggered |  |
| 4 | Event\_Download\_MT\_Call is sent | 1- Applet1 is triggered    2- NullPointerException is thrown  3- Applet2 is triggered |  |

### 5.5.6 Envelope Response Posting

#### 5.5.6.1 General Behaviour

Test Area Reference: Cre\_Erp\_Genb.

##### 5.5.6.1.1 Conformance requirement

5.5.6.1.1.1 Normal execution

* CRRN1: A Toolkit Applet can post a response to some events with the *post()* or the *postAsBERTLV()* methods and can continue its processing after the call to these methods.
* CRRN2: The CAT Runtime Environment shall send the response before the emission of the next proactive command or when all the Toolkit Applets triggered by the event have finished their processing.
* CRRN3: The Boolean parameter passed to the *post()* or *postAsBERTLV()* method shall be mapped by the CAT Runtime Environment to the correct status word. If the value is true it corresponds to a successful ending of the command status word "9000". If the value is false it corresponds to a warning status word "6200".

5.5.6.1.1.2 Parameter errors

No requirements.

5.5.6.1.1.3 Context errors

No requirements.

##### 5.5.6.1.2 Test area files

None.

##### 5.5.6.1.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | See Api\_2\_Erh\_Postb: CRRN1  See Api\_2\_Erh\_Postbb: CRRN1 |
| CRRN2 | See Api\_2\_Erh\_Postb: CRRN3  See Api\_2\_Erh\_Postbb: CRRN3 |
| CRRN3 | See Api\_2\_Erh\_Postb: CRRN4  See Api\_2\_Erh\_Postbb: CRRN4 |

##### 5.5.6.1.4 Test procedure

None.

#### 5.5.6.2 EVENT\_CALL\_CONTROL\_BY\_NAA

Test Area Reference: Cre\_Erp\_Eccn.

##### 5.5.6.2.1 Conformance requirement

5.5.6.2.1.1 Normal execution

* CRRN1: The CAT Runtime Environment can't reply busy when an Envelope(Call Control) is sent to the UICC.

5.5.6.2.1.2 Parameter errors

No requirements.

5.5.6.2.1.3 Context errors

No requirements.

##### 5.5.6.2.2 Test area files

Test Source: Test\_Cre\_Erp\_Eccn.java.

Test Applet: Cre\_Erp\_Eccn\_1.java.

Cre\_Erp\_Eccn\_2.java.

Cre\_Erp\_Eccn\_3.java.

Cap File: Cre\_erp\_eccn.cap.

##### 5.5.6.2.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1, 2 |

##### 5.5.6.2.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applet1 is registered on the EVENT\_CALL\_CONTROL\_BY\_NAA, 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 UICC  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 UICC  5- A Terminal Response command is sent to the UICC  6- Delete Applet1 & Applet2  7- Install Applet3 | 1- Applet2 is suspended  2- Applet1 is triggered  5- Applet2's execution shall continue | 3- The dialling number is retrieved and the status words is 91xx |
| 2 | Applet3 is registered on both the events EVENT\_CALL\_CONTROL\_BY\_NAA and EVENT\_MENU\_SELECTION  1- Envelope Menu Selection is sent to the UICC  2- Applet3 invokes the method send()and no fetch is performed  3- Envelope(Call Control) is sent to the UICC  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 UICC  6- A Terminal Response command is sent to the UICC | 1- Applet3 is triggered on the EVENT\_MENU\_SELECTION  2- Applet3 is suspended on the send() method  3- Applet3 is triggered on the EVENT\_CALL\_CONTROL\_BY\_NAA  6- The Applet3's execution shall continue | 4- The dialling number is retrieved and the status words is 91xx |

#### 5.5.6.3 EVENT\_UNRECOGNIZED\_ENVELOPE

Test Area Reference: Cre\_Erp\_Euen.

##### 5.5.6.3.1 Conformance requirement

5.5.6.3.1.1 Normal execution

* CRRN1: The EnvelopeResponseHandler is available for the EVENT\_UNRECOGNIZED\_ENVELOPE.

5.5.6.3.1.2 Parameter errors

No requirements.

5.5.6.3.1.3 Context errors

No requirements.

##### 5.5.6.3.2 Test area files

Test Source: Test\_Cre\_Erp\_Euen.java.

Test Applet: Cre\_Erp\_Euen\_1.java.

Cap File: Cre\_erp\_euen.cap.

##### 5.5.6.3.3 Test coverage

| CRR Number | Test Case Number |
| --- | --- |
| CRRN1 | 1 |

##### 5.5.6.3.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | An applet triggered on the EVENT\_UNRECOGNIZED\_ENVELOPE calls the EnvelopeResponseHandler.post() method | The post() method returns no exception | The UICC answers to the Envelope with status words 9000. The data retrieved are the ones posted by the applet. |

### 5.5.7 Toolkit Installation

#### 5.5.7.1 General Behaviour

Test Area Reference: Cre\_Tin\_Genb

##### 5.5.7.1.1 Conformance requirement

5.5.7.1.1.1 Normal execution

* CRRN1: The UICC Toolkit Application specific parameters (Tag 80h) are mandatory for applications using the *uicc.toolkit.ToolkitInterface* defined in ETSI TS 102 241 [9].
* CRRN2: Any additional parameters of the UICC Toolkit Application specific parameters field (Tag 80h) shall be ignored by the card.
* CRRN3: Some unused byte may be added at the end of the UICC Toolkit Application specific parameters field (Tag 80h).
* CRRN4: The UICC Access Application specific parameters (Tag 81h) are applicable to applications using the *uicc.access.FileView* defined in ETSI TS 102 241 [9].
* CRRN5: The UICC Toolkit Application specific parameters field (Tag 80h) is not required for applications that do not use the *uicc.toolkit.ToolkitInterface* defined in ETSI TS 102 241 [9].
* CRRN6: The UICC Access Application specific parameters field (Tag 81h) is not required for applications that a do not use the *uicc.access.FileView* defined in ETSI TS 102 241 [9].

5.5.7.1.1.2 Parameter errors

No requirements.

5.5.7.1.1.3 Context errors

No requirements.

##### 5.5.7.1.2 Test area files

Test Source: Test\_Cre\_Tin\_Genb.java.

Test Applet: Cre\_Tin\_Genb\_1.java (use *uicc.toolkit.ToolkitInterface*).

Cre\_Tin\_Genb\_2.java (use *uicc.access.FileView*).

Cre\_Tin\_Genb\_3.java (use *uicc.toolkit.ToolkitInterface* and *uicc.access.FileView*).

Cap File: Cre\_tin\_genb.cap.

##### 5.5.7.1.3 Test coverage

| CRR number | Test case number |
| --- | --- |
| CRRN1 | 1, 2 |
| CRRN2 | 2 |
| CRRN3 | 2 |
| CRRN4 | 3 |
| CRRN5 | 3 |
| CRRN6 | 1, 2 |

##### 5.5.7.1.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Good installation with the only tag 80h  1- Install(install) Applet1 with only the UICC Toolkit Application specific parameters field |  | 1- RAPDU = 00 90 00 |
| 2 | Good installation with the only tag 80h  1- Install(install) Applet1 with only the UICC Toolkit Application specific parameters field which contains some unused bytes |  | 1- RAPDU = 00 90 00 |
| 3 | Good installation with the only tag 81h  1- Install(install) Applet2 with only the UICC Access Application specific parameters field |  | 1- RAPDU = 00 90 00 |

#### 5.5.7.2 Timers Allocation

Test Area Reference: Cre\_Tin\_Tmal.

##### 5.5.7.2.1 Conformance requirement

5.5.7.2.1.1 Normal execution

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

5.5.7.2.1.2 Parameter errors

No requirements.

5.5.7.2.1.3 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. If the maximum number of timers required is greater than '08' (maximum numbers of timers specified in ETSI TS 102 223 [6], the card shall return the Status Word '6A80', incorrect parameters in data field, to the Install(Install) command.

##### 5.5.7.2.2 Test area files

Test Source: Test\_Cre\_Tin\_Tmal.java.

Test Applet: Cre\_Tin\_Tmal\_1.java.

Cre\_Tin\_Tmal\_2.java.

Cre\_Tin\_Tmal\_3.java.

Cap File: Cre\_tin\_tmal.cap.

##### 5.5.7.2.3 Test coverage

| CRR number | Test case number |
| --- | --- |
| CRRN1 | 2, 3, 8 |
| CRRC1 | 1, 7 |
| CRRC2 | 4, 5, 6 |

##### 5.5.7.2.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | **More than 8 timers at the instantiation of Applet1: check that Applet1 is not installed.**  1- Install Applet1 with maximum 9 timers allocated  2- Applet1 is selected |  | 1- The installation failed with the status word 6A80  2- Applet1 is not found, RAPDU != <applet selected data> 90 00 |
|  | Reset the card |  |  |
| 2 | **Good installation of Applet2**  1- Install Applet2 with maximum 4 timers allocated |  |  |
| 3 | **Allocate 4 timers**  **Applet2**  1- An envelope menu selection is send to trigger Applet2  2- Applet2 allocates 4 timers | 2- No exception shall be thrown |  |
| 4 | **Allocate one more timer**  **Applet2**  1- Applet2 allocates one more timer | 1- Shall throw a ToolkitException with reason NO\_TIMER\_AVAILABLE |  |
| 5 | **Good installation of applet3**  1- Install Applet3 with maximum 8 timers allocated |  |  |
| 6 | **Allocate 4 timers**  **Applet3**  1- an envelope menu selection is send to trigger Applet3  2- Applet3 allocates 4 timers | 2-No exception shall be thrown |  |
| 7 | **Allocate one more timer**  **Applet3**  1- Applet3 allocates one more timer | 1- 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** |  |  |

#### 5.5.7.3 Item Identifier

Test Area Reference: Cre\_Tin\_Itid

##### 5.5.7.3.1 Conformance requirement

5.5.7.3.1.1 Normal execution

* CRRN1: If the requested item identifier in the range [1 to 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].

5.5.7.3.1.2 Parameter errors

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

5.5.7.3.1.3 Context errors

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

##### 5.5.7.3.2 Test area files

Test Source: Test\_Cre\_Tin\_Itid.java.

Test Applet: Cre\_Tin\_Itid\_1.java.

Cre\_Tin\_Itid\_2.java.

Cre\_Tin\_Itid\_3.java.

Cap File: Cre\_tin\_itid.cap.

##### 5.5.7.3.3 Test coverage

| CRR number | Test case number |
| --- | --- |
| CRRN1 | 2 |
| CRRN2 | 4, 5, 6 |
| CRRP1 | 1 |
| CRRC1 | 3 |

##### 5.5.7.3.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Bad installation of Applet1  1- Installation of Applet1  The following parameters item Id equal to 128  2- Applet1 is selected |  | 1- The installation failed with the status word 6A80  2- Applet1 is not found, RAPDU != <applet selected data> 90 00 |
| 2 | Good installation of Applet1  1- Installation of Applet1  Item Id = 1 for the first menu and 127 for the second one  2- A Terminal Profile is sent to the card with only PROFILE\_DOWNLOAD, MENU\_SELECTION, SET\_UP\_MENU and COMMAND\_RESULT facilities. |  | 2- The UICC answers 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**  1- Installation of Applet2  item Id = 127  2- Applet2 is selected |  | 1- The installation failed with the status word 6A80  2- Applet2 is not found, RAPDU != <applet selected data> 90 00 |
| 4 | Good installation of Applet2  1- Installation of Applet2  item Id = 0 |  | 1- The UICC answers 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  1- Installation of Applet3  item Id = 0 |  | 1- The UICC answers 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 deletion and installation of Applet2  1- Delete instance of Applet2  2- Install for install of Applet2  item Id = 0 |  | 1- The UICC answers with status words 91xx to send back to the ME the 3 menus  The menus are  01/01/menu11  02/127/menu12  03/129/menu31  3- The UICC answers 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 |

#### 5.5.7.4 Item Position

Test Area Reference: Cre\_Tin\_Itpo.

##### 5.5.7.4.1 Conformance requirement

5.5.7.4.1.1 Normal execution

* CRRN1: If the new Menu Entry has to be inserted at an already occupied position, the entries from the requested position to the last element of the Menu Entries' list are shifted to the next positions.
* CRRN2: If the position indicated is greater than the number of elements in the Menu Entries' list, then the Menu Entry takes the last position in the Menu Entries' list.
* CRRN3: If the position indicated is equal to '00', then the Menu Entry takes the last position in the Menu Entries' list.

5.5.7.4.1.2 Parameter errors

No requirements.

5.5.7.4.1.3 Context errors

No requirements.

##### 5.5.7.4.2 Test area files

Test Source: Test\_Cre\_Tin\_Itpo.java.

Test Applet: Cre\_Tin\_Itpo\_1.java.

Cre\_Tin\_Itpo\_2.java.

Cre\_Tin\_Itpo\_3.java.

Cre\_Tin\_Itpo\_4.java.

Cre\_Tin\_Itpo\_5.java.

Cre\_Tin\_Itpo\_6.java.

Cap File: Cre\_tin\_itpo.cap.

##### 5.5.7.4.3 Test coverage

| CRR number | Test case number |
| --- | --- |
| CRRN1 | 1 to 10 |
| CRRN2 | 5 |
| CRRN3 | 4 |

##### 5.5.7.4.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Installation of Applet1  1- Install Applet1  Position/ItemId  01/01  02/02  03/03  04/04 |  | 1- The UICC answers to the Envelope with status words 91xx to send back to the ME the 4 menus  The menus are  (position(1)/itemId/text)  01/01/menu11  02/02/menu12  03/03/menu13  04/04/menu14  (1) position is the position in the set up menu proactive command |
| 2 | Installation of Applet2  1- Install Applet2  Position/ItemId  03/05 |  | 1- The UICC 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  03/05/menu21  04/03/menu13  05/04/menu14 |
| 3 | Installation of Applet3  1- Install Applet3  Position/ItemId  02/06  03/07 |  | 1- The UICC answers to the Envelope with status words 91xx to send back to the ME the 7 menus  The menus are  (position/itemId/text)  01/01/menu11  02/06/menu31  03/07/menu32  04/02/menu12  06/05/menu21  07/03/menu13  08/04/menu14 |
| 4 | Installation of Applet4  1- Install Applet4  Position/ItemId  00/08 |  | 1- The UICC answers to the Envelope with status words 91xx to send back to the ME the 8 menus  The menus are  (position/itemId/text)  01/01/menu11  02/06/menu31  03/07/menu32  04/02/menu12  06/05/menu21  07/03/menu13  08/04/menu14  09/08/menu41 |
| 5 | Installation of Applet5  1- Install Applet5  Position/ItemId  20/09 |  | 1- The UICC answers to the Envelope with status words 91xx to send back to the ME the 9 menus  The menus are  (position/itemId/text)  01/01/menu11  02/06/menu31  03/07/menu32  04/02/menu12  06/05/menu21  07/03/menu13  08/04/menu14  09/08/menu41  10/09/menu51 |
| 6 | Disabling of the first menu of Applet1 and locking of Applet2  1- An envelope menu selection is sent with Item Id = 02  2- Applet1 disables its first menu (Item Id = 01)  3- Lock the Applet2 | 1- Applet1 is triggered | 2- The UICC answers to the Envelope with status words 91xx to send back to the ME the 08 menus  The menus are  (position/itemId/text)  01/06/menu31  02/07/menu32  03/02/menu12  05/05/menu21  06/03/menu13  07/04/menu14  08/08/menu41  09/09/menu51  3- The UICC answers to the Envelope with status words 91xx to send back to the ME the 07 menus  The menus are  (position/itemId/text)  01/06/menu31  02/07/menu32  03/02/menu12  05/03/menu13  06/04/menu14  07/08/menu41  08/09/menu51 |
| 7 | Installation of Applet6  1- Install Applet6  Position/ItemId  01/10  04/11  15/12 |  | 1- The UICC answers to the Envelope with status words 91xx to send back to the ME the 10 menus  The menus are  (position/itemId/text)  01/10/menu61  02/06/menu31  03/11/menu62  04/07/menu32  05/02/menu12  07/03/menu13  08/04/menu14  09/08/menu41  10/09/menu51  11/12/menu63 |
| 8 | Enabling of the first menu of Applet1 and unlocking of Applet2  1- An envelope menu selection is sent with Item Id = 02  2- Applet1 enables its first menu (Item Id = 01)  3- Unlock the Applet2 | 1- Applet1 is triggered | 2- The UICC answers to the Envelope with status words 91xx to send back to the ME the 11 menus  The menus are  (position/itemId/text)  01/10/menu61  02/01/menu11  03/06/menu31  04/11/menu62  05/07/menu32  06/02/menu12  08/03/menu13  09/04/menu14  10/08/menu41  11/09/menu51  12/12/menu63  3- The UICC answers to the Envelope with status words 91xx to send back to the ME the 12 menus  The menus are  (position/itemId/text)  01/10/menu61  02/01/menu11  03/06/menu31  04/11/menu62  05/07/menu32  06/02/menu12  08/05/menu21  09/03/menu13  10/04/menu14  11/08/menu41  12/09/menu51  13/12/menu63 |
| 09 | Deletion of Applet2  1- Delete Applet2 |  | 1- The UICC answers to the Envelope with status words 91xx to send back to the ME the 11 menus  The menus are  (position/itemId/text)  01/10/menu61  02/01/menu11  03/08/menu31  04/11/menu62  05/07/menu32  06/02/menu12  08/03/menu13  09/04/menu14  10/08/menu61  11/09/menu51  12/12/menu63 |
| 10 | Installation of Applet2  1- Install Applet2  Position/ItemId  03/05 |  | 1- The UICC answers to the Envelope with status words 91xx to send back to the ME the 12 menus  The menus are  (position/itemId/text)  01/10/menu61  02/01/menu11  03/05/menu21  04/06/menu31  05/11/menu62  06/07/menu32  07/02/menu12  09/03/menu13  10/04/menu14  11/08/menu41  12/09/menu51  13/12/menu63 |

#### 5.5.7.5 Maximum Text Length for a menu entry

Test Area Reference: Cre\_Tin\_Mlme.

##### 5.5.7.5.1 Conformance requirement

5.5.7.5.1.1 Normal execution

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

5.5.7.5.1.2 Parameter 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.

5.5.7.5.1.3 Context errors

No requirements.

##### 5.5.7.5.2 Test area files

Test Source: Test\_Cre\_Tin\_Mlme.java.

Test Applet: Cre\_Tin\_Mlme\_1.java.

Cap File: Cre\_tin\_mlme.cap.

##### 5.5.7.5.3 Test coverage

| CRR number | Test case number |
| --- | --- |
| CRRN1 | 1, 3, 4 |
| CRRP1 | 2 |
| CRRP2 | 5 |

##### 5.5.7.5.4 Test procedure

| Id | Description | API / Framework Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Installation of applet with 2 menus not exceeding the maximum text length  Install one applet with 3 menu entries allowed and max. text length equal to 10.  initMenuEntry defined at the install (install) command  MenuEntry = "MenuEntry1", "MenuEntry2"  Offset = 0  Length = 10  NextAction = '00'  HelpSupported = false  IconQualifier = '00'  IconIdentifier = 0 |  |  |
| 2 | initMenuEntry with a too large length  initMenuEntry with length equal to 11  MenuEntry = " MenuEntry03"  Offset = 0  Length = 11  NextAction = '00'  HelpSupported = false  IconQualifier = '00'  IconIdentifier = 0 | ToolkitException ALLOWED\_LENGTH\_EXCEEDED is thrown |  |
| 3 | initMenuEntry with a right length  initMenuEntry with length parameter equal to 10  MenuEntry = " MenuEntry3"  Offset = 0  Length = 10  NextAction = '00'  HelpSupported = false  IconQualifier = '00'  IconIdentifier = 0 |  | a SET UP MENU (3 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 (3 items) is issued with TLV item length equal to 11 (Identifier + Text string of item) |
| 5 | changeMenuEntry with a too large length  Applet1 is triggered by a EVENT\_MENU\_SELECTION.  ChangeMenuEntry of menu 1, with length parameter equal to 11  Id = '02'  MenuEntry = "MenuEntry05"  Offset = 0  Length = menuEntry.length  NextAction = 0  HelpSupported = false  IconQualifier = 0  IconIdentifier = 0  Return from processToolkit | ToolkitException ALLOWED\_LENGTH\_EXCEEDED is thrown | SW = 90 00 |

#### 5.5.7.6 Maximum number of menu entries

Test Area Reference: Cre\_Tin\_Nbme.

##### 5.5.7.6.1 Conformance requirement

5.5.7.6.1.1 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.

5.5.7.6.1.2 Parameter errors

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

5.5.7.6.1.3 Context errors

No requirements.

##### 5.5.7.6.2 Test area files

Test Source: Test\_Cre\_Tin\_Nbme.java.

Test Applet: Cre\_Tin\_Nbme\_1.java.

Cre\_Tin\_Nbme\_2.java.

Cap File: Cre\_tin\_nbme.cap.

##### 5.5.7.6.3 Test coverage

| CRR number | Test case number |
| --- | --- |
| CRRN1 | 1 |
| CRRP1 | 2, 3 |

##### 5.5.7.6.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Installation of applet with 3 menus  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 | No Exception is thrown |  |
| 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: “menu1”, “menu2”, “menu3”) |
| 3 | Installation of 2nd applet with 0 menu  1- Install (install) another applet, with max. number of menu entry is '0', defined at the install (install) command.  initMenuEntry once  MenuEntry = "menu5"  Offset = 0  Length = 5  NextAction = '00'  HelpSupported = false  IconQualifier = '00'  IconIdentifier = 0  2- Perform a RESET and a Terminal Profile with the facilities of PROFILE\_DOWNLOAD, MENU\_SELECTION, COMMAND\_RESULT and SET\_UP\_MENU | ToolkitException REGISTRY\_ERROR is thrown | 2- SET UP MENU (3 items: “menu1”, “menu2”, “menu3”) |

#### 5.5.7.7 Access Domain

Test Area Reference: Cre\_Tin\_Acdo.

##### 5.5.7.7.1 Conformance requirement

5.5.7.7.1.1 Normal execution

* CRRN1: The UICC access specific parameters (Tag 80h) indicate the mechanism used to control the application instance access to the File System ('00' means full access to the File System, '02' means UICC access mechanism and 'FF' means no access to the File System).
* CRRN2: The UICC access specific parameters are applicable to applications using the *uicc.access.FileView* defined in ETSI TS 102 241 [9].
* CRRN3: The UICC administrative access parameter (Tag 83h) indicate the mechanism used to control the application instance access to the File System ('00' means full access to the File System, '02' means UICC access mechanism and 'FF' means no access to the File System).
* CRRN4: The UICC administrative access parameters are applicable to applications using the *uicc.access.fileadministration.AdminFileView* defined in ETSI TS 102 241 [9].
* CRRN5: If an application has Access Domain Parameter '00' (i.e. Full Access to the File System), all actions can be performed on a file except the ones with NEVER access condition.
* CRRN6: If an application has Access Domain Parameter '02' (i.e. UICC access mechanism). The UICC access mechanism shall be coded as follows:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte 1 : |  | b8 | | b7 | | b6 | | b5 | | b4 | | b3 | | b2 | | b1 | |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | Application PIN 1 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | Application PIN 2 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | Application PIN 3 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | Application PIN 4 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | Application PIN 5 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | Application PIN 6 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | Application PIN 7 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | Application PIN 8 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte 2 : |  | b8 | | b7 | | b6 | | b5 | | b4 | | b3 | | b2 | | b1 | |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | ADM1 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | ADM2 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | ADM3 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | ADM4 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | ADM5 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | ADM6 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | ADM7 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | ADM8 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte 3 : |  | b8 | | b7 | | b6 | | b5 | | b4 | | b3 | | b2 | | b1 | |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | ADM9 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | ADM10 |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | ALWAYS |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | Local PIN ( only applicable for ADF ) |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | RFU |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | RFU |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | RFU |
|  |  | |  | |  | |  | |  | |  | |  | |  | |  | | RFU |

These access rights shall be checked against SE ID 01 access rules as defined in ETSI TS 102 221 [5].

5.5.7.7.1.2 Parameter 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 application with Access Domain Parameter 'FF' (i.e. No Access to the File System) tries to access a file the CAT Runtime Environement shall throw a UICCException with a SECURITY\_STATUS\_NOT\_SATISFIED reason.
* CRRP3: If an application with Access Domain Parameter '02' (i.e. UICC access mechanism) tries to access a file without the correct rights, the CAT Runtime Environement shall throw a UICCException with a SECURITY\_STATUS\_NOT\_SATISFIED reason.

5.5.7.7.1.3 Context errors

No requirements.

##### 5.5.7.7.2 Test area files

Test Source: Test\_Cre\_Tin\_Acdo.java.

Test Applet: Cre\_Tin\_Acdo\_1.java.

Cre\_Tin\_Acdo\_2.java.

Cre\_Tin\_Acdo\_3.java.

Cre\_Tin\_Acdo\_4.java.

Cre\_Tin\_Acdo\_5.java.

Cre\_Tin\_Acdo\_6.java.

Cre\_Tin\_Acdo\_7.java.

Cre\_Tin\_Acdo\_8.java.

Cap File: Cre\_tin\_acdo.cap.

##### 5.5.7.7.3 Test coverage

| CRR number | Test case number |
| --- | --- |
| CRRN1 | 1 to 6 |
| CRRN2 | 1 to 6 |
| CRRN3 | 1 to 6 |
| CRRN4 | 1 to 6 |
| CRRN5 | 1 |
| CRRN6 | 3 to 6 |
| CRRP1 | Not tested |
| CRRP2 | 2 |
| CRPP3 | 3 to 6 |

##### 5.5.7.7.4 Test procedure

The following table summarizes tests performed in the test procedure.

|  | | Cyclic files (EFCARRx) | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Linear fixed files(EFLARRx) | | | | |  |
| Transparent files(EFTARRx) | | | |  |  |
| Applet rights | File rights | Activate | Deactivate | Read Bin/Rec | Update  Bin/Rec | Search | Increase | Create | Delete | Resize |
| Full access | Always  (EFxARR1) | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| Never  (EFxARR2) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Global PIN1 & ADM1  (EFxARR3) | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| Global PIN1 | ADM1  (EFxARR4) | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| Local PIN 1 / ADM2 (See note)  (EFxARR5) | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| No Access | Always  (EFxARR1) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Never  (EFxARR2) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Global PIN1 & ADM1  (EFxARR3) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Global PIN1 | ADM1  (EFxARR4) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Local PIN 1 / ADM2 (See note)  (EFxARR5) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Always | Always  (EFxARR1) | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| Never  (EFxARR2) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Global PIN1 & ADM1  (EFxARR3) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Global PIN1 | ADM1  (EFxARR4) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Local PIN 1 / ADM2 (See note)  (EFxARR5) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Global PIN1 | Always  (EFxARR1) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Never  (EFxARR2) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Global PIN1 & ADM1  (EFxARR3) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Global PIN1 | ADM1  (EFxARR4) | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| Local PIN 1 / ADM2 (See note)  (EFxARR5) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Global PIN1 & ADM1 | Always  (EFxARR1) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Never  (EFxARR2) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Global PIN1 & ADM1  (EFxARR3) | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| Global PIN1 | ADM1  (EFxARR4) | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| Local PIN 1 / ADM2 (See note)  (EFxARR5) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Local PIN & ADM2 | Always  (EFxARR1) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Never  (EFxARR2) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Global PIN1 & ADM1  (EFxARR3) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Global PIN1 | ADM1  (EFxARR4) | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK | NOK |
| Local PIN 1 / ADM2 (See note)  (EFxARR5) | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| NOTE: For EFXARR5, the file access condition is Local PIN 1 if the file is located under the MF and ADM2 if it is located under ADF1. | | | | | | | | | | |

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Full access Applet  0- Applet1 installation with full access right  1- Send an envelope Menu Selection to trigger Applet1 on menu Id 1  2- Send an envelope Menu Selection to trigger Applet1 on menu Id 2  3- Applet1 deletion | 1- Applet1 is triggered and gets Fileviews on UICC and ADF1  1.1- For each EFCARRx Applet1 calls all associated methods with success, except on file EFCARR2 where UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown.  1.2- For each EFLARRx Applet1 calls all associated methods with success, except on file EFLARR2 where UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown.  1.3- For each EFTARRx Applet1 calls all associated methods with success, except on file EFTARR2 where UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown.  2- Applet1 is triggered and gets AdminFileviews on UICC and ADF1  2.1- For each EFCARRx Applet1 calls all associated methods with success, except on file EFCARR2 where UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown.  2.2- For each EFLARRx Applet1 calls all associated methods with success, except on file EFLARR2 where UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown.  2.3- For each EFTARRx Applet1 calls all associated methods with success, except on file EFTARR2 where UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown.  2.4- Under each DFARRx Applet1 resize the EFTARxT with success, except on file EFTAR2T where an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown.  2.5- Under each DFARRx Applet1 delete the EFTARxT with success, except on file EFTAR2T where an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown.  2.6- Under each DFARRx Applet1 create the EFTARxT with success, except under DFARR2 where an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown at creation of a file similar to EFTAR2T. | 1- SW = 90 00  2- SW = 90 00 |
| 2 | No access Applet  0- Applet2 installation with no access right  1- Send an envelope Menu Selection to trigger Applet2 on menu Id 1  2- Send an envelope Menu Selection to trigger Applet2 on menu Id 2  3- Applet2 deletion | 1- Applet2 is triggered and gets Fileviews on UICC and ADF1  1.1- For each EFCARRx Applet2 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown.  1.2- For each EFLARRx Applet2 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown.  1.3- For each EFTARRx Applet2 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown.  2- Applet2 is triggered and gets AdminFileviews on UICC and ADF1  2.1- For each EFCARRx Applet2 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown.  2.2- For each EFLARRx Applet2 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown.  2.3- For each EFTARRx Applet2 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown.  2.4- Under each DFARRx Applet2 resize the EFTARxT; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown.  2.5- Under each DFARRx Applet2 delete the EFTARxT; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown.  2.6- Under each DFARRx Applet2 create an EF; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown. | 1- SW = 90 00  2- SW = 90 00 |
| 3 | Always access right Applet  0- Applet3 installation with Always access right  1- Send an envelope Menu Selection to trigger Applet3 on menu Id 1  2- Send an envelope Menu Selection to trigger Applet3 on menu Id 2  3- Applet3 deletion | 1- Applet3 is triggered and gets Fileviews on UICC and ADF1  1.1- For each EFCARRx Applet3 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFCARR1 where the execution is successful.  1.2- For each EFLARRx Applet3 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFLARR1 where the execution is successful.  1.3- For each EFTARRx Applet3 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFTARR1 where the execution is sucessful.  2- Applet3 is triggered and gets AdminFileviews on UICC and ADF1  2.1- For each EFCARRx Applet3 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFCARR1 where the execution is successful.  2.2- For each EFLARRx Applet3 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFLARR1 where the execution is successful.  2.3- For each EFTARRx Applet3 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFTARR1 where the execution is sucessful.  2.4- Under each DFARRx Applet3 resize the EFTARxT; an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown, except on file EFTAR1T where the execution is successful.  2.5- Under each DFARRx Applet3 delete the EFTARxT; an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown, except on file EFTAR1T where the execution is successful.  2.6- Under each DFARRx Applet3 create an EF like the EFTARxT; an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown, except on file EFTAR1T where the execution is successful. | 1- SW = 90 00  2- SW = 90 00 |
| 4 | Global PIN1 access right Applet  0- Applet4 installation with Global PIN1 access right  1- Send an envelope Menu Selection to trigger Applet4 on menu Id 1  2- Send an envelope Menu Selection to trigger Applet4 on menu Id 2  3- Applet4 deletion | 1- Applet4 is triggered and gets Fileviews on UICC and ADF1  1.1- For each EFCARRx Applet4 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFCARR4 where the execution is successful.  1.2- For each EFLARRx Applet4 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFLARR4 where the execution is successful.  1.3- For each EFTARRx Applet4 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFTARR4 where the execution is sucessful.  2- Applet4 is triggered and gets AdminFileviews on UICC and ADF1  2.1- For each EFCARRx Applet4 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFCARR4 where the execution is successful.  2.2- For each EFLARRx Applet4 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFLARR4 where the execution is successful.  2.3- For each EFTARRx Applet4 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFTARR4 where the execution is sucessful.  2.4- Under each DFARRx Applet4 resize the EFTARxT; an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown, except on file EFTAR4T where the execution is successful.  2.5- Under each DFARRx Applet4 delete the EFTARxT; an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown, except on file EFTAR4T where the execution is successful.  2.6- Under each DFARRx Applet4 create an EF like the EFTARxT; an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown, except on file EFTAR4T where the execution is successful. | 1- SW = 90 00  2- SW = 90 00 |
| 5 | Global PIN1 & ADM1 access right Applet  0- Applet5 installation with Global PIN1 & ADM1 access right  1- Send an envelope Menu Selection to trigger Applet5 on menu Id 1  2- Send an envelope Menu Selection to trigger Applet5 on menu Id 2  3- Applet5 deletion | 1- Applet5 is triggered and gets Fileviews on UICC and ADF1  1.1- For each EFCARRx Applet5 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on files EFCARR3 and EFCARR4 where the execution is successful.  1.2- For each EFLARRx Applet5 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on files EFCARR3 and EFCARR4 where the execution is successful.  1.3- For each EFTARRx Applet5 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on files EFCARR3 and EFCARR4 where the execution is sucessful.  2- Applet3 is triggered and gets AdminFileviews on UICC and ADF1  2.1- For each EFCARRx Applet5 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on files EFCARR3 and EFCARR4 where the execution is successful.  2.2- For each EFLARRx Applet5 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on files EFCARR3 and EFCARR4 where the execution is successful.  2.3- For each EFTARRx Applet5 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on files EFCARR3 and EFCARR4 where the execution is sucessful.  2.4- Under each DFARRx Applet5 resize the EFTARxT; an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown, except on files EFCARR3 and EFCARR4 where the execution is successful.  2.5- Under each DFARRx Applet5 delete the EFTARxT; an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown, except on files EFCARR3 and EFCARR4 where the execution is successful.  2.6- Under each DFARRx Applet5 create an EF like the EFTARxT; an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown, except on files EFCARR3 and EFCARR4 where the execution is successful. | 1- SW = 90 00  2- SW = 90 00 |
| 6 | Local Pin & ADM2 access right Applet  0- Applet6 installation with Local PIN & ADM2 access right  1- Send an envelope Menu Selection to trigger Applet6 on menu Id 1  2- Send an envelope Menu Selection to trigger Applet6 on menu Id 2  3- Applet6 deletion | 1- Applet6 is triggered and gets Fileviews on UICC and ADF1  1.1- For each EFCARRx Applet6 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFCARR5 where the execution is successful.  1.2- For each EFLARRx Applet6 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFLARR5 where the execution is successful.  1.3- For each EFTARRx Applet6 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFTARR5 where the execution is sucessful.  2- Applet6 is triggered and gets AdminFileviews on UICC and ADF1  2.1- For each EFCARRx Applet6 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFCARR5 where the execution is successful.  2.2- For each EFLARRx Applet6 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFLARR5 where the execution is successful.  2.3- For each EFTARRx Applet6 calls all associated methods; UICCException.SECURITY\_STATUS\_NOT\_SATISFIED are thrown, except on file EFTARR5 where the execution is sucessful.  2.4- Under each DFARRx Applet6 resize the EFTARxT; an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown, except on file EFTAR5T where the execution is successful.  2.5- Under each DFARRx Applet6 delete the EFTARxT; an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown, except on file EFTAR5T where the execution is successful.  2.6- Under each DFARRx Applet6 create an EF like the EFTARxT; an UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown, except on file EFTAR5T where the execution is successful. | 1- SW = 90 00  2- SW = 90 00 |
| 7 | AdminFileview and Fileview acces domain parameter differenciation  0- Applet7 installation with Always & ADM1 access right for Fileview access domain  1- Send an envelope Menu Selection on menu id 1 to trigger Applet7  2- Applet7 deletion  3- Applet7 installation with Global PIN1 & ADM2 access right for AdminFileView access domain  4- Send an envelope Menu Selection on menu id 2 to trigger Applet7  5- Applet7 deletion  6- Applet7 installation with Always & ADM1 access right for Fileview access domain and Global PIN1 & ADM2 access right for AdminFileView access domain,  7- Send an envelope Menu Selection on menu id 3 to trigger Applet7  8- Applet7 deletion | 1- Applet7 is triggered and gets Fileview on UICC.  1.1- Using the Fileview, Applet7 reads file EFTARR1. No exception is expected  1.2- Using the Fileview, Applet7 reads file EFTARR5. An UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown  1.3- Using the Fileview, Applet7 reads file EFTARR4. No exception is expected  4- Applet7 is triggered and gets AdminFileview on UICC.  4.1- Using the AdminFileview, Applet7 reads file EFTARR1. An UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown  4.2- Using the AdminFileview, Applet7 reads file EFTARR5. No exception is expected  4.3- Using the AdminFileview, Applet7 reads file EFTARR4. No exception is expected  7- Applet7 is triggered and gets Fileview and AdminFileview on UICC.  7.1- Using the Fileview, Applet7 reads file EFTARR1. No exception is expected  7.2- Using the AdminFileview, Applet7 reads file EFTARR1. An UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown  7.3- Using the AdminFileview, Applet7 reads file EFTARR5. No exception is expected  7.4- Using the Fileview, Applet7 reads file EFTARR5. An UICCException.SECURITY\_STATUS\_NOT\_SATISFIED is thrown  7.5- Using the Fileview, Applet7 reads file EFTARR4. No exception is expected  7.6- Using the AdminFileview, Applet7 reads file EFTARR4. No exception is expected | 1- SW = 90 00  4- SW = 90 00  7- SW = 90 00 |

#### 5.5.7.8 Priority Level

Test Area Reference: Cre\_Tin\_Prlv.

##### 5.5.7.8.1 Conformance requirement

5.5.7.8.1.1 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).

5.5.7.8.1.2 Parameter errors

No requirements.

5.5.7.8.1.3 Context errors

No requirements.

##### 5.5.7.8.2 Test area files

Test Source: Test\_Cre\_Tin\_Prlv\_x.java, x from 1 to 12, 8A, 8B, 9A, 9B, 10A, 10B.

Test Applet: Cre\_Tin\_Prlv\_x.java, x from 1 to 12, 8A, 8B, 9A, 9B, 10A, 10B.

Cap File: Cre\_tin\_prlv\_x.cap, x from 1 to 12, 8A, 8B, 9A, 9B, 10A, 10B.

##### 5.5.7.8.3 Test coverage

| CRR number | Test case number |
| --- | --- |
| CRRN1 | 1, 2, 3, 4, 6, 8, 10, 12 |
| CRRN2 | 5, 7, 9, 11 |

##### 5.5.7.8.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 0 | All applets are registered on an EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event |  |  |
| 1 | Trigger 2 applets with 2 different maximum Priority Levels  1- Install (install) Applet1 with priority level '2' and Applet2 with priority level '1', from package Cre\_tin\_prlv\_1  2- Send an Envelope that triggers the 2 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event  3- Delete applets instances and packages | 2- A static variable is used to validate triggering order: Applet2 is triggered before Applet1 |  |
| 2 | Trigger 2 applets with 2 different maximum Priority Levels  1- Install (install) Applet1 with priority level '1' and Applet2 with priority level '2', from package Cre\_tin\_prlv\_2.  2- Send an Envelope that triggers the 2 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event.  3- Delete applets instances and packages | 2- A static variable is used to validate triggering order: Applet1 is triggered before Applet2 |  |
| 3 | Trigger 2 applets with 2 different Priority Levels  1- Install (install) Applet1 with priority level '80' and Applet2 with priority level '7F', from package Cre\_tin\_prlv\_3.  2- Send an Envelope that triggers the 2 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event  3- Delete applets instances and packages | 2- A static variable is used to validate triggering order: Applet2 is triggered before Applet1 |  |
| 4 | Trigger 2 applets with 2 different Priority Levels  1- Install (install) Applet1 with priority level '7F' and Applet2 with priority level '80', from package Cre\_tin\_prlv\_4  2- Send an Envelope that triggers the 2 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event  3- Delete applets instances and packages | 2- A static variable is used to validate triggering order: Applet2 is triggered before Applet1 |  |
| 5 | Trigger 3 applets with the same Priority Level  1- Install (install) applet 1, 2, 3 in this order with same priority level from package Cre\_tin\_prlv\_5  2- Send an Envelope that triggers the 3 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event  3- Delete applets instances and packages | 2-A static variable is used to validate triggering order: Applet3 is triggered before Applet2, and Applet2 is triggered before Applet1 |  |
| 6 | Trigger 2 applets from 2 classes, with 2 different Priority Level  1- Install (install) Applet1 from class A with priority level '2'  Install (install) Applet2 from class B with priority level '1'  2- Send an Envelope that triggers the 2 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event  3- Delete applets instances and packages | 2- A static variable is used to validate triggering order: Applet2 is triggered before Applet1 |  |
| 7 | Trigger 2 applets from 2 classes, with the same Priority Level  1- Install (install) Applet1 from class A with priority level '1'  Install (install) Applet2 from class B with priority level '1'  2- Send an Envelope that triggers the 2 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event  3- Delete applets instances and packages | 2- A static variable is used to validate triggering order: Applet2 is triggered before Applet1 |  |
| 8 | Trigger 2 applets from 2 packages, with 2 different Priority Level  1- Install package Cre\_tin\_prlv\_8.  Install (install) Applet1 from package Cre\_tin\_prlv\_8A with priority level '2'  Install (install) Applet2 from package Cre\_tin\_prlv\_8B with priority level '1'  2- Send an Envelope that triggers the 2 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event  3- Delete applets instances ad packages | 2- A static variable is used to validate triggering order: Applet2 is triggered before Applet1 |  |
| 9 | Trigger 2 applets from 2 packages, with the same Priority Level  1- Install package Cre\_tin\_prlv\_9.  Install (install) Applet1 from package Cre\_tin\_prlv\_9A and Applet2 from package Cre\_tin\_prlv\_9B in this order, with same priority level  2- Send an Envelope that triggers the 2 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event  3- Delete applets instances and packages | 2- A static variable is used to validate triggering order: Applet2 is triggered before Applet1 |  |
| 10 | Trigger 4 applets from 2 packages  1- Install packages Cre\_tin\_prlv\_10, Cre\_tin\_prlv\_10A and Cre\_tin\_prlv\_10B Install (install) 2 applets 1 then 2 from package Cre\_tin\_prlv\_10A, with respectively priority levels 1 and 2  2- Send an Envelope that triggers the 2 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event  3- Install (install) 2 applets 3 then 4 from package Cre\_tin\_prlv\_10B, with respectively priority levels 1 and 2  4- Send an Envelope that triggers the 4 applets  5- Delete applets instances and packages | 2- A static variable is used to validate triggering order: Applet1 is triggered before Applet2  4- 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) applets 1, 2, 3, 4 in this order with same priority level from package Cre\_tin\_prlv\_11  2- Send an Enveloppe that triggers the 4 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event  3- Delete applet instance 4  4- Send an Enveloppe that triggers the 3 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event  5- Delete applet instance 3  6- Send an Enveloppe that triggers the 2 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event  7- Delete remaining applet instances and packages | 2- A static variable is used to validate triggering order: applets are triggered in order 4, 3, 2, 1  4- Applets are triggered in order 3, 2, 1  6- 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    2- Send an Enveloppe that triggers the 4 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event  3- Delete applet instance 1 and install (install) applet5 with priority level 2  4- Send an Enveloppe that triggers the 4 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event  5- Re-install (install) Applet1 with priority level 1  6- Send an Enveloppe that triggers the 5 applets with the EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY event | 2- A static variable is used to validate triggering order: applets are triggered in order 3, 1, 4, 2  4- Applets are triggered in order 3, 5, 4, 2  6- Applets are triggered in order 1, 3, 5, 4, 2 |  |

#### 5.5.7.9 Channel Allocation

Test Area Reference: Cre\_Tin\_Chal.

##### 5.5.7.9.1 Conformance requirement

5.5.7.9.1.1 Normal execution

* CRRN1: One toolkit applet can register to several channels, but a channel can only be allocated to one toolkit applet.

5.5.7.9.1.2 Parameter errors

No requirements.

5.5.7.9.1.3 Context errors

* CRRC1 : Allocated channels shall not exceed the maximum number of channels allowed for this applet instance.
* CRRC2 : The total number of channels allocated for all the applets shall not exceed 7. If the maximum number of channels required is greater than '07' (maximum numbers of channels specified in ETSI TS 102 223 [6]), the card shall return the Status Word '6A80', incorrect parameters in data field, to the Install(Install) command.

##### 5.5.7.9.2 Test area files

Test Source: Test\_Cre\_Tin\_Chal.java.

Test Applet: Cre\_Tin\_Chal\_1.java.

Cre\_Tin\_Chal\_2.java.

Cre\_Tin\_Chal\_3.java.

Cap File: Cre\_tin\_chal.cap.

##### 5.5.7.9.3 Test coverage

| CRR number | Test case number |
| --- | --- |
| CRRN1 | 2,3 |
| CRRC1 | 1, 7 |
| CRRC2 | 4,5,6 |

##### 5.5.7.9.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | **More than 7 channels at the instantiation of Applet1: check that Applet1 is not installed**  1-Install for install of Applet1 with maximum 8 channels allocated  2- Select the Applet1 |  | 2- SW = 6A 80 |
|  | **Reset the card** |  |  |
| 2 | **Good installation of Applet2**  Install for install of Applet2 (maximum 4 channels allocated). |  | The UICC answers with status words 90 00 |
| 3 | **Open 4 channels**  **Applet2**  Applet2 builds a proactive command OPEN CHANNEL 4 times, calling init() and send() methods. | No exception shall be thrown. | OPEN CHANNEL proactive command are fetched.  Successful TERMINAL RESPONSE of OPEN CHANNEL are sent to the UICC with Channel Id = 01 to 04 |
| 4 | **Open one more channel**  **Applet2**  Applet2 builds a proactive command OPEN CHANNEL once again, calling init() and send() methods | Shall throw a ToolkitException with reason COMMAND\_NOT\_ALLOWED |  |
| 5 | **Good installation of applet3**  Install for install of Applet3 (maximum 7 channels allocated) |  | The UICC answers with status words 90 00 |
| 6 | Open 3 channels  Applet3  Applet3 builds a proactive command OPEN CHANNEL 3 times, calling init() and send() methods | No exception shall be thrown. | OPEN CHANNEL proactive command is fetched  Successful TERMINAL RESPONSE of OPEN CHANNEL are sent to the UICC with Channel Id from 05 to 07 |
| 7 | Open one more channel  Applet3  Applet3 builds a proactive command OPEN CHANNEL once again, calling init() and send() methods | No exception shall be thrown. | OPEN CHANNEL proactive command is fetched.  Unsuccessful Terminal Response is sent to the UICC with 'No Channel Available' as Additional Information on Result |

#### 5.5.7.10 Minimum Security Level

Test Area Reference: Cre\_Tin\_Mslv.

##### 5.5.7.10.1 Conformance requirement

5.5.7.10.1.1 Normal execution

* CRRN1: The Receiving Entity shall check the Minimum Security Level during processing the security of the Command Packet.
* CRRN2: The Receiving Entity shall reject the message if the MSL check fails.
* CRRN3: If the check fails, the Receiving Entity shall reject the messages and a Response Packet with the 'Insufficient Security Level' Response Status Code shall be sent if required.
* CRRN4: If the length of the Minimum Security Level field is greater than zero, the Minimum Security Level is used to specify the minimum level of security to be applied to Secured Packets. The first byte shall be the MSL Parameter, other bytes shall be the MSL Data.
* CRRN5: If the length of the Minimum Security Level field is zero, no minimum security level check shall be performed by the receiving entity.

5.5.7.10.1.2 Parameter errors

No requirements.

5.5.7.10.1.3 Context errors

No requirements.

##### 5.5.7.10.2 Test area files

None.

##### 5.5.7.10.3 Test coverage

| CRR number | Test case number |
| --- | --- |
| CRRN1 | Not applicable |
| CRRN2 | Not applicable |
| CRRN3 | Not applicable |
| CRRN4 | Not applicable |
| CRRN5 | Not applicable |

##### 5.5.7.10.4 Test procedure

Not applicable.

#### 5.5.7.11 TAR Value(s) of the Toolkit Application instance

Test Area Reference: Cre\_Tin\_Tarv.

##### 5.5.7.11.1 Conformance requirement

5.5.7.11.1.1 Normal execution

* CRRN1: It is possible to define several TAR Values at the installation of a Toolkit Application.
* CRRN2: If the length of TAR Value(s) is zero, the TAR may be taken out of the AID, if any.
* CRRN3: If the length of the TAR Value(s) is greater than zero then the application instance shall be installed with the TAR Value(s) field defined above and the TAR indicated in the AID if any shall be ignored.

5.5.7.11.1.2 Parameter errors

No requirements.

5.5.7.11.1.3 Context errors

* CRRC1 If a TAR Value(s) is already assigned on the card for a Toolkit Application instance, the card shall return the Status Word '6A80', incorrect parameters in data field, to the INSTALL [for install] command.
* CRRC1 If the length of TAR Value(s) field is incorrect, the card shall return the Status Word '6A80', incorrect parameters in data field, to the INSTALL [for install] command.

##### 5.5.7.11.2 Test area files

Test Source: Test\_Cre\_Tin\_Tarv.java.

Test Applet: Cre\_Tin\_Tarv\_1.java.

Cre\_Tin\_Tarv\_2.java.

Cap File: Cre\_tin\_tarv.cap.

##### 5.5.7.11.3 Test coverage

| CRR number | Test case number |
| --- | --- |
| CRRN1 | 1 (but partially tested only) |
| CRRN2 | Not applicable |
| CRRN3 | Not applicable |
| CRRC1 | 1 |
| CRRC2 | 2 |

##### 5.5.7.11.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | TAR value(s) already allocated  1- Install Applet2 with the TAR “020202”  2- Install Applet1 with the TAR values:  “010101”  “020202”  “030303” |  | 1- SW = RAPDU = 00 90 00  2- SW = 6A 80 |
| 2 | Bad TAR value(s)  1- Install Applet1 with the TAR values:  “010101”  “0303” |  | 1- SW = 6A 80 |

#### 5.5.7.12 Services Allocation

Test Area Reference: Cre\_Tin\_Sval.

##### 5.5.7.12.1 Conformance requirement

5.5.7.12.1.1 Normal execution

* CRRN1: One toolkit applet can allocates several services, but a service can only be allocated to one toolkit applet.

5.5.7.12.1.2 Parameter errors

No requirements.

5.5.7.12.1.3 Context errors

* CRRC1: Allocated services shall not exceed the maximum number of services allowed for this applet instance defined during installation.
* CRRC2: The total number of services allocated for all the applets shall not exceed 8. If the maximum number of services required is greater than '08' (maximum numbers of services specified in ETSI TS 102 223 [6]), the card shall return the Status Word '6A80', incorrect parameters in data field, to the Install(Install) command.

##### 5.5.7.12.2 Test area files

Test Source: Test\_Cre\_Tin\_Sval.java.

Test Applet: Cre\_Tin\_Sval\_1.java.

Cre\_Tin\_Sval\_2.java.

Cre\_Tin\_Sval\_3.java.

Cap File: Cre\_tin\_sval.cap.

##### 5.5.7.12.3 Test coverage

| CRR number | Test case number |
| --- | --- |
| CRRN1 | 2, 3, 8 |
| CRRC1 | 1, 7 |
| CRRC2 | 4, 5, 6 |

##### 5.5.7.12.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | **More than 8 services at the instantiation of Applet1: check that Applet1 is not installed.**  1- Install Applet1 with maximum 9 services allocated |  | 1- SW = 6A 80 |
|  | Reset the card |  |  |
| 2 | **Good installation of Applet2**  1- Install Applet2 with maximum 4 services allocated |  | 1- RAPDU = 00 90 00 |
| 3 | **Allocate 4 services**  **Applet2** | No exception shall be thrown |  |
| 4 | **Allocate one more service**  **Applet2** | Shall throw a ToolkitException with reason NO\_SERVICE\_ID\_AVAILABLE |  |
| 5 | **Good installation of applet3**  1- Install Applet3 with maximum 8 services allocated |  | 1- RAPDU = 00 90 00 |
| 6 | **Allocate 4 services**  **Applet3** | No exception shall be thrown |  |
| 7 | **Allocate one more service**  **Applet3** | Shall throw a ToolkitException with reason NO\_SERVICE\_ID\_AVAILABLE |  |
| 8 | **Check that each service identifier (allocated by Applet2 and applet3) is between 0 and 7 and is different from each other** |  |  |

### 5.5.8 UICC File Access

#### 5.5.8.1 FileView

Test Area Reference: Cre\_Ufa\_View.

##### 5.5.8.1.1 Conformance requirement

5.5.8.1.1.1 Normal execution

* CRRN1: Any Applet (not only Toolkit Applets) is allowed to retrieve and use a *FileView*.
* CRRN2: The UICC *FileView* can be retrieved by invoking the *getTheUICCView()* method from the *UICCSystem*.
* CRRN3: An ADF *FileView* can be retrieved by invoking the *getTheFileView(…)* method with passing as parameter the full AID of the application owning the ADF.
* CRRN4: The UICC *FileView* allows to access the MF and all DFs and EFs that are located under the MF, including DF Telecom and any access technology specific DF located under the MF, but not the files located under any ADF.
* CRRN5: An ADF *FileView* allows to access only the DFs and EFs located under the ADF.
* CRRN6: Each *FileView* object shall be provided as a permanent JCRE entry point object.
* CRRN7: A separate and independent file context shall be associated with each and every *FileView* object: the operation performed on files in a given *FileView* object shall not affect the file context associated with any other *FileView* object.
* CRRN8: The file context can be transient or persistent depending on what was required by the Applet during the creation of the *FileView* object.
* CRRN9: Each *FileView* shall be given the access control privileges associated with the UICC File System or the corresponding ADF for the Applet.
* CRRN10: The access control privileges are checked each time a method of the *FileView* object is invoked. The access control privileges are defined by the access domain parameters specified in ETSI TS 102 226 [8].
* CRRN11: The root of the context of a *FileView* object is the MF for the UICC *FileView*.
* CRRN12: The root of the context of a *FileView* object is the ADF for an ADF *FileView*.
* CRRN13: When the transient context of a *FileView* is cleared, the current DF becomes the root of the *FileView*.

5.5.8.1.1.2 Parameter errors

No requirements.

5.5.8.1.1.3 Context errors

* CRRC1: It is not possible to access the MF or any DF or EF located under the MF from an ADF *FileView.*

##### 5.5.8.1.2 Test area files

Test Source: Test\_Cre\_Ufa\_View.java.

Test Applet: Cre\_Ufa\_View\_1.java.

Cap File: Cre\_ufa\_view.cap.

##### 5.5.8.1.3 Test coverage

| CRR number | Test case number |
| --- | --- |
| CRRN1 | 1, 2, 3 |
| CRRN2 | 1, 2, 3 and see also Api\_1\_Cont test cases for CRRN1 |
| CRRN3 | 1, 2, 3 and see also Api\_1\_Cont test cases for CRRN1 |
| CRRN4 | 2 and see also Api\_1\_Cont test cases for CRRN1 |
| CRRN5 | 3 and see also Api\_1\_Cont test cases for CRRN1 |
| CRRN6 | 4 |
| CRRN7 | 5, 7 and see also Api\_1\_Cont test cases for CRRN1 |
| CRRN8 | 6 |
| CRRN9 | See Cre\_Tin\_Acdo |
| CRRN10 | See Cre\_Tin\_Acdo |
| CRRN11 | 5, 6 |
| CRRN12 | 5, 6 |
| CRRN13 | 6 |
| CRRC1 | 3 |

##### 5.5.8.1.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | Applets can get a *FileView*  1- A toolkit applet, Applet1, and a non toolkit applet, Applet2, are installed with full access to the file system  2- An envelope menu selection is sent to trigger Applet1  3- Applet1 gets the UICC *FileView* using method getUICCFileView() and the ADF1 *FileView* getTheFileView()  4- Send an APDU to trigger Applet2  5- Applet2 gets the UICC *FileView* using method getUICCFileView() and the ADF1 *FileView* getTheFileView() | 2- Applet1 is triggered  3- No exception is thrown  4- Applet2 is triggered  5- No exception is thrown |  |
| 2 | Applets can only access all files under the MF using the UICC file view  1- An envelope menu selection is sent to trigger Applet1  2- Applet1 gets the UICC *FileView*  3- Applet1 selects the MF  4- Applet1 selects the EFDIR  5- Applet1 selects the DFTELECOM  6- Applet1 selects the MF  7- Applet1 selects the DFTEST  8- Applet1 selects the DFTEST1  9- Applet1 selects the EFTAA  10- Applet1 tries to select DFADF1  11- Send an APDU to trigger Applet2  12- Applet2 gets the UICC *FileView*  13- Applet2 selects the MF  14- Applet2 selects the EFDIR  15- Applet2 selects the DFTELECOM  16- Applet2 selects the MF  17- Applet2 selects the DFTEST  18- Applet2 selects the DFTEST1  19- Applet2 selects the EFTAA  20- Applet2 tries to select DFADF1 | 1- Applet1 is triggered  3- No exception is thrown  4- No exception is thrown  5- No exception is thrown  6- No exception is thrown  7- No exception is thrown  8- No exception is thrown  9- No exception is thrown  10- An exception is thrown  11- Applet2 is triggered  13- No exception is thrown  14- No exception is thrown  15- No exception is thrown  16- No exception is thrown  17- No exception is thrown  18- No exception is thrown  19- No exception is thrown  20- An exception is thrown |  |
| 3 | Applets can access all files under the ADF1 using the ADF1 file view  1- An envelope menu selection is sent to trigger Applet1  2- Applet1 gets the ADF1 *FileView*  3- Applet1 selects the ADF1  4- Applet1 selects the DFTELECOM  5- Applet1 selects the ADF1  6- Applet1 selects the DFTEST  7- Applet1 selects the DFTEST1  8- Applet1 selects the EFTAA  9- Applet1 tries to select the MF  10- Applet1 tries to select the EFDIR  11- Applet1 tries to select DFADF2  12- Send an APDU to trigger Applet2  13- Applet2 gets the ADF1 *FileView*  14- Applet2 selects the ADF1  15- Applet2 selects the DFTELECOM  16- Applet2 selects the ADF1  17- Applet2 selects the DFTEST  18- Applet2 selects the DFTEST1  19- Applet2 selects the EFTAA  20- Applet1 tries to select the MF  21- Applet1 tries to select the EFDIR  22- Applet2 tries to select DFADF2 | 1- Applet1 is triggered  3- No exception is thrown  4- No exception is thrown  5- No exception is thrown  6- No exception is thrown  7- No exception is thrown  8- No exception is thrown  9- An exception is thrown  10- An exception is thrown  11- An exception is thrown  12- Applet2 is triggered  14- No exception is thrown  15- No exception is thrown  16- No exception is thrown  17- No exception is thrown  18- No exception is thrown  19- No exception is thrown  20- An exception is thrown  21- An exception is thrown  22- An exception is thrown |  |
| 4 | FileView object shall be provided as a permanent JCRE entry point object  1- An envelope menu selection is sent to trigger Applet1  2- Applet1 gets an UICC *FileView* in a static field  3- Applet1 gets an ADF1 *FileView* in a static field  4- Applet1 gets an UICC *FileView* in a field of the toolkit applet  5- Applet1 gets an ADF1 *FileView* in a field of the toolkit applet  6- Send an APDU to trigger Applet2  7- Applet2 gets an UICC *FileView* in a static field  8- Applet2 gets an ADF1 *FileView* in a static field  9- Applet2 gets an UICC *FileView* in a field of the toolkit applet  10- Applet2 gets an ADF1 *FileView* in a field of the toolkit applet | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  4- No exception is thrown  5- No exception is thrown  6- Applet2 is triggered  7- No exception is thrown  8- No exception is thrown  9- No exception is thrown  10- No exception is thrown |  |
| 5 | Context independence on FileView object  *The following sequence shall be performed twice, once with the UICC FileView, then once with the ADF1 FileView*  1- An envelope menu selection is sent to trigger Applet1  2- Applet1 gets 2 UICC(*or ADF1*) *FileView* and stores them in objects FileView1 and FileView2  3- Applet1 selects DFTEST/EFCARU using the FileView1 object  4- Applet1 calls the readRecord() method in the NEXT mode using the FileView1 object  5- Applet1 calls the readRecord() method in the NEXT mode using the FileView1 object  6- Applet1 calls the readRecord() method in the NEXT mode using the FileView2 object  7- Applet1 selects EFLARU using the FileView2 object  8- Applet1 selects DFtest using the FileView2 object  9- Applet1 selects EFCARU using the FileView2 object  10- Applet1 calls the readRecord() method in the NEXT mode using the FileView2 object  *The same test sequence is done using the non toolkit applet Applet2* | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  4- The record value is “55 55 55”  5- The record value is “AA AA AA”  6- An exception is thrown  7- An exception is thrown  8- No exception is thrown  9- No exception is thrown  10- The record value is “55 55 55” |  |
| 6 | File Context can be transient or persistent  *The following sequence shall be performed twice, once with the UICC FileView, then once with the ADF1 FileView*  1- An envelope menu selection is sent to trigger Applet1  2- Applet1 gets 2 UICC(*or ADF1*) *FileView*and stores one in a transient object FileView1 and the other in a persistent object FileView2  3- Applet1 selects DFTEST/EFCARU using the FileView1 object then the FileView2 object  4- Applet1 calls the readRecord() method in the NEXT mode using the FileView1 object then the FileView2 object  5- Reset the card  6- An envelope menu selection is sent to trigger Applet1  7- Applet1 calls the status() command using the FileView1  8- Applet1 calls the status() command using the FileView2  9- Applet1 calls the readRecord() method in the NEXT mode using the FileView2  10- Applet1 calls the readRecord() method in the NEXT mode using the FileView1 | 1- Applet1 is triggered  2- No exception is thrown  3- No exception is thrown  4- The record value is “55 55 55”  6- Applet1 is triggered  7- The current DF is DFTEST  8- The current DF is the root (MF or ADF1)  9- The record value is “AA AA AA”  10- An exception is thrown |  |
| 7 | File Context integrity  1- An envelope menu selection is sent to trigger Applet1  2- Applet1 gets a UICC *FileView* and selects DFTEST, with it  3- Applet1 sends a Display Text proactive command  4- Send a fetch command  5- An envelope call control by NAA is sent  6- Applet1 selects DFTEST/DFTEST1, using the previous UICC *FileView*, then finalizes  7- Send terminal response of Display Text command  8- Applet1 resumes and calls status() command, using the same UICC *FileView* | 1- Applet1 is triggered  2- No exception is thrown  3- SW = 91 XX  4- Display Text is fetched  5- Applet1 is triggered  6- No exception is thrown  7- SW = 90 00  8- The current DF is DFTEST1 |  |

#### 5.5.8.2 File Access

Test Area Reference: Cre\_Ufa\_Facc.

Shall be covered in the API access part.

### 5.5.9 Other parts transferred to framework from API

#### 5.5.9.1 A handler is a temporary JCRE Entry Point object

Test Area Reference: Cre\_Api\_Hepo.

##### 5.5.9.1.1 Conformance requirement

5.5.9.1.1.1 Normal execution

* CRRN1: The EnvelopeHandler is a Temporary JCRE Entry Point Object (see Javacard 2.2.1 Runtime Environment (JCRE) Specification [2]).
* CRRN2: The EnvelopeResponseHandler is a Temporary JCRE Entry Point Object (see Javacard 2.2.1 Runtime Environment (JCRE) Specification [2]).
* CRRN3: The ProactiveHandler is a Temporary JCRE Entry Point Object (see Javacard 2.2.1 Runtime Environment (JCRE) Specification [2]).
* CRRN4: The ProactiveResponseHandler is a Temporary JCRE Entry Point Object (see Javacard 2.2.1 Runtime Environment (JCRE) Specification [2]).

5.5.9.1.1.2 Parameter errors

No requirements.

5.5.9.1.1.3 Context errors

No requirements.

##### 5.5.9.1.2 Test area files

Test Source: Test\_Cre\_Api\_Hepo.java.

Test Applet: Cre\_Api\_Hepo\_1.java.

Cap File: Cre\_api\_hepo.cap.

##### 5.5.9.1.3 Test coverage

|  |  |
| --- | --- |
| CRR number | Test case number |
| CRRN1 | 1, 2 |
| CRRN2 | 3, 4 |
| CRRN3 | 5, 6 |
| CRRN4 | 7, 8 |

##### 5.5.9.1.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 1 | EnvelopeHandlerSystem.getTheHandler()  Store it in a static field of the toolkit applet | SecurityException is thrown |  |
| 2 | EnvelopeHandlerSystem.getTheHandler()  Store it in a field of the toolkit applet | SecurityException is thrown |  |
| 3 | EnvelopeResponseHandlerSystem.getTheHandler()  Store it in a static field of the toolkit applet | SecurityException is thrown |  |
| 4 | EnvelopeResponseHandlerSystem.getTheHandler()  Store it in a field of the toolkit applet | SecurityException is thrown |  |
| 5 | ProactiveHandlerSystem.getTheHandler()  Store it in a static field of the toolkit applet | SecurityException is thrown |  |
| 6 | ProactiveHandlerSystem.getTheHandler()  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 |  | Proactive command is fetched and terminal response is issued |
|  | ProactiveResponseHandlerSystem.getTheHandler()  Store it in a static field of the toolkit applet | SecurityException is thrown |  |
| 8 | ProactiveResponseHandlerSystem.getTheHandler()  Store it in a field of the toolkit applet | SecurityException is thrown |  |

#### 5.5.9.2 Transaction

Test Area Reference: Cre\_Api\_Tran.

##### 5.5.9.2.1 Conformance requirement

5.5.9.2.1.1 Normal execution

* CRRN1: A pending toolkit applet transaction at the ProactiveHandler.send() method invocation is aborted.
* CRRN2: A pending toolkit applet transaction is aborted on the termination of the toolkit applet (return from the *processToolkit()* method).
* CRRN3: At the invocation of the *processToolkit()* method there shall be no transaction in progress.

5.5.9.2.1.2 Parameter errors

No requirements.

5.5.9.2.1.3 Context errors

No requirements.

##### 5.5.9.2.2 Test area files

Test Source: Test\_Cre\_Api\_Tran.java.

Test Applet: Cre\_Api\_Tran\_1.java.

Cre\_Api\_Tran\_2.java.

Cap File: Cre\_api\_tran.cap.

##### 5.5.9.2.3 Test coverage

|  |  |
| --- | --- |
| CRR number | Test case number |
| CRRN1 | 1 |
| CRRN2 | 2 |
| CRRN3 | Not testable |

##### 5.5.9.2.4 Test procedure

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Description | API/CAT RE Expectation | APDU Expectation |
| 1 | Verify that transaction is aborted when a proactive command is sent  1- Applet1 is triggered and performed the following sequence  - Initialize a byte field with 0x05  - Build a display text proactive command  - beginTransaction()  - Update the byte with 0x02  - send the proactive command  3- Applet is resumed  - Verify that the byte value is 0x05  - JCSystem.getTransactionDepth() | JCSystem.getTransactionDepth()shall return 0 | 2- Proactive command fetched and terminal response is issued |
| 2 | Verify that transaction is aborted when a proactive command is sent  1- Applet2 is triggered and send a display text proactive command  Applet1 is triggered and performed the following sequence  - Initialize a static byte field with 0x05  - beginTransaction()  - Update the byte with 0x02  - Finalize  3- Applet2 is resumed and  - Verify that the byte value is 0x05  - JCSystem.getTransactionDepth() | JCSystem.getTransactionDepth()shall return 0 | 2- SW = 91 XX  Proactive command fetched and terminal response is issued |

#### 5.5.9.3 Timer Id between Applets

Test Area Reference: Cre\_Api\_Tmid.

##### 5.5.9.3.1 Conformance requirement

5.5.9.3.1.1 Normal execution

No requirements.

5.5.9.3.1.2 Parameter errors

No requirements.

5.5.9.3.1.3 Context errors

* CRRC1: The method ToolkitRegistry.releaseTimer() shall throw a ToolkitException with INVALID\_TIMER\_ID reason if the timer is valid but is not allocated to this applet.

##### 5.5.9.3.2 Test area files

Test Source: Test\_Cre\_Api\_Tmid.java.

Test Applet: Cre\_Api\_Tmid\_1.java.

Cap File: Cre\_api\_tmid.cap.

##### 5.5.9.3.3 Test coverage

|  |  |
| --- | --- |
| CRR number | Test case number |
| CRRC1 | 1 |

##### 5.5.9.3.4 Test procedure

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Description | API/CAT RE 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.  Trigger second instance and try to call releaseTimer() method with the static field value. | releaseTimer() method call shall throw a ToolkitException with INVALID\_TIMER\_ID reason |  |

### 5.5.10 Registration

#### 5.5.10.1 Event registration

Test Area Reference: Cre\_Reg\_Evtr.

##### 5.5.10.1.1 Conformance requirement

5.5.10.1.1.1 Normal execution

* CRRN1: A Toolkit Applet can change its registration to toolkit events during its whole life cycle.
* CRRN2: The registration of a Toolkit Applet to an event shall not be affected by its life cycle state.
* CRRN3: The *getShareableInterfaceObject()* has to be called before the applet is triggered the first time.
* CRRN4: The byte parameter of *getShareableInterfaceObject()* method has to be set to '01'.
* CRRN5: The AID parameter of the *getShareableInterfaceObject()* method shall be set to null.

5.5.10.1.1.2 Parameter errors

No requirements.

5.5.10.1.1.3 Context errors

No requirements.

##### 5.5.10.1.2 Test area files

Test Source: Test\_Cre\_Reg\_Evtr.java.

Test Applet: Cre\_Reg\_Evtr\_1.java.

Cre\_Reg\_Evtr\_2.java.

Cap File: Cre\_reg\_evtr.cap.

##### 5.5.10.1.3 Test coverage

|  |  |
| --- | --- |
| CRR number | Test case number |
| CRRN1 | 1 but partially tested |
| CRRN2 | 1,  and see also  CRRN9 of Cre\_Pcs\_Spco  and  CRRN3 of Cre\_Apt\_Eccn |
| CRRN3 | 2 |
| CRRN4 | 2 |
| CRRN5 | 2 |

##### 5.5.10.1.4 Test procedure

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Description | API/CAT RE Expectation | APDU Expectation |
| 1 | Toolkit applet can change its registration during its whole life cycle  1- Install Applet1 to let it in the INSTALL state  During its install() method, Applet1 registers to EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY and EVENT\_EVENT\_DOWNLOAD\_MT\_CALL  2- Make selectable Applet1  3- An envelope Event Download User Activity is sent to trigger Applet1  4- Applet1 registers to EVENT\_EVENT\_DOWNLOAD\_LOCATION\_STATUS and deregisters to EVENT\_EVENT\_DOWNLOAD\_USER\_ACTIVITY | 3- Applet1 is triggered | 2- Set Up Event List proactive command is fetched with User Activity and MT Call events  4- Set Up Event List proactive command is fetched with Location Status and MT Call events |
| 2 | *getShareableInterfaceObject()* has to be called before the first triggering  1- Install of Applet2  Applet2 *getShareableInterfaceObject()* method increments a counter  2- Trigger Applet2 | 2- Applet2 performs the following checks:  - the counter is incremented  - byte parameter of *getShareableInterfaceObject()* method is set to '01'  - AID parameter *getShareableInterfaceObject()* method is null |  |

### 5.5.11 UICC Toolkit Applet

#### 5.5.11.1 Data and function sharing

Test Area Reference: Cre\_Uta\_Dafs.

5.5.11.1.1 Conformance requirement

The sharing mechanism defined in "Java Card™ 2.2.1 Application Programming Interface Specification" ([1]) and "Java Card™ 2.2.1 Runtime Environment Specification" ([2]) shall be used by the Toolkit Applet(s) to share data and function.

5.5.11.1.1.1 Normal execution

* CRRN1:The interface shall extend the javacard.framework.shareable interface.
* CRRN2: The server Applet shall overwrite the Applet.getShareableInterfaceObject() method.
* CRRN3: The client Applet shall use the JCSystem.getAppletShareableInterfaceObject() to retrieve a reference to the server Applet shareable interface.
* CRRN4: When the client Applet calls JCSystem.getAppletShareableInterfaceObject() method the Applet.getShareableInterfaceObject() method of the server Applet is called by the CAT RE.

5.5.11.1.1.2 Parameter errors

No requirements.

5.5.11.1.1.3 Context errors

No requirements.

5.5.11.1.2 Test area files

This clause uses 2 packages:

uicc.test.catre.cre\_uta\_dafs.cre\_uta\_dafs\_a

uicc.test.catre.cre\_uta\_dafs.cre\_uta\_dafs\_b

Test Source: Test\_Cre\_Uta\_Dafs.java.

Test Applet: Cre\_Uta\_Dafs\_A\_1.java (server applet).

Cre\_Uta\_Dafs\_A\_2.java (server interface).

Cre\_Uta\_Dafs\_A\_3.java (class).

Cre\_Uta\_Dafs\_B\_1.java (client applet).

Cap File: Cre\_uta\_dafs\_a.cap.

Cre\_uta\_dafs\_b.cap.

##### 5.5.11.1.3 Test coverage

|  |  |
| --- | --- |
| CRR number | Test case number |
| CRRN1 | 1 |
| CRRN2 | 1 |
| CRRN3 | 1 |
| CRRN4 | 1 |

##### 5.5.11.1.4 Test procedure

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Description | API/CAT RE Expectation | APDU Expectation |
| 0 | 1- Install packages PackageA then PackageB  2- Install AppletA1 and AppletB1 |  |  |
| 1 | Use of a shareable interface  1- Send an envelope Menu Selection to trigger AppletA1 (Menu Id = 01)  2- AppletA1 stores the menu Id  3- Send an envelope Menu Selection to trigger AppletB1  4- AppletB1 uses the shareable interface of AppletA1 to retrieve the Menu Id that was used to trigger AppletA1 previously | 1- AppletA1 is triggered  3- AppletB1 is triggered  4- Menu Id retrieved shall be 01 |  |

#### 5.5.11.2 Package deletion

Test Area Reference: Cre\_Uta\_Pdel.

##### 5.5.11.2.1 Conformance requirement

The Package deletion mechanism defined in "Java Card™ 2.2.1 Runtime Environment Specification" ([2]) shall be used to delete the content from the UICC.

5.5.11.2.1.1 Normal execution

* CRRN1: If the applet/library package is resident in mutable memory, then the Java Card RE shall delete the applet/library package.
* CRRN2: Following a successful applet/library package deletion, it shall not be possible to install another package which depends on the deleted package.

5.5.11.2.1.2 Parameter errors

No requirements.

5.5.11.2.1.3 Context errors

* CRRC1: The deletion shall fail if a reachable (non-garbage) instance of a class belonging to the package being deleted exists on the card.
* CRRC2: The deletion shall fail if another package on the card depends on this package (as expressed in the CAP file's import component).

##### 5.5.11.2.2 Test area files

This clause uses 2 packages:

uicc.test.catre.cre\_uta\_pdel.cre\_uta\_pdel\_a

uicc.test.catre.cre\_uta\_pdel.cre\_uta\_pdel\_b (depends on uicc.test.catre.cre\_uta\_pdel.cre\_uta\_pdel\_a)

Test Source: Test\_Cre\_Uta\_Pdel.java.

Test Applet: Cre\_Uta\_Pdel\_A\_1.java.

Cre\_Uta\_Pdel\_A\_2.java.

Cre\_Uta\_Pdel\_A\_3.java (server interface) Cre\_Uta\_Pdel\_B\_1.java (use class Cre\_Uta\_Pdel\_A\_2).

Cre\_Uta\_Pdel\_B\_2.java.

Cap File: Cre\_uta\_pdel\_a.cap.

Cre\_uta\_pdel\_b.cap.

##### 5.5.11.2.3 Test coverage

|  |  |
| --- | --- |
| CRR number | Test case number |
| CRRN1 | 1 to 5 |
| CRRN2 | 3 |
| CRRC1 | 1 |
| CRRC2 | 2 |

##### 5.5.11.2.4 Test procedure

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Description | API/CAT RE Expectation | APDU Expectation |
| 0 | 1- Install packages PackageA then PackageB  2- Install AppletB1 |  |  |
| 1 | The package deletion is unsuccessful if a reachable instance of a class belonging to the package exists on the card  1- Delete PackageB  2- Install AppletB2  3- Delete AppletB1 and AppletB2 |  | 1- SW shall be different from 90 00  2- RAPDU = 00 90 00 |
| 2 | The package deletion is unsuccessful if another package on the card depends on this package  1- Delete PackageA  2- Install AppletA1  3- Delete AppletA1 |  | 1- SW shall be different from 90 0  02- RAPDU = 00 90 00 |
| 3 | The installation of a package which depends on a deleted package shall fail  1- Delete PackageB  2- Delete PackageA  3- Install PackageB |  | 1- SW = 90 00  2- SW = 90 00  3- SW shall be different from 90 00 |
| 4 | Once a package is deleted, it shall not be possible to install an applet from this package  1- Install AppletA1  2- Install AppletB2 |  | 1- SW = SW shall be different from 90 00  2- SW = SW shall be different from 90 00 |
| 5 | This test checks that it is possible to re-install the same package  1- Install PackageA  2- Install AppletA1 |  | 2- RAPDU = 00 90 00 |

#### 5.5.11.3 Applet deletion

Test Area Reference: Cre\_Uta\_Adel.

##### 5.5.11.3.1 Conformance requirement

The Applet deletion mechanism defined in "Java Card™ 2.2.1 Runtime Environment Specification" ([2]) shall be used to delete the content from the UICC.

5.5.11.3.1.1 Normal execution

* CRRN1: Following a successful applet instance deletion, the Java Card RE shall delete the applet instance.
* CRRN2: Following an unsuccessful applet instance deletion, the applet instance shall be selectable, and all objects owned by the applet shall remain unchanged.
* CRRN3: Following a successful applet instance deletion, it shall not be possible to select that applet, and no object owned by the applet can be accessed by any applet currently on the card or by a new applet created in the future.

5.5.11.3.1.2 Parameter errors

No requirements.

5.5.11.3.1.3 Context errors

* CRRC1: The deletion shall fail if any object owned by the applet instance is referenced from an object owned by another applet instance on the card.
* CRRC2: The deletion shall fail if any object owned by the applet instance is referenced from a static field on any package on the card.
* CRRC3: The deletion shall fail if an applet instance, belonging to the context of the applet instance being deleted, is active (selected) on the card.

##### 5.5.11.3.2 Test area files

This clause uses 2 packages:

uicc.test.catre.cre\_uta\_adel.cre\_uta\_adel\_a

uicc.test.catre.cre\_uta\_adel.cre\_uta\_adel\_b

Test Source: Test\_Cre\_Uta\_Adel.java.

Test Applet: Cre\_Uta\_Adel\_A\_1.java.

Cre\_Uta\_Adel\_A\_2.java.

Cre\_Uta\_Adel\_A\_3.java.

Cre\_Uta\_Adel\_B\_1.java.

Cre\_Uta\_Adel\_B\_2. java.

Cap File: Cre\_uta\_adel\_a.cap.

Cre\_uta\_adel\_b.cap.

##### 5.5.11.3.3 Test coverage

|  |  |
| --- | --- |
| CRR number | Test case number |
| CRRN1 | 3, 4 |
| CRRN2 | 1, 2, 3 |
| CRRN3 | 4, 5 |
| CRRC1 | 1 |
| CRRC2 | 2 |
| CRRC3 | 3 |

##### 5.5.11.2.4 Test procedure

| Id | Description | API/CAT RE Expectation | APDU Expectation |
| --- | --- | --- | --- |
| 0 | 1- Install packages PackageA then PackageB  2- Install AppletA1, AppletA2, AppletB1 and AppletB2 |  |  |
| 1 | The deletion shall fail if any object owned by the applet instance is referenced from an object owned by another applet instance on the card  1- Send an envelope Menu Selection to trigger AppletA1  2- AppletA1 store the menu Id  3- Send an envelope Menu Selection to trigger AppletA2  4- AppletA2 gets a reference to the shareable interface to retrieve the menu Id used to trigger AppletA1  5- Delete AppletA1  6- Send an envelope Menu Selection to trigger AppletA1  7- Send an envelope Menu Selection to trigger AppletA2  8- AppletA2 frees the reference to the shareable interface  9- Delete AppletA1  10- Install AppletA1  11- Send an envelope Menu Selection to trigger AppletA1  12- AppletA1 store the menu Id  13- Send an envelope Menu Selection to trigger AppletB2  14- AppletB2 gets a reference to the shareable interface to retrieve the menu Id used to trigger AppletA1  15- Delete AppletA1  16- Send an envelope Menu Selection to trigger AppletA1  17- Send an envelope Menu Selection to trigger AppletB2  18- AppletB2 frees reference to the shareable interface  19- Delete AppletA1  20- Install AppletA1 | 1- AppletA1 is triggered  3- AppletA2 is triggered  6- AppletA1 is triggered  7- AppletA2 is triggered  11- AppletA1 is triggered  13- AppletB2 is triggered  16- AppletA1 is triggered  17- AppletB2 is triggered | 5- SW shall be different from 90 00  7- SW = 90 00  9- SW = 90 00  10- RAPDU = 00 90 00  15- SW shall be different from 90 00  17- SW = 90 00  19- SW = 90 00  20- RAPDU = 00 90 00 |
| 2 | The deletion shall failed if any object owned by the applet instance is referenced from a static field on any package on the card  1- Send an envelope Menu Selection to trigger AppletA1  2- AppletA1 store the menu Id  3- Send an envelope Menu Selection to trigger AppletA2  4- AppletA2 gets a reference to the shareable interface and stores it in a static field of PackageA, then retrieves the menu Id used to trigger AppletA1  5- Delete AppletA2  6- Delete AppletA1  7- Send an envelope Menu Selection to trigger AppletA1  8- AppletA1 frees the reference to the shareable interface  9- Delete AppletA1  10- Install AppletA1  11- Send an envelope Menu Selection to trigger AppletA1  12- AppletA1 store the menu Id  13- Send an envelope Menu Selection to trigger AppletB2  14- AppletB2 gets a reference to the shareable interface and stores it in a static field of PackageB, then retrieves the menu Id used to trigger AppletA1  15- Delete AppletB2  16- Delete AppletA1  17- Send an envelope Menu Selection to trigger AppletA1  18- Send an envelope Menu Selection to trigger AppletB1  19- AppletB1 then frees the reference to the shareable interface  20- Delete AppletA1  21- Install AppletA1 | 1- AppletA1 is triggered  3- AppletA2 is triggered  7- AppletA1 is triggered  11- AppletA1 is triggered  13- AppletB2 is triggered  17- AppletA1 is triggered  18- AppletB1 is triggered | 5- SW = 90 00  6- SW shall be different from 90 00  9- SW = 90 00  10- RAPDU = 00 90 00  15- SW = 90 00  16- SW shall be different from 90 00  20- SW = 90 00  21- RAPDU = 00 90 00 |
| 3 | Deletion of an active applet instance  1- Delete AppletB1  2- Install AppletB1  3- Open another channel and select AppletB1 the new open channel  4- Delete AppletB1 on channel 0  5- Select AppletB1 on Channel 0  6- Reset |  | 1- SW = 90 00  2- RAPDU = 00 90 00  3- RAPDU = <applet selected data> 90 00  4- SW shall be different from 90 00  5- RAPDU = <applet selected data> 90 00 |
| 4 | Selection of a deleted applet instance  1- Delete AppletB1  2- Select AppletB1 on Channel 0  3- Install AppletB1 |  | 1- SW = 90 00  2- SW shall be different from 90 00  3- RAPDU = 00 90 00 |
| 5 | Object owned by a deleted applet can't be accessed by other applets  1- Delete AppletA1  2- Send an envelope Menu Selection to trigger AppletB1  3- AppletB1 gets a reference to the shareable interface to retrieve the menu Id used to trigger AppletA1 | 2- AppletB1 is triggered  3- An exception is thrown | 1- SW = 90 00 |

#### 5.5.11.4 Object deletion

Test Area Reference: Cre\_Uta\_Odel.

##### 5.5.11.4.1 Conformance requirement

5.5.11.4.1.1 Normal execution

* CRRN1: If an object deletion mechanism is supported then the one defined in "Java Card™ 2.2.1 Application Programming Interface Specification" ([1]) shall be used.

5.5.11.4.1.2 Parameter errors

No requirements.

5.5.11.4.1.3 Context errors

No requirements.

##### 5.5.11.4.2 Test area files

Test Script: Test\_Cre\_Uta\_Odel.java

Test Applet: Cre\_Uta\_Odel\_1.java

Cap File: Cre\_uta\_odel.cap

##### 5.5.11.4.3 Test coverage

|  |  |
| --- | --- |
| CRR number | Test case number |
| CRRN1 | 1 |

##### 5.5.11.4.4 Test procedure

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Description | API/CAT RE Expectation | APDU Expectation |
| 1 | Object deletion  1- Applet calls JCSystem.isObjectDeletionSupported() | 2- Returns TRUE |  |

### 5.5.12 Proactive Command Handling

#### 5.5.12.1 General behaviour

Test Area Reference: Cre\_Pch\_Genb.

##### 5.5.12.1.1 Conformance requirement

5.5.12.1.1.1 Normal execution

* CRRN1: On the call to the *send()* method, the CAT Runtime Environment shall handle the transmission of the proactive command to the terminal, and the reception of the response.
* CRRN2: The CAT Runtime Environment shall resume the Toolkit Applet execution on the return from the *send()* method.

5.5.12.1.1.2 Parameter errors

No requirements.

5.5.12.1.1.3 Context errors

No requirements.

##### 5.5.12.1.2 Test area files

None.

##### 5.5.12.1.3 Test coverage

|  |  |
| --- | --- |
| CRR number | Test case number |
| CRRN1 | See test case 2 of Cre\_Hin\_Prhd |

##### 5.5.12.1.4 Test procedure

None.

### 5.5.13 CAT Runtime Environment behaviour

#### 5.5.13.1 Context

Test Area Reference: Cre\_Cat\_Ctxt.

##### 5.5.13.1.1 Conformance requirement

5.5.13.1.1.1 Normal execution

* CRRN1: At the invocation of the *processToolkit()* method the context as defined in Java Card shall be set to the context of the Toolkit Applet. The previous context (context of the caller) shall be the context of the CAT Runtime Environment.
* CRRN2: During the execution there might be other context switches, but at the return of the *send()* method the toolkit applet context is restored.

5.5.13.1.1.2 Parameter errors

No requirements.

5.5.13.1.1.3 Context errors

No requirements.

##### 5.5.13.1.2 Test area files

Test Source: Test\_Cre\_Cat\_Ctxt.java.

Test Applet: Cre\_Cat\_Ctxt\_1.java.

Cap File: Cre\_cat\_ctxt.cap.

##### 5.5.13.1.3 Test coverage

|  |  |
| --- | --- |
| CRR number | Test case number |
| CRRN1 | Not Testable |
| CRRN2 | see Cre\_Hin\_Enhd |

##### 5.5.13.1.4 Test procedure

Not applicable.

### 5.5.14 UICC and ADF File System Administration API

#### 5.5.14.1 AdminFile View

Test Area Reference: Cre\_Fsa\_View

##### 5.5.14.1.1 Conformance requirement

5.5.14.1.1.1 Normal execution

* CRRN1: *AdminFileView* objects follow the behaviour of *FileView* objects and inherit *FileView* functionality.
* CRRN2: An *AdminFileView* object can be retrieved by invoking one of the *getAdminFileView()* methods defined in the AdminFileViewBuilder class.
* CRRN3: Each *AdminFileView* shall be given the access control privileges associated with the UICC or the corresponding ADF for the Applet.
* CRRN4: The access control privileges are checked against the access rules defined in ETSI TS 102 221 [5] each time a method of the *AdminFileView* object is invoked.

5.5.14.1.1.2 Parameter errors

No requirements.

5.5.14.1.1.3 Context errors

No requirements.

##### 5.5.14.1.2 Test area files

None.

##### 5.5.14.1.3 Test coverage

| CRR number | Test case number |
| --- | --- |
| CRRN1 | See API\_4\_Afv\_xxxx |
| CRRN2 | See API\_4\_Afb\_xxxx |
| CRRN3 | See Cre\_Tin\_Acdo |
| CRRN4 | See Cre\_Tin\_Acdo |

##### 5.5.14.1.4 Test procedure

None.

#### 5.5.14.2 AdminFile Access

Test Area Reference: Cre\_Fsa\_Aacc.

See API access.administration part.