ETSI TS 134 131 V7.0.0 (2007-06)

Technical Specification

**Digital cellular telecommunications system (Phase 2+);
Universal Mobile Telecommunications System (UMTS);
Test Specification for C-language binding
to (Universal) Subscriber Interface Module ((U)SIM)
Application Programming Interface (API)
(3GPP TS 34.131 version 7.0.0 Release 7)**

Reference
RTS/TSGC-0634131v700

Keywords
GSM, UMTS

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

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

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

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

The present document may refer to technical specifications or reports using their 3GPP identities, UMTS identities or GSM identities. These should be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between GSM, UMTS, 3GPP and ETSI identities can be found under http://webapp.etsi.org/key/queryform.asp.

Contents

Foreword

This Technical Specification has been produced by the 3GPP.

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

Version x.y.z

where:

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

1 Scope

The present document describes the technical characteristics and methods of test for testing the SIM API for the C programming language 3GPP TS 31.131 [11] implemented in the subscriber identity modules for GSM and 3G networks. It specifies the following parts:

- test applicability
- test environment description
- tests format
- test area reference
- conformance requirements
- test suite files
- test procedure
- test coverage and,
- a description of the associated testing tools that may be used.

2 References

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

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

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

[3] 3GPP TS 23.048: "3rd Generation Partnership Project; Technical Specification Group Terminals; Security Mechanisms for the SIM application toolkit".

[4] 3GPP TS 42.019: "3rd Generation Partnership Project; Technical Specification Group Terminals; Subscriber Identity Module Application Programming Interface (SIM API); Stage 1".

[5] ISO 639 (1988): "Code for the representation of names of languages".

[6] 3GPP TS 23.038: "Alphabets and language-specific information".

[7] ISO/IEC 9899 Second Edition 1999-12-01: "Programming Languages – C".

[8] 3GPP TS 11.14: "Specification of the SIM Application Toolkit for the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface" version 4.0.0 Release 4.

[9] Tool Interface Standard (TIS) Executable and Linking Format Specification Version 1.2

[10] SYSTEM V Application Binary Interface, Edition 4.1

[11] 3GPP TS 31.131 V1.1.2: "'C'-language binding to (U)SIM API".

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

[13] 3GPP TS 51.011: "Specification of the Subscriber Identity Module – Mobile Equipment (SIM-ME) interface".

[14] ETSI TS 102.221: "UICC-Terminal interface; Physical and logical characteristics".

[15] ETSI TS 102.226: "Remote APDU Structure for UICC based Applications."

3 Definitions and Acronyms

3.1 Definitions

The definitions specified in GSM 11.10-1 [12] shall apply, unless otherwise specified in the present clause.

Application: A computer program that defines and implements a useful domain-specific functionality. The term may apply to the functionality itself, to the representation of the functionality in a programming language, or to the realization of the functionality as executable code.

Application Executable: The representation of an application as collection of executable codes.

Application Program: The representation of an application in a programming language such as assembly language, C, Java, WML or XHTML.

Application Programming Interface: A collection of entry points and data structures that an application program can access when translated into an application executable.

Byte Code: A processor-independent representation of a basic computer operation such as "increment by one" that is executed by computer program called a byte code interpreter.

Data Structure: A memory address that can be accessed by an application executable in order to read or write data.

Entry Point: A memory address that can be branched to by an application executable in order to access functionality defined by an application-programming interface. Depending on the software technology, an entry point is also called a subroutine, a function or a method.

Executable Code: The generic term for either byte code or native code.

Framework : A framework defines a set of Application Programming Interface (API) functions for developing applications and for providing system services to those applications.

Native Code: A processor-dependent representation of a basic computer operation such as "increment by one" that is executed by the hardware circuitry of a computer's central processing unit.

Null Operation: A computer operation that accomplishes nothing. Abbreviated as NOP and pronounced "No Op".

Toolkit Application: An application that uses the commands described in [2].

3.2 Acronyms

For the purpose of the present document, the following abbreviations apply:

AC	Application Code
AID	Application Identifier
APDU	Application Protocol Data Unit
API	Application Programming Interface
CAD	Card Acceptance Device
CPDU	Command Protocol Data Unit
CAT	Card Application Toolkit
DF	Dedicated File

DTMF	Dual Tone Multiple Frequency
EF	Elementary File
FID	File Identifier
GSM	Global System for Mobile communications
IFD	Interface Device
NOP	Null OPeration
ME	Mobile Equipment
NVM	Non-Volatile Memory
ROM	Read-Only Memory
SE	Sending Entity
SIM	Subscriber Identity Module
SMS	Short Message Service
STK	SIM ToolKit
TLV	Tag, Length, Value
TPDU	Transport Protocol Data Unit
UICC	(not an acronym)
URL	Uniform Resource Locator
USIM	Universal Subscriber Interface Module

4 Test Environment

This clause specifies requirements that shall be met and the testing rules that shall be followed during the test procedure.

4.1 Applicability

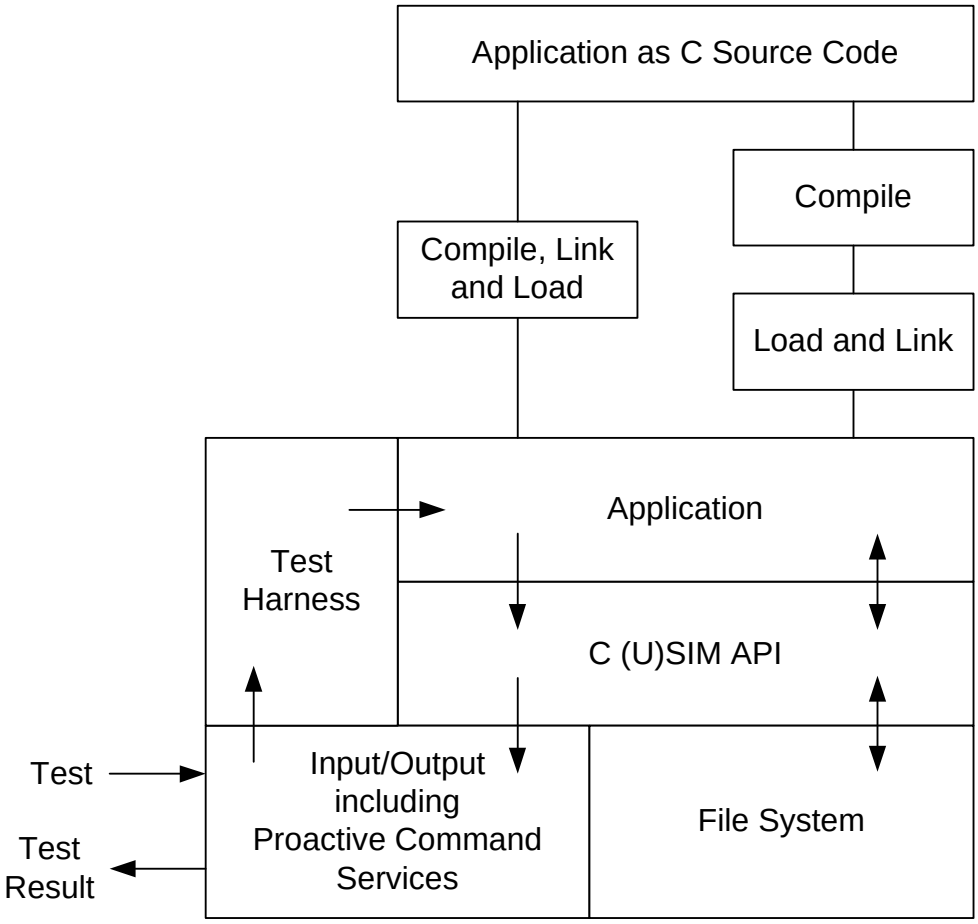
The tests defined in this specification shall be performed taking into account the services supported by the card as specified in the EF_{SST} file.

The tests can be performed with either C source code files (.c) or with ELF loadfiles (.elf).

4.2 Test Environment Description

The C SIM API test specification covers applications that are linked into the smart card mask and stored in read-only memory as well as applications that are loaded into read/write memory after it has been manufactured. The C SIM API test specification also covers applications whose executable form is either as byte codes or as native codes.

The general architecture for the test environment is given in the following diagram:



4.3 Test Format

4.3.1 Test Area Reference

Each test area is referenced as follows:

API: API Testing: 'API_[entry point name]' where

entry point name:

API entry point defined in 3GPP TS 31.131 [11]

4.3.1.1 Conformance Requirements

The conformance requirements are expressed in the following way:

- Entry point prototype as listed in 3GPP TS 31.131 [11] specification.
- Normal execution:
  - Contains normal execution and correct parameters limit values, each referenced as a Conformance Requirement Reference Normal (CRRN)
- Parameters error:
  - Contains parameter errors and incorrect parameter limit values, each referenced as a Conformance Requirement Reference Parameter Error (CRRP)
- Context error:

- Contains errors due to the context the entry point is used in, each referenced as a Conformance Requirement Reference Context Error (CRRC)

4.3.1.2 Test Area Files

The test area contains the source code of a program that translates the test procedure found in each section below into C test application program (.c) and an APDU Tool test script (.s). A test is conducted by compiling the C program, installing it on the UICC, and running the associated test script.

The test area also contains the C header files that are referenced by the output of the translator. The test scripts must be processed through a C pre-processor [7] before they are provided to the APDU Tool.

4.3.1.3 Test Procedure

Each test procedure contains pseudo-code that describes the test together with the expected responses from the API and the expected command and response APDU traffic. The test procedure may be translated into a C test application program and APDU Tool test script using the program in the Test Area

4.3.1.4 Test Coverage

The table at the end of each test procedure indicates the correspondence between the Conformance Requirements Reference (CRR) and the different test cases.

4.4 Initial Conditions

The Initial Conditions are a set of general prerequisites for the (U)SIM prior to the execution of testing. For each test procedure described in this document, the following rules apply to the Initial Conditions:

- unless otherwise stated, the file system and the files' content shall fulfill the requirements described in the "Default Prepersonalisation" paragraph at the time of running the test with test number 1;
- unless otherwise stated, tests with test number 1 through i have been successfully executed before the test with test number i+1 is executed.

When both statements apply, a test procedure is said to be in the "Default Initial Conditions" state.

4.5 Test Equipment

These subclauses recommend a minimum specification for each of the items of test equipment referenced in the tests.

4.5.1 APDU Tool

This test tool shall meet the following requirements:

- be able to send commands to the card TPDU;
- be able to check none, only a part, or all of the data returned;
- be able to check none, only part, or all of the status returned;
- be able to accept all valid status codes returned;
- be able to support CAD commands;
- be able to generate a log file for each test execution.
- if more data is returned than defined in the test specification the event shall be noted in the log and the tool shall continue;
- if less data is returned than defined in the test specification the event shall be noted in the log and the tool shall continue;

- if there is an error in data or status returned the event shall be noted in the log and the tool shall continue.

The log file produced by the test tool shall include the following information:

- all commands issued;
- all data returned;
- all status returned;
- all errors codes;
- expected data and status in case of error;
- comments from the scripts;

a log message to report success or failure of the test;

transmission error events including anomalies in returned data;

4.6 Testing Methodology

4.6.1 Test Interfaces and Facilities

The SIM-ME interface provides the main transport interface for the purpose of performing conformance tests.

The SIM API interface provides the main test interface for the purpose of performing conformance tests.

5 Test Plan

The test plan is divided according to the entry point sections of 3GPP TS 31.131 [11] although the sections of the test plan do not appear in the same order as the corresponding sections in TS 31.131 [11].

6 API Test Plan

6.1 UICC File Store Access

6.1.1 CatSelect

Test Area Reference: API_CatSelect

6.1.1.1 Conformance Requirements

The entry point with the following signature shall be compliant to its definition in the API.

```
WORD CatSelect(CatFID FileIdentifier, CatFileStatus *Status)
```

6.1.1.1.1 Normal execution

CRRN1: If the desired file is selected, the proper status information has been returned in *Status.

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

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

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

CRRN5: The file with a file identifier that matches fid shall be found according to the following selection rules:

1) An immediate child EF or DF of the current MF/DF can be selected,

2) A sibling DF of the current DF can be selected,

3) The current MF/DF it self can be selected,

4) The parent MF/DF of the current DF can be selected,

The MF can always be selected.

CRRN6: If status is NULL the invocation is a NOP.

6.1.1.1.2 Parameter errors

6.1.1.1.3 Context errors

CRRC1: If the file with a file identitifier which matches FileIdentitifer could not be found according to the selection rules listed in CRRN5 then the FILE_NOT_FOUND status word shall be returned.

CRRC2: If the entry point call causes a memory problem (e.g. memory access error), the MEMORY_PROBLEM status word shall be returned.

CRRC3: If the entry point call causes an error to occur that is not expected and thus not handled, the INTERNAL_ERROR status word shall be returned.

6.1.1.2 Test Procedure

```
SET Test TO CatSelect.

/* Test Case 1: Select MF in MF (Transparent EF) */
IF   {CatSelect(FID_MF, &FCP) == SW_OK}
THEN {FCP.fileType == FT_MF}.

/* Test Case 2: Select EF-ICCID in MF (Transparent EF) */
IF   {CatSelect(FID_EF_ICCID, &FCP) == SW_OK}
THEN {FCP.fileType == FT_EF}.

/* Test Case 3: Select DF-GSM in MF*/
IF   {CatSelect(FID_DF_GSM, &FCP) == SW_OK}
THEN {FCP.fileType == FT_DF}.

/* Test Case 4: Select EF-ACM in DF-GSM (Cyclic EF) */
IF   {CatSelect(FID_EF_ACM, &FCP) == SW_OK}
THEN {FCP.fileType == FT_EF}.

/* Test Case 5: Select MF*/
IF   {CatSelect(FID_MF, &FCP) == SW_OK}
THEN {FCP.fileType == FT_MF}.

/* Test Case 6: Select DF-TELECOM in MF */
IF   {CatSelect(FID_DF_TELECOM, &FCP) == SW_OK}
THEN {FCP.fileType == FT_DF}.

/* Test Case 7: Select EF-FDN in DF-TELECOM (Linear Fixed EF) */
IF   {CatSelect(FID_EF_FDN, &FCP) == SW_OK}
THEN {FCP.fileType == FT_EF}.

/* Test Case 8: Status is null*/
IF   {CatSelect(FID_EF_FDN, NULL) == SW_OK}
THEN {FCP.fileType == FT_EF}.

/* Test Case 9: EF not selected after MF/DF selection */
IF   {CatSelect(FID_MF, &FCP) == SW_OK}
AND  {CatSelect(FID_EF_ICCID, &FCP) == SW_OK}
AND  {CatSelect(FID_MF, &FCP) == SW_OK}
THEN {CatReadBinary(0, &bytes, buffer) == SW_NO_CURRENT_FILE}
AND  {FCP.fileType == FT_UNKNOWN}.
```

```
/* Test Case 10: Selected file not available */
IF   {CatSelect(FID_MF, &FCP) == SW_OK}
THEN {CatSelect(FID_EF_ACM, &FCP) == SW_FILE_NOT_FOUND}
AND  {FCP.fileType == FT_UNKNOWN}.

/* Test Case 11: No record is selected after selecting linear fixed EF */
IF   {CatSelect(FID_MF, &FCP) == SW_OK}
AND  {CatSelect(FID_DF_SIMTEST, &FCP) == SW_OK}
AND  {CatSelect(FID_EF_LARU, &FCP) == SW_OK}
THEN {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_RECORD_NOT_FOUND}
AND  {FCP.fileType == FT_EF}.
```

6.1.1.3 Test Coverage

CRR Number	Test Case Number
N1	1,2, 3
N2	4
N3	5
N4	6
N5	7
N6	8
C1	9, 10, 11
C2, C3	Not Tested

6.1.2 CatStatus

Test Area Reference: API_CatStatus

6.1.2.1 Conformance Requirements

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

```
WORD CatStatus(CatFileStatus *Status)
```

6.1.2.1.1 Normal execution

CRRN1: The status information of the current EF is returned.

CRRN2: If Status is NULL the invocation is a NOP.

6.1.2.1.2 Parameter errors

6.1.2.1.3 Context errors

6.1.2.2 Test Procedure

```
SET Test TO CatStatus.

/* Test Case 1: Status of MF */
IF   {CatSelect(FID_MF, &FCP) == SW_OK}
AND  {CatStatus(&Status) == SW_OK}
THEN {memcmp(&FCP, &Status, sizeof(CatDFStatus)) == 0}
AND  {Status.fileType == FT_MF}.

/* Test Case 2: Status of EF-ICCID */
IF   {CatSelect(FID_EF_ICCID, &FCP) == SW_OK}
AND  {CatStatus(&Status) == SW_OK}
THEN {memcmp(&FCP, &Status, sizeof(CatEFStatus)) == 0}.

/* Test Case 3: Status of DF-TELECOM /
```

```
IF   {CatSelect(FID_DF_TELECOM, &FCP) == SW_OK}
AND  {CatStatus(&Status) == SW_OK}
THEN {memcmp(&FCP, &Status, sizeof(CatDFStatus)) == 0}.

/* Test Case 4: Status with NULL argument */
IF   {CatStatus(NULL) == SW_OK}
THEN {NOP}.
```

6.1.2.3 Test Coverage

CRR Number	Test Case Number
N1	1, 2, 3
N2	4

6.1.3 CatGetCHVStatus

Test Area Reference: API_CatGetCHVStatus

6.1.3.1 Conformance Requirements

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

```
WORD CatGetCHVStatus(BYTE *CHV)
```

6.1.3.1.1 Normal execution

CRRN1: The current CHV status information is returned.

6.1.3.1.2 Parameter errors

6.1.3.1.3 Context errors

6.1.3.2 Test Procedure

```
SET Test TO CatGetCHVStatus.

/* Test Case 1: Get the ATR CHV Status */
IF   {(CatGetCHVStatus(CHVStatus),SW_OK) == SW_OK}
THEN {CHVStatus[0] == 0}
AND  {CHVStatus[1] == 0}
AND  {CHVStatus[2] == 0}
AND  {CHVStatus[3] == 0}.
```

6.1.3.3 Test Coverage

CRR Number	Test Case Number
N1	1

6.1.4 CatReadBinary

Test Area Reference: API_CatReadBinary

6.1.4.1 Conformance Requirements

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

```
WORD CatReadBinary(DWORD Offset, DWORD *Bytes, void *Buffer);
```

6.1.4.1.1 Normal execution

CRRN1: If data can be accessed at the specified offset in the currently selected transparent file, an attempt is made to read the number of bytes in Bytes. The bytes read are returned in Buffer and the actual number of bytes read returned in Bytes.

CRRN2: If Bytes is non-null and Buffer is null, then the number of bytes that could be read is returned in Bytes.

CRRN3: If Bytes is null or *Bytes is zero the invocation is a NOP

6.1.4.1.2 Parameter errors

CRRP1: If Offset exceeds the length of the file, the status word OUT_OF_FILE_BOUNDARIES shall be returned.

6.1.4.1.3 Context errors

CRRC1: If there is no currently selected EF, the status word NO_EF_SELECTED shall be returned.

CRRC2: If the currently selected EF is not transparent, the status word FILE_INCONSISTENT shall be returned.

CRRC3: If the calling application does not fulfil the access condition, READ, to perform this function, the status word SECURITY_CONDITION_NOT_SATISFIED shall be returned.

CRRC4: If the currently selected EF is invalidated and the file status of the EF does not allow for the reading of an invalidated file, the status word INVALIDATION_STATUS_CONTRADICTION shall be returned.

CRRC5: If the entry point call causes a memory problem (e.g. memory access error), an instance the status word MEMORY_PROBLEM shall be returned.

CRRC6: If the entry point call causes an error to occur that is not expected and thus not handled, an instance of the status word INTERNAL_ERROR shall be returned.

6.1.4.2 Test Procedure

```
SET Test TO CatReadBinary
AND Data TO {0x0F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF}.

/* Test Case 1: Read from EF-ICCID in MF */
IF   {CatSelect(FID_EF_ICCID, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 5}
THEN {CatReadBinary(0, &bytes, buffer) == SW_OK}
AND  {bytes == 5}
AND  {memcmp(buffer, Data, bytes) == 0}.

/* Test Case 2: Offset < EOF < Offset+Bytes */
IF   {CatSelect(FID_EF_ICCID, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 5}
THEN {CatReadBinary(6, &bytes, buffer) == SW_OK}
AND  {bytes == 4}
AND  {memcmp(buffer, &Data[1], bytes) == 0}.

/* Test Case 3: EOF < Offset */
IF   {CatSelect(FID_EF_ICCID, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 5}
THEN {CatReadBinary(100, &bytes, buffer) == SW_OK}
AND  {bytes == 0}.

/* Test Case 4: Buffer is null */
IF   {CatSelect(FID_EF_ICCID, NULL) == SW_OK}
AND  {bytes = 5}
THEN {CatReadBinary(6, &bytes, NULL) == SW_OK}
AND  {bytes == 4}.

/* Test Case 5: EF is not Transparent */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
THEN {CatReadBinary(0, &bytes, NULL) == SW_INCOMPATIBLE_FILE_STRUCTURE}.
```

```
/* Test Case 6: Access condition not fulfilled */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_TNR, NULL) == SW_OK}
THEN {CatReadBinary(0, &bytes, NULL) == SW_SECURITY_STATUS_NOT_SATISFIED}.

/* Test Case 7: EF is invalidated */
IF   {CatSelect(FID_EF_TNU, NULL) == SW_OK}
AND  {(CatInvalidate(), SW_OK) == SW_OK}
THEN {CatReadBinary(0, &bytes, NULL) == SW_REFERENCED_DATA_INVALIDATED}.

/* Test Case 8: No EF selected */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
THEN {CatReadBinary(0, &bytes, NULL) == SW_NO_CURRENT_FILE}.

/* Test Case 9: NULL arguments */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_TARU, NULL) == SW_OK}
THEN {CatReadBinary(0, NULL, NULL) == SW_OK}.
```

6.1.4.3 Test Coverage

CRR Number	Test Case Number
N1	1,2
N2	4
N3	9
P1	3
C1	8
C2	5
C3	6
C4	7
C5, C6	Not tested

6.1.5 CatUpdateBinary

Test Area Reference: API_CatUpdateBinary

6.1.5.1 Conformance Requirements

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

```
UINT16 CatUpdateBinary(DWORD Offset, DWORD Bytes, void *Buffer);
```

6.1.5.1.1 Normal execution

CRRN1: The currently selected transparent file is updated starting at Offset with the sequence of Bytes bytes in Buffer.

6.1.5.1.2 Parameter errors

CRRP1: If Bytes is positive and Buffer is null, then INCORRECT_PARAMETERS shall be returned.

6.1.5.1.3 Context errors

CRRC1: If there is no currently selected EF, the status word NO_EF_SELECTED shall be returned.

CRRC2: If the currently selected EF is not transparent, the status word FILE_INCONSISTENT shall be returned.

CRRC3: If the calling application does not fulfil the access condition, READ, to perform this function, the status word SECURITY_CONDITION_NOT_SATISFIED shall be returned.

CRRC4: If the currently selected EF is invalidated and the file status of the EF does not allow for the reading of an invalidated file, the status word INVALIDATION_STATUS_CONTRADICTION shall be returned.

CRRC5: If the entry point call causes a memory problem (e.g. memory access error), an instance the status word MEMORY_PROBLEM shall be returned.

CRRC6: If the entry point call causes an error to occur that is not expected and thus not handled, an instance of the status word INTERNAL_ERROR shall be returned.

6.1.5.2 Test Procedure

```
SET Test TO CatUpdateBinary
AND Data TO {1,2,3,4,5}.

/* Test Case 1: Update Transparent EF */
IF   {CatSelect(FID_EF_ICCID, NULL) == SW_OK}
AND  {bytes = 5}
THEN {CatUpdateBinary(0, bytes, Data) == SW_OK}
AND  {CatReadBinary(0, &bytes, buffer) == SW_OK}
AND  {bytes == 5}
AND  {memcmp(Data, buffer, bytes) == 0}.

/* Test Case 2: Offset < EOF and Offset + Bytes > EOF*/
IF   {CatSelect(FID_EF_ICCID, NULL) == SW_OK}
AND  {bytes = 5}
THEN {CatUpdateBinary(7, bytes, Data) == SW_OK}
AND  {CatReadBinary(7, &bytes, buffer) == SW_OK}
AND  {bytes == 5}
AND  {memcmp(Data, buffer, bytes) == 0}.

/* Test Case 3: Offset > EOF */
IF   {CatSelect(FID_EF_ICCID, NULL) == SW_OK}
THEN {CatUpdateBinary(11, 5, Data) == SW_OK}
AND  {CatReadBinary(11, &bytes, buffer) == SW_OK}
AND  {bytes == 5}
AND  {memcmp(Data, buffer, bytes) == 0}.

/* Test Case 4: Buffer is NULL */
IF   {CatSelect(FID_EF_ICCID, NULL) == SW_OK}
THEN {CatUpdateBinary(0, 5, NULL) == SW_OK}.

/* Test Case 5: No EF selected */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
THEN {CatUpdateBinary(0, 5, Data) == SW_NO_CURRENT_FILE}.

/* Test Case 6: EF is not transparent */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
THEN {CatUpdateBinary(0, 5, Data) == SW_INCOMPATIBLE_FILE_STRUCTURE}.

/* Test Case 7: Access condition not fulfilled */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_TNU, NULL) == SW_OK}
THEN {CatUpdateBinary(0, 5, Data) == SW_SECURITY_STATUS_NOT_SATISFIED}.

/* Test Case 8: EF is invalidated*/
IF   {CatSelect(FID_EF_TNU, NULL) == SW_OK}
AND  {(CatInvalidate(), SW_OK) == SW_OK}
THEN {CatUpdateBinary(0, 5, Data) == SW_REFERENCED_DATA_INVALIDATED}.
```

6.1.5.3 Test Coverage

CRR Number	Test Case Number
N1	1,2,3
P1	4
C1	5
C2	6
C3	7
C4	8
C5, C6	Not Tested

6.1.6 CatReadRecord

Test Area Reference: API_CatReadRecord

6.1.6.1 Conformance Requirements

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

```
UINT16 CatReadRecord(DWORD RecordNumber, CatRecordAccessModes Mode,
                     DWORD Offset, DWORD *NumBytes, void *Buffer);
```

6.1.6.1.1 Normal execution

CRRN1: NumBytes bytes starting at Offset from the record specified by Mode and RecordNumber of the currently selected linear fixed or cyclic EF are read into Buffer.

CRRN2: If the access mode is ABSOLUTE or CURRENT:

- if RecordNumber is not 0, the record addressed by RecordNumber will be read;
- if RecordNumber is 0 the current selected record will be read; and
- the current record pointer shall not change.

CRRN3: If the access mode is NEXT:

- the next record relative to the current selected record will be selected and read;
- if no current record is selected, the first record will be selected and read;
- if the current record pointer is set to the last record for a cyclic EF the record pointer is set to the first record and the record is read;
- the current record pointer of any other Application shall not be changed.

CRRN4: If the access mode is PREVIOUS:

- the previous record relative to the current selected record will be selected and read;
- if no current record is selected, the last record will be selected and read;
- if the current record pointer is set to the first record, for a linear fixed EF the entry point responses with an error exception and for a cyclic EF the record pointer is set to the last record and the record is read;
- the current record pointer of any other Application shall not be changed.

6.1.6.1.2 Parameter errors

CRRP1: If the currently selected EF is linear fixed and the access mode is ABSOLUTE and RecordNumber is greater than records available, the status word RECORD_NUMBER_NOT_AVAILABLE shall be returned

CRRP2: If the currently selected EF is linear fixed and the access mode is CURRENT, RecordNumber is 0 and there is no current record selected, the status word RECORD_NUMBER_NOT_AVAILABLE shall be returned.

CRRP3: If the currently selected EF is linear fixed and the access mode is NEXT and the current record pointer is set to the last record, the status word RECORD_NUMBER_NOT_AVAILABLE shall be returned.

CRRP4: If the currently selected EF is linear fixed and the access mode is PREVIOUS and the current record pointer is set to the first record, the status word RECORD_NUMBER_NOT_AVAILABLE shall be returned.

6.1.6.1.3 Context errors

CRRC1: If the calling Application has currently no EF selected, the status word NO_EF_SELECTED shall be returned.

CRRC2: If the currently selected EF is neither linear fixed nor cyclic, the status word FILE_INCONSISTENT shall be returned.

CRRC3: If the calling Application does not fulfil the access condition, READ, to perform this function, the status word AC_NOT_FULFILLED shall be returned.

CRRC4: If the currently selected EF is invalidated and the file status of the EF does not allow for reading an invalidated file, the status word INVALIDATION_STATUS_CONTRADICTION shall be returned.

CRRC5: If the entry point call causes a memory problem (e.g. memory access error), the status word MEMORY_PROBLEM shall be returned.

CRRC6: If the entry point call causes an error to occur that is not expected and thus not handled, the status word INTERNAL_ERROR shall be returned.

6.1.6.2 Test Procedure

```
SET Test TO CatReadRecord
AND Data55 TO {0x55,0x55,0x55,0x55}
AND DataAA TO {0xAA,0xAA,0xAA,0xAA}.

/* Test Case 1: Read Absolute Linear Fixed EF */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
THEN {CatReadRecord(1, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_OK}
AND  {bytes == 4}
AND  {memcmp(buffer, Data55, bytes) == 0}.

/* Test Case 2: Read Current Fixed EF */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
THEN {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_OK}
AND  {bytes == 4}
AND  {memcmp(buffer, Data55, bytes) == 0}.

/* Test Case 3: Read Next from Linear Fixed EF */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
THEN {CatReadRecord(0, REC_ACC_MODE_NEXT, 0, &bytes, buffer) == SW_OK}
AND  {bytes == 4}
AND  {memcmp(buffer, DataAA, bytes) == 0}.

/* Test Case 4: Read  Next from Linear Fixed EF */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 0}
THEN {CatReadRecord(0, REC_ACC_MODE_NEXT, 0, &bytes, buffer) == SW_RECORD_NOT_FOUND}.

/* Test Case 5: Read Previous from Linear Fixed EF */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
THEN {CatReadRecord(0, REC_ACC_MODE_PREVIOUS, 0, &bytes, buffer) == SW_OK}
AND  {bytes == 4}
AND  {memcmp(buffer, Data55, bytes) == 0}.

/* Test Case 6: Read Previous from Linear Fixed EF */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 0}
THEN {CatReadRecord(0, REC_ACC_MODE_PREVIOUS, 0, &bytes, buffer) == SW_RECORD_NOT_FOUND}.

/* Test Case 7: Read Absolute and Current from Cyclic EF */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
THEN {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_OK}
AND  {bytes == 4}
AND  {memcmp(buffer, Data55, bytes) == 0}.

/* Test Case 8: Read Next from Cyclic EF */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
THEN {CatReadRecord(0, REC_ACC_MODE_NEXT, 0, &bytes, buffer) == SW_OK}
AND  {bytes == 4}
AND  {memcmp(buffer, DataAA, bytes) == 0}.
```

```
/* Test Case 9: Read Next from Cyclic EF */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
THEN {CatReadRecord(0, REC_ACC_MODE_NEXT, 0, &bytes, buffer) == SW_OK}
AND  {bytes == 4}
AND  {memcmp(buffer, Data55, bytes) == 0}.

/* Test Case 10: Read Previous from Cyclic EF */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
THEN {CatReadRecord(0, REC_ACC_MODE_PREVIOUS, 0, &bytes, buffer) == SW_OK}
AND  {bytes == 4}
AND  {memcmp(buffer, DataAA, bytes) == 0}.

/* Test Case 11: Read Previous from Cyclic EF */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
THEN {CatReadRecord(0, REC_ACC_MODE_PREVIOUS, 0, &bytes, buffer) == SW_OK}
AND  {bytes == 4}
AND  {memcmp(buffer, Data55, bytes) == 0}.

/* Test Case 12: NumBytes > Record Length */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {bytes = 5}
THEN {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_INCORRECT_P1_P2}.

/* Test Case 13: No current record in linear fixed EF */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {bytes = 4}
THEN {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_RECORD_NOT_FOUND}.

/* Test Case 14: No EF selected */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_TNR, NULL) == SW_OK}
THEN {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) ==
SW_INCOMPATIBLE_FILE_STRUCTURE}.

/* Test Case 15: EF is neither Cyclic nor Linear Fixed */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
THEN {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_NO_CURRENT_FILE}.

/* Test Case 16: Access condition not fulfilled */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LNR, NULL) == SW_OK}
THEN {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) ==
SW_SECURITY_STATUS_NOT_SATISFIED}.

/* Test Case 17: EF is invalidated */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {(CatInvalidate(), SW_OK) == SW_OK}
THEN {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) ==
SW_REFERENCED_DATA_INVALIDATED}.
```

6.1.6.3 Test Coverage

CRR Number	Test Case Number
N1	1,2,3,5,7,8,9,10,11
N2	1,7
N3	3,4,8,9
N4	5,6,10,11
P1	Not Tested
P2	Not Tested
P3	4
P4	6
C1	14
C2	15
C3	16
C4	17
C5, C6	Not Tested

6.1.7 CatUpdateRecord

Test Area Reference: API_CatUpdateRecord

6.1.7.1 Conformance Requirements

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

```
UINT16 CatUpdateRecord(DWORD RecordNumber, CatRecordAccessModes Mode,
                       DWORD Offset, DWORD NumBytes, void *Buffer);
```

6.1.7.1.1 Normal execution

CRRN1: NumBytes of data starting at Offset in the record specified by Mode and RecordNumber of the current selected linear fixed or cyclic EF are overwritten by the data in Buffer.

CRRN2: If the access mode is ABSOLUTE or CURRENT and the file is a linear fixed EF:

- the record addressed by RecordNumber will be updated;
- if RecordNumber is 0 the current selected record will be updated; and
- the current record pointer shall not change.

CRRN3: If the access mode is NEXT and the file is a linear fixed EF:

- the next record relative to the current selected record will be selected and updated;
- if no current record is selected, the first record will be selected and updated;
- the current record pointer of any other application shall not be changed.

CRRN4: If the access mode is PREVIOUS:

- the previous record relative to the current selected record will be selected and updated;
- if no current record is selected, the last record will be selected and updated;
- if a cyclic EF is updated, the oldest record will be updated independent of the current record pointer and this record becomes record number 1 and the current record;
- the current record pointer of any other application shall not be changed in case of a linear fixed EF.

6.1.7.1.2 Parameter errors

CRRP1: If the currently selected EF is linear fixed and the access mode is ABSOLUTE and RecordNumber is greater than records available, the status word RECORD_NUMBER_NOT_AVAILABLE shall be returned

CRRP2: If the currently selected EF is linear fixed and the access mode is CURRENT, RecordNumber is 0 and there is no current record selected, the status word RECORD_NUMBER_NOT_AVAILABLE shall be returned.

CRRP3: If the currently selected EF is linear fixed and the access mode is NEXT and the current record pointer is set to the last record, the status word RECORD_NUMBER_NOT_AVAILABLE shall be returned.

CRRP4: If the currently selected EF is linear fixed and the access mode is PREVIOUS and the current record pointer is set to the first record, the status word RECORD_NUMBER_NOT_AVAILABLE shall be returned.

CRRP5: If the currently selected EF is linear fixed and Offset plus NumBytes is greater than the record length, then the status word OUT_OF_FILE_BOUNDARIES shall be returned.

6.1.7.1.3 Context errors

CRRC1: If the calling Application has currently no EF selected, the status word NO_EF_SELECTED shall be returned.

CRRC2: If the currently selected EF is neither linear fixed nor cyclic, the status word FILE_INCONSISTENT shall be returned.

CRRC3: If the calling Application does not fulfil the access condition, UPDATE, to perform this function, the status word AC_NOT_FULFILLED shall be returned.

CRRC4: If the currently selected EF is invalidated and the file status of the EF does not allow for reading an invalidated file, the status word INVALIDATION_STATUS_CONTRADICTION shall be returned.

CRRC5: If the entry point call causes a memory problem (e.g. memory access error), the status word MEMORY_PROBLEM shall be returned.

CRRC6: If the entry point call causes an error to occur that is not expected and thus not handled, the status word INTERNAL_ERROR shall be returned.

6.1.7.2 Test Procedure

```
SET Test TO CatUpdateRecord
AND Data TO {1,2,3,4}.

/* Test Case 1: Update Absolute Record in Linear Fixed EF */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
THEN {CatUpdateRecord(1, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, bytes, Data) == SW_OK}
AND  {bytes == 4}
AND  {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_OK}
AND  {memcmp(buffer, Data, bytes) == 0}.

/* Test Case 2: Update Current  Record in Linear Fixed EF */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
THEN {CatUpdateRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, bytes, Data) == SW_OK}
AND  {bytes == 4}
AND  {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_OK}
AND  {memcmp(buffer, Data, bytes) == 0}.

/* Test Case 3: Update Next from Linear Fixed EF, no record pointer set */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
THEN {CatUpdateRecord(0, REC_ACC_MODE_NEXT, 0, bytes, Data) == SW_OK}
AND  {bytes == 4}
AND  {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_OK}
AND  {memcmp(buffer, Data, bytes) == 0}.

/* Test Case 4: Update Next from Linear Fixed EF, record pointer set */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
AND  {CatReadRecord(1, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_OK}
THEN {CatUpdateRecord(0, REC_ACC_MODE_NEXT, 0, bytes, Data) == SW_OK}
AND  {bytes == 4}
AND  {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_OK}
AND  {memcmp(buffer, Data, bytes) == 0}.

/* Test Case 5: Update Next from Linear Fixed EF, no more records */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {bytes = 4}
THEN {CatUpdateRecord(0, REC_ACC_MODE_NEXT, 0, bytes, Data) == SW_RECORD_NOT_FOUND}.

/* Test Case 6: Update Previous from Linear Fixed EF, no record pointer set */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
THEN {CatUpdateRecord(0, REC_ACC_MODE_PREVIOUS, 0, bytes, Data) == SW_OK}
AND  {bytes == 4}
AND  {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_OK}
AND  {memcmp(buffer, Data, bytes) == 0}.
```

```
/* Test Case 7: Update Previous from Linear Fixed EF, record pointer set */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
AND  {CatReadRecord(2, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_OK}
THEN {CatUpdateRecord(0, REC_ACC_MODE_PREVIOUS, 0, bytes, Data) == SW_OK}
AND  {bytes == 4}
AND  {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_OK}
AND  {memcmp(buffer, Data, bytes) == 0}.

/* Test Case 8: Update Previous from Linear Fixed EF, no more records */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
AND  {CatReadRecord(1, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_OK}
THEN {CatUpdateRecord(0, REC_ACC_MODE_PREVIOUS, 0, bytes, Data) == SW_RECORD_NOT_FOUND}.

/* Test Case 9: Update Previous from Cyclic EF */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_CARU, NULL) == SW_OK}
AND  {memset(buffer, 0, sizeof(buffer))}
AND  {bytes = 4}
THEN {CatUpdateRecord(0, REC_ACC_MODE_PREVIOUS, 0, bytes, Data) == SW_OK}
AND  {bytes == 4}
AND  {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_OK}
AND  {memcmp(buffer, Data, bytes) == 0}.

/* Test Case 10: No EF selected */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
THEN {CatUpdateRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, bytes, buffer) == SW_NO_CURRENT_FILE}.

/* Test Case 11: Update Absolute from Linear Fixed EF beyond Records */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {bytes = 5}
THEN {CatUpdateRecord(3, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, bytes, buffer) == SW_INCORRECT_P1_P2}.

/* Test Case 12: No current record in linear fixed EF, update current */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {bytes = 4}
THEN {CatUpdateRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, bytes, buffer) == SW_RECORD_NOT_FOUND}.

/* Test Case 13: Record Length < Offset + NumBytes  */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {bytes = 5}
THEN {CatUpdateRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, bytes, buffer) == SW_INCORRECT_P1_P2}.

/* Test Case 14: EF is neither Cyclic nor Linear Fixed */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_TNR, NULL) == SW_OK}
THEN {CatUpdateRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, bytes, buffer) ==
SW_INCOMPATIBLE_FILE_STRUCTURE}.

/* Test Case 15: Access condition not fulfilled */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LNU, NULL) == SW_OK}
THEN {CatUpdateRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, bytes, buffer) ==
SW_SECURITY_STATUS_NOT_SATISFIED}.

/* Test Case 16: EF is invalidated */
IF   {CatSelect(FID_EF_LARU, NULL) == SW_OK}
AND  {(CatInvalidate(), SW_OK) == SW_OK}
THEN {CatUpdateRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, bytes, buffer) ==
SW_REFERENCED_DATA_INVALIDATED}.
```

6.1.7.3 Test Coverage

CRR Number	Test Case Number
N1	1,2,3,4,6,7,9
N2	2
N3	3,4
N4	6,7,9
P1	11
P2	12
P3	5
P4	8
P5	13
C1	10
C2	14
C3	15
C4	16
C5, C6	Not Tested

6.1.8 CatSearch

Test Area Reference: API_CatSearch

6.1.8.1 Conformance Requirements

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

```
UINT16 CatSearch(CatSeekModes Mode, DWORD Offset, DWORD PatternLength, const void *Pattern);
```

6.1.8.1.1 Normal execution

CRRN1: If the pattern in Pattern with the length PatternLength is found in the record being specified by mode starting at Offset, the current record pointer is set to that record and the record number is returned.

CRRN2: If mode is BEGINNING_FORWARD, the search starts with the first record forward towards the end of the file.

CRRN3: If mode is END_BACKWARD, the search starts with the last record backward towards the beginning of the file.

CRRN4: If mode is NEXT_FORWARD, the search starts from the next record after the current record pointer forward towards the end of file. If no current record pointer is selected, the search starts with the first record.

CRRN5: If mode is PREVIOUS_BACKWARD, the search starts from the previous record before the current record pointer backward towards the beginning of the file. If no current record pointer is selected the search starts with the last record.

CRRN6: If pattern in patt is not found, the status word PATTERN_NOT_FOUND shall be returned.

CRRN7: If mode is NEXT_FORWARD and the record pointer is at the last record, the status word PATTERN_NOT_FOUND shall be returned.

CRRN8: If mode is PREVIOUS_BACKWARD and the record pointer is at the first record, the status word PATTERN_NOT_FOUND shall be returned.

6.1.8.1.2 Parameter errors

CRRP1: If PatternLength is greater than the size of the record of the currently selected EF, the status word OUT_OF_RECORD_BOUNDARIES shall be returned.

CRRP2: If Offset plus PatternLength is greater than the length of the pattern array patt.length, the status word OUT_OF_RECORD_BOUNDARIES shall be returned.

6.1.8.1.3 Context errors

CRRC1: If the calling Application has currently no EF selected, the status word NO_EF_SELECTED shall be returned.

CRRC2: If the currently selected EF is neither linear fixed nor cyclic, the status word FILE_INCONSISTENT shall be returned.

CRRC3: If the calling Application does not fulfil the access condition, READ, to perform this function, the status word AC_NOT_FULFILLED shall be returned.

CRRC4: If the currently selected EF is invalidated and the file status of the EF does not allow for reading an invalidated file, the status word INVALIDATION_STATUS_CONTRADICTION shall be returned.

CRRC5: If the entry point call causes a memory problem (e.g. memory access error), the status word MEMORY_PROBLEM shall be returned.

CRRC6: If the entry point call causes an error to occur that is not expected and thus not handled, the status word INTERNAL_ERROR shall be returned.

6.1.8.2 Test Procedure

```
SET Test TO CatSearch
AND PatDA TO {0xDA, 0xDA, 0xDA}
AND Pat55 TO {0x55, 0x55, 0x55}
AND PatAA TO {0xAA, 0xAA, 0xAA}
AND Pat12 TO {1,2,3,4,5,6,7,8,9,0}.

/* Test Case 1: Pattern not found */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
THEN {CatSearch(BEGINNING_FORWARD, 0, sizeof(PatDA), PatDA)== SW_REFERENCED_DATA_NOT_FOUND}.

/* Test Case 2: Search from beginning forward */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
THEN {CatSearch(BEGINNING_FORWARD, 0, sizeof(Pat55), Pat55)== SW_OK}.

/* Test Case 3: Search from end backward */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
THEN {CatSearch(END_BACKWARD, 0, sizeof(Pat55), Pat55)== SW_OK}.

/* Test Case 4: Search from next forward */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
THEN {CatSearch(NEXT_FORWARD, 0, sizeof(PatAA), PatAA)== SW_OK}.

/* Test Case 5: Last record, search next forward */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
THEN {CatSearch(NEXT_FORWARD, 0, sizeof(Pat55), Pat55)== SW_REFERENCED_DATA_NOT_FOUND}.

/* Test Case 6: Search from next forward */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
THEN {CatSearch(PREVIOUS_BACKWARD, 0, sizeof(Pat55), Pat55)== SW_OK}.

/* Test Case 7: First record, search previous backward */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
THEN {CatSearch(NEXT_FORWARD, 0, sizeof(Pat55), Pat55)== SW_REFERENCED_DATA_NOT_FOUND}.

/* Test Case 8: Pattern not found, start beyond end of record */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
THEN {CatSearch(BEGINNING_FORWARD, 10, sizeof(Pat55), Pat55)== SW_REFERENCED_DATA_NOT_FOUND}.

/* Test Case 9: Pattern not found, pattern longer than record */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
THEN {CatSearch(BEGINNING_FORWARD, 0, sizeof(Pat12), Pat12)== SW_REFERENCED_DATA_NOT_FOUND}.

/* Test Case 10: Pattern not found, offset < record length < offest+pattern length*/
```

```
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_LARU, NULL) == SW_OK}
THEN {CatSearch(BEGINNING_FORWARD, 2, sizeof(Pat55), Pat55)== SW_REFERENCED_DATA_NOT_FOUND}.

/* Test Case 11: Pattern not found, EF not linear */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_TARU, NULL) == SW_OK}
THEN {CatSearch(BEGINNING_FORWARD, 0, sizeof(Pat55), Pat55)== SW_INCOMPATIBLE_FILE_STRUCTURE}.

/* Test Case 12: No EF selected */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
THEN {CatSearch(BEGINNING_FORWARD, 0, sizeof(Pat55), Pat55) == SW_NO_CURRENT_FILE}.

/* Test 13: Security condition not satisfied */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_TNR, NULL) == SW_OK}
THEN {CatSearch(BEGINNING_FORWARD, 0, sizeof(Pat55), Pat55) == SW_SECURITY_STATUS_NOT_SATISFIED}.

/* Test Case 14: EF is invalidated */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_TNU, NULL) == SW_OK}
AND  {(CatInvalidate(), SW_OK) == SW_OK}
THEN {CatSearch(BEGINNING_FORWARD, 0, sizeof(Pat55), Pat55) == SW_REFERENCED_DATA_INVALIDATED}.
```

6.1.8.3 Test Coverage

CRR Number	Test Case Number
N1	2
N2	3
N3	4
N4	6
N5	8
N6	1
N7	5
N8	7
P1	9
P2	10
C1	12
C2	11
C3	13
C4	14
C5, C6	Not tested

6.1.9 CatIncrease

Test Area Reference: API_CatIncrease

6.1.9.1 Conformance Requirements

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

```
UINT16 CatIncrease(DWORD Increment, DWORD *Value);
```

6.1.9.1.1 Normal execution

CRRN1: The value Increment is added to the value of the last increased / updated record in the currently selected cyclic EF. The result is stored in the oldest record and returned in Value if Value is non-null. The updated record becomes record number 1 and is selected as current record.

6.1.9.1.2 Parameter errors

CRRP1: If the result of the addition is greater than the maximum value a DWORD, the status word MAX_VALUE_REACHED shall be returned.

6.1.9.1.3 Context errors

CRRC1: If the calling Application has currently no EF selected, the status word NO_EF_SELECTED shall be returned.

CRRC2: If the currently selected EF is neither linear fixed nor cyclic, the status word FILE_INCONSISTENT shall be returned.

CRRC3: If the calling Application does not fulfil the access condition, READ, to perform this function, the status word AC_NOT_FULFILLED shall be returned.

CRRC4: If the currently selected EF is invalidated and the file status of the EF does not allow for reading an invalidated file, the status word INVALIDATION_STATUS_CONTRADICTION shall be returned.

CRRC5: If the entry point call causes a memory problem (e.g. memory access error), the status word MEMORY_PROBLEM shall be returned.

CRRC6: If the entry point call causes an error to occur that is not expected and thus not handled, the status word INTERNAL_ERROR shall be returned.

CRRC 7: If Increase not allowed due to an entry in the FCP them the status word FILE_INCONSISTENT shall be returned.

6.1.9.2 Test Procedure

```
SET Test TO CatIncrease.

/* Test Case 1: Increase, verify response */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_CARU, NULL) == SW_OK}
THEN {CatIncrease((DWORD)0x00000100, &bytes) == SW_OK}
AND  {bytes == (DWORD)0x00000100}.

/* Test Case 2: Increase, verify file */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_CARU, NULL) == SW_OK}
THEN {CatIncrease((DWORD)0x00000100, &bytes) == SW_OK}
AND  {CatReadRecord(0, REC_ACC_MODE_ABSOLUTE_CURRENT, 0, &bytes, buffer) == SW_OK}
AND  {*(DWORD *)buffer == (DWORD)0x00000100}.

/* Test Case 3: Increase, exceed maximum */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_CARU, NULL) == SW_OK}
THEN {CatIncrease((DWORD)0xFFFFFFFF, &bytes) == SW_INCREASE_AT_MAX_VALUE}.

/* Test Case 4: No EF selected */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
THEN {CatIncrease((DWORD)0x00000100, &bytes) == SW_NO_CURRENT_FILE}.

/* Test Case 5: EF not cyclic */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_TARU, NULL) == SW_OK}
THEN {CatIncrease((DWORD)0x00000100, &bytes) == SW_INCOMPATIBLE_FILE_STRUCTURE}.

/* Test Case 6: Security condition not satisfied */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_CNU, NULL) == SW_OK}
THEN {CatIncrease((DWORD)0x00000100, &bytes) == SW_SECURITY_STATUS_NOT_SATISFIED}.

/* Test Case 7: EF is invalidated */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_CARU, NULL) == SW_OK}
AND  {(CatInvalidate(), SW_OK) == SW_OK}
THEN {CatIncrease((DWORD)0x00000100, &bytes) == SW_REFERENCED_DATA_INVALIDATED}.
```

6.1.9.3 Test Coverage

CRR Number	Test Case Number
N1	1,2
P1	3
C1	4
C2	5
C3	6
C4	7
C5	Not tested
C6, C7	Not tested

6.1.10 CatInvalidate

Test Area Reference: API_CatInvalidate

6.1.10.1 Conformance Requirements

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

```
UINT16 CatInvalidate(void)
```

6.1.10.1.1 Normal execution

CRRN1: The currently selected EF of the calling application shall be invalidated, i.e. the flag in the EF file status shall be changed accordingly.

6.1.10.1.2 Parameter errors

6.1.10.1.3 Context errors

CRRC1: If the calling application has currently no EF selected, the status word NO_EF_SELECTED shall be returned.

CRRC2: If the calling application does not fulfill the access condition, INVALIDATE, the status word AC_NOT_FULFILLED shall be returned.

CRRC3: If the currently selected EF is already invalidated, the status word INVALIDATION_STATUS_CONTRADICTION shall be returned.

CRRC4: If the entry point call causes a memory problem (e.g. memory access error), an instance of the status word MEMORY_PROBLEM shall be returned.

CRRC5: If the entry point call causes an error to occur that is not expected and thus not handled, the status word INTERNAL_ERROR shall be returned.

6.1.10.2 Test Procedure

```
SET Test TO CatInvalidate.

/* Test Case 1: Invalidate */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_TNR, NULL) == SW_OK}
THEN {CatInvalidate() == SW_OK}.

/* Test Case 2: No EF selected */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
THEN {CatInvalidate() == SW_NO_CURRENT_FILE}.

/* Test Case 3: Security condition not satisfied */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_CNIV, NULL) == SW_OK}
```

```
THEN {CatInvalidate() == SW_SECURITY_STATUS_NOT_SATISFIED}.

/* Test Case 4: EF is already invalidated */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_CARU, NULL) == SW_OK}
AND  {(CatInvalidate(), SW_OK) == SW_OK}
THEN {CatInvalidate() == SW_REFERENCED_DATA_INVALIDATED}.
```

6.1.10.3 Test Coverage

CRR number	Test Case Number
N1	1
C1	2
C2	3
C3	4
C4, C5	Not Tested

6.1.11 CatRehabilitate

Test Area Reference: API_CatRehabilitate

6.1.11.1 Conformance Requirements

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

```
UINT16 CatRehabilitate(void)
```

6.1.11.1.1 Normal execution

CRRN1: The currently selected EF of the calling application shall be rehabilitated, i.e. the flag in the EF file status shall be changed accordingly.

6.1.11.1.2 Parameter errors

6.1.11.1.3 Context errors

CRRC1: If the calling application has currently no EF selected, the status word NO_EF_SELECTED shall be returned.

CRRC2: If the calling application does not fulfill the access condition, INVALIDATE, the status word AC_NOT_FULFILLED shall be returned.

CRRC3: If the currently selected EF is already invalidated, the status word INVALIDATION_STATUS_CONTRADICTION shall be returned.

CRRC4: If the entry point call causes a memory problem (e.g. memory access error), an instance of the status word MEMORY_PROBLEM shall be returned.

CRRC5: If the entry point call causes an error to occur that is not expected and thus not handled, the status word INTERNAL_ERROR shall be returned.

6.1.11.2 Test Procedure

```
SET Test TO CatRehabilitate.

/* Test Case 1: Rehabilitate */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_TNR, NULL) == SW_OK}
AND  {(CatInvalidate(), SW_OK) == SW_OK}
THEN {CatRehabilitate() == SW_OK}.

/* Test Case 2: No EF selected */
```

```
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
THEN {CatRehabilitate() == SW_NO_CURRENT_FILE}.

/* Test Case 3: Security condition not satisfied */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_TNR, NULL) == SW_OK}
THEN {CatRehabilitate() == SW_SECURITY_STATUS_NOT_SATISFIED}.

/* Test Case 4: EF is may not be rehabilitated */
IF   {CatSelect(FID_DF_SIMTEST, NULL) == SW_OK}
AND  {CatSelect(FID_EF_CNRI, NULL) == SW_OK}
THEN {CatRehabilitate() == SW_REFERENCED_DATA_INVALIDATED}.
```

6.1.11.3 Test Coverage

CRR number	Test Case Number
N1	1
C1	2
C2	3
C3	4
C4, C5	Not Tested

6.2 Registry

6.2.1 CatSetMenuString

Test Area Reference: API_CatSetMenuString

6.2.1.1 Conformance Requirements

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

```
void CatSetMenuString (BYTE MenuID, BYTE MenuStringLength, const void *MenuString,
                       const CatIconIdentifier *IconIdentifier,
                       BYTE HelpAvailable, BYTE NextAction);
```

6.2.1.1.1 Normal execution

CRRN1: Menu string with optional modifiers is added to the main menu.

CRNN2: If MenuString is null, the invocation is a NOP.

6.2.1.1.2 Parameter errors

6.2.1.1.3 Context errors

6.2.1.2 Test Procedure

```
SET Test TO CatSetMenuString.

/* Test Case 1: Set non-null menu string */
IF   {(CatSetMenuString(1, 12, "Hello, world", NULL, 0, 0), SW_OK) == SW_OK}
THEN {NOP}.

/* Test Case 2: Set null menu string */
IF   {(CatSetMenuString(1, 0, NULL, NULL, 0, 0), SW_OK) == SW_OK}
THEN {NOP}.
```

6.2.1.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.2.2 CatNotifyOnFrameworkEvent

Test Area Reference: API_CatNotifyOnFrameworkEvent

6.2.2.1 Conformance Requirements

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

```
void CatNotifyOnFrameworkEvent(CatFrameworkEventType Event, BYTE Enabled);
```

6.2.2.1.1 Normal execution

CRRN1: Enable or disable monitoring of the framework event indicated by Event depending on the setting of Enabled.

6.2.2.1.2 Parameter errors

6.2.2.1.3 Context errors

6.2.2.2 Test Suite Files

Additional requirements for the UICC personalisation: None

Test Script: API_CatNotifyOnFrameworkEvent.tst

Test Application: API_ CatNotifyOnFrameworkEvent.c

6.2.2.3 Test Procedure

Id	Description	API Expectation	APDU Expectation
0	**UICC Initialisation**	Responses ignored.	
1	**Enable Monitoring of an Event** `1-CatNotifyOnFrameworkEvent(EVENT_UNRECOGNIZED_ENVELOPE, 1);`	1 – No return	
2	**Disable Monitoring of an Event** `1 –CatNotifyOnFrameworkEvent(EVENT_UNRECOGNIZED_ENVELOPE, 0);`	1 – No return	

6.2.2.4 Test Coverage

CRR number	Test Case Number
N1	1,2

6.2.3 CatNotifyOnEnvelope

Test Area Reference: API_CatNotifyOnEnvelope

6.2.3.1 Conformance Requirements

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

```
void CatNotifyOnEnvelope(CatEnvelopeTagType Tag, BYTE Enabled);
```

6.2.3.1.1 Normal execution

CRRN1: Enable or disable monitoring of the envelope event indicated by Tag depending on the setting of Enabled.

6.2.3.1.2 Parameter errors

6.2.3.1.3 Context errors

6.2.3.2 Test Procedure

```
SET Test TO CatNotifyOnFrameworkEvent.

/* Test Case 1: Enable monitoring of event */
IF   {(CatNotifyOnFrameworkEvent(SMS_PP_DOWNLOAD_TAG, 1), SW_OK) == SW_OK}
THEN {NOP}.

/* Test Case 2: Disable monitoring of event */
IF   {(CatNotifyOnFrameworkEvent(SMS_PP_DOWNLOAD_TAG, 0), SW_OK) == SW_OK}
THEN {NOP}.
```

6.2.3.3 Test Coverage

CRR number	Test Case Number
N1	1, 2

6.2.4 CatNotifyOnEvent

Test Area Reference: API_CatNotifyOnEvent

6.2.4.1 Conformance Requirements

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

```
void CatNotifyOnEvent(CatEventType EventType, BYTE Enabled);
```

6.2.4.1.1 Normal execution

CRRN1: Enable or disable monitoring of the terminal event indicated by Event depending on the setting of Enabled.

6.2.4.1.2 Parameter errors

6.2.4.1.3 Context errors

6.2.4.2 Test Procedure

```
SET Test TO CatNotifyOnEvent.

/* Test Case 1: Enable monitoring of event */
IF   {(CatNotifyOnEvent(MT_CALL_EVENT, 1), SW_OK) == SW_OK}
```

```
THEN {NOP}.

/* Test Case 2: Disable monitoring of event */
IF   {(CatNotifyOnEvent(MT_CALL_EVENT, 0), SW_OK) == SW_OK}
THEN {NOP}.
```

6.2.4.3 Test Coverage

CRR number	Test Case Number
N1	1, 2

6.3 Man-Machine Interface

6.3.1 CatAddItem

Test Area Reference: API_CatAddItem

6.3.1.1 Conformance Requirements

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

```
void CatAddItem(BYTE ItemTextLength, const void *ItemText, BYTE ItemIdentifier);
```

6.3.1.1.1 Normal execution

CRRN1: Add the provided item with the provided item identifier to the list of items in the current select list.

CRRN2: If ItemTextLength is zero or ItemText is null, the invocation is a NOP.

6.3.1.1.2 Parameter errors

6.3.1.1.3 Context errors

6.3.1.2 Test Procedure

```
SET Test TO CatAddItem.

/* Test Case 1: Add non-null item */
IF   {(CatAddItem(12, "Hello, world", 0), SW_OK) == SW_OK}
THEN {NOP}.

/* Test Case 2: Add null item */
IF   {(CatAddItem(0, NULL, 0), SW_OK) == SW_OK}
THEN {NOP}.
```

6.3.1.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.3.2 CatSelectItem

Test Area Reference: API_CatSelectItem

6.3.2.1 Conformance Requirements

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

```
void CatSelectItem(BYTE TitleLength, const void *Title, CatSelectItemOptions Options);
```

6.3.2.1.1 Normal execution

CRRN1: Create a new select list with the provided title and options.

CRRN2: If TitleLength is zero or Title is null, the select list has no title.

6.3.2.1.2 Parameter errors

6.3.2.1.3 Context errors

6.3.2.2 Test Procedure

```
SET Test TO CatSelectItem.

/* Test Case 1: Initialize with title */
IF   {(CatSelectItem(5, "Title", 0), SW_OK) == SW_OK}
THEN {NOP}.

/* Test Case 2: Initialize without title */
IF   {(CatSelectItem(0, NULL, 0), SW_OK) == SW_OK}
THEN {NOP}.
```

6.3.2.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.3.3 CatEndSelectItem

Test Area Reference: API_CatEndSelectItem

6.3.3.1 Conformance Requirements

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

```
CatGeneralResult CatEndSelectItem(BYTE *SelectedItem, const CatIconIdentifier *IconIdentifier);
```

6.3.3.1.1 Normal execution

CRRN1: Send a SELECT ITEM proactive command to the terminal using the current select list and the indicated icon; return the item picked in SelectedItem.

CRNN2: If SelectedItem is null, the invocation is for the return value only.

CRNN3: If there is no current select list or if no items have been added to the current select list, the invocation does not send a proactive command to the terminal but rather returns CAT_SUCCESS and sets *SelelectedItem to zero if SelelectedItem is non-null.

6.3.3.1.2 Parameter errors

6.3.3.1.3 Context errors

6.3.3.2 Test Procedure

```
SET Test TO CatEndSelectItem.

/* Test Case 1: Non-null select list and non-null select item */
IF   {(CatSelectItem(4, "Pick", 0), SW_OK) == SW_OK}
AND  {(CatAddItem(1, "1", 0), SW_OK) == SW_OK}
AND  {(CatAddItem(1, "2", 0), SW_OK) == SW_OK}
THEN {CatEndSelectItem(buffer, NULL) == CAT_COMMAND_SUCCESSFUL}
AND  {memcmp(proactiveCommandBuffer, endSelectItem, sizeof(endSelectItem)) == 0}.

/* Test Case 2: Null select item */
IF   {(CatSelectItem(4, "Pick", 0), SW_OK) == SW_OK}
AND  {(CatAddItem(1, "1", 0), SW_OK) == SW_OK}
AND  {(CatAddItem(1, "2", 0), SW_OK) == SW_OK}
THEN {CatEndSelectItem(buffer, NULL) == CAT_COMMAND_SUCCESSFUL}
AND  {memcmp(proactiveCommandBuffer, endSelectItem, sizeof(endSelectItem)) == 0}.

/* Test Case 3: Null select list */
IF   {CatEndSelectItem(NULL, NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.
```

6.3.3.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2
N3	3

6.3.4 CatDisplayText

Test Area Reference: API_CatDisplayText

6.3.4.1 Conformance Requirements

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

```
CatGeneralResult CatDisplayText(CatDCSValue TextDCS, BYTE TextLength, const void *Text,
            CatDisplayTextOptions Options, const CatIconIdentifier *IconIdentifier,
            BYTE ImmediateResponse);
```

6.3.4.1.1 Normal execution

CRRN1: Send a DISPLAY TEXT proactive command to the terminal with the provided text with the provided options and icon.

CRRN2: If TextLength is zero or Text is null, the invocation is a NOP that returns CAT_SUCCESS.

6.3.4.1.2 Parameter errors

6.3.4.1.3 Context errors

6.3.4.2 Test Procedure

```
SET Test TO CatDisplayText.
```

```
/* Test Case 1: Non-null text and non-null text length */
IF   {CatDisplayText(DCS_UCS2, 12, "Hello, world",
                NORMAL_PRIORITY_AUTO_CLEAR, NULL, 0) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, displayText, sizeof(displayText)) == 0}.

/* Test Case 2: Zero length text */
IF   {CatDisplayText(DCS_UCS2, 0, "Hello, world",
                NORMAL_PRIORITY_AUTO_CLEAR, NULL, 0) == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.

/* Test Case 3: Null text */
IF   {CatDisplayText(DCS_UCS2, 0, NULL,
                NORMAL_PRIORITY_AUTO_CLEAR, NULL, 0) == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.
```

6.3.4.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2, 3

6.3.5 CatGetInKey

Test Area Reference: API_CatGetInKey

6.3.5.1 Conformance Requirements

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

```
CatGeneralResult CatGetInKey(CatDCSValue TitleDCS, BYTE TitleLength, const void *Title,
         CatGetInKeyOptions Options, const CatIconIdentifier *IconIdentifier,
         CatDCSValue *DCSOut, void *KeyOut);
```

6.3.5.1.1 Normal execution

CRRN1: Send a GET IN KEY proactive command to the terminal with the text in Title and according to the provided Options; return a key value in *KeyOut and the alphabet indicator of the key value in *DCSOut.

CRRN2: If TitleLength is zero or Title is null, then no title is included in the proactive command.

CRRN3: If KeyOut is null, then no key value is returned.

CRRN4: If DCSOut is null, then no key value alphabet indicator is returned.

6.3.5.1.2 Parameter errors

6.3.5.1.3 Context errors

6.3.5.2 Test Procedure

```
SET Test TO CatGetInKey.

/* Test Case 1: All arguments non-null */
IF   {CatGetInKey(DCS_UCS2, 9, "Hit a Key", 0, NULL, &bytes, buffer) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getInKey1, sizeof(getInKey1)) == 0}.

/* Test Case 2: Text null */
IF   {CatGetInKey(DCS_UCS2, 0, NULL, 0, NULL, &bytes, buffer) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getInKey2, sizeof(getInKey2)) == 0}.

/* Test Case 3: Key out null */
IF   {CatGetInKey(DCS_UCS2, 9, "Hit a Key", 0, NULL, &bytes, NULL) == CAT_COMMAND_SUCCESSFUL}
```

```
THEN {memcmp(proactiveCommandBuffer, getInKey1, sizeof(getInKey1)) == 0}.

/* Test Case 4: DCS out null */
IF   {CatGetInKey(DCS_UCS2, 9, "Hit a Key", 0, NULL, NULL, buffer) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getInKey1, sizeof(getInKey1)) == 0}.
```

6.3.5.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2
N3	3
N4	4

6.3.6 CatGetInput

Test Area Reference: API_CatGetInput

6.3.6.1 Conformance Requirements

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

```
CatGeneralResult CatGetInput(CatDCSValue TitleDCS, BYTE TitleLength, const void *Title,
        CatGetInputOptions Options, CatDCSValue DefaultReplyDCS,
        BYTE DefaultReplyLength, const void *DefaultReply,
        BYTE MinimumResponseLength, BYTE MaximumResponseLength,
        const CatIconIdentifier *IconIdentifier,
        CatDCSValue *MsgOutDCS, BYTE *MsgOutLength, void *MsgOut);
```

6.3.6.1.1 Normal execution

CRRN1: Send a GET INPUT proactive command to the terminal with the provided Title and acquire a sequence of key hits according to Options, MinimumResponseLength and MaximumResponseLength; return the MsgOutLength length sequence in MsgOut and the alphabet indicator of the returned values in MsgOutDCS.

CRRN2: If TitleLength is zero or Title is null then no title is displayed.

CRRN3: If MsgOut is null, then the sequence is not returned.

CRRN4: If MsgOutDCS is null, then no alphabet indicator is returned.

CRRN5: If MinimumResponseLength is greater than MaximumResponseLength then it is taken to be equal to MaximumResponseLength.

6.3.6.1.2 Parameter errors

6.3.6.1.3 Context errors

6.3.6.2 Test Procedure

```
SET Test TO CatGetInput.

/* Test Case 1: All arguments non-null */
IF   {CatGetInput(DCS_UCS2, 4, "Pick", 0, DCS_UCS2, 5, buffer,
      1, 10, NULL, &dcs, &bytes, buffer)== CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getInput1, sizeof(getInput1)) == 0}.

/* Test Case 2: Text null */
IF   {CatGetInput(DCS_UCS2, 0, NULL, 0, DCS_UCS2, 5, buffer,
      1, 10, NULL, &dcs, &bytes, buffer)== CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getInput2, sizeof(getInput2)) == 0}.
```

```
/* Test Case 3: Message out null */
IF   {CatGetInput(DCS_UCS2, 4, "Pick", 0, DCS_UCS2, 5, buffer,
      1, 10, NULL, &dcs, &bytes, NULL)== CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getInput1, sizeof(getInput1)) == 0}.

/* Test Case 4: DCS out null */
IF   {CatGetInput(DCS_UCS2, 4, "Pick", 0, DCS_UCS2, 5, buffer,
      1, 10, NULL, NULL, &bytes, buffer)== CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getInput1, sizeof(getInput1)) == 0}.

/* Test Case 5: Maximum response length < Minimum response length */
IF   {CatGetInput(DCS_UCS2, 4, "Pick", 0, DCS_UCS2, 5, buffer,
      10, 1, NULL, &dcs, &bytes, buffer)== CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getInput3, sizeof(getInput3)) == 0}.
```

6.3.6.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2
N3	3
N4	4
N5	5

6.3.7 CatSetupIdleModeText

Test Area Reference: API_CatSetupIdleModeText

6.3.7.1 Conformance Requirements

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

```
CatGeneralResult CatSetupIdleModeText(CatDCSValue TextDCS, BYTE TextLength, const void *Text,
                                      const CatIconIdentifier *IconIdentifier);
```

6.3.7.1.1 Normal execution

CRRN1: Send a SETUP IDLE MODE TEXT proactive command to the terminal with Text in the indicated alphabet and with an optional icon.

CRRN2: If TextLength is zero or Text is null, then the invocation is a NOP.

6.3.7.1.2 Parameter errors

6.3.7.1.3 Context errors

6.3.7.2 Test Procedure

```
SET Test TO CatSetupIdleModeText.

/* Test Case 1: All arguments non-null */
IF   {CatSetupIdleModeText(DCS_UCS2, 3, "MMS", NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, setupIdleModeText1, sizeof(setupIdleModeText1)) == 0}.

/* Test Case 2: Text null */
IF   {CatSetupIdleModeText(DCS_UCS2, 3, NULL, NULL)== CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, setupIdleModeText2, sizeof(setupIdleModeText2)) == 0}.

/* Test Case 3: Length null */
IF   {CatSetupIdleModeText(DCS_UCS2, 0, "MMS", NULL)== CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, setupIdleModeText3, sizeof(setupIdleModeText3)) == 0}.
```

6.3.7.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2, 3

6.3.8 CatPlayTone

Test Area Reference: API_CatPlayTone

6.3.8.1 Conformance Requirements

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

```
CatGeneralResult CatPlayTone(BYTE TextLength, const void *Text,
                             CatTone Tone, CatTimeUnit Units, BYTE Duration,
                             const CatIconIdentifier *IconIdentifier);
```

6.3.8.1.1 Normal execution

CRRN1: Send a PLAY TONE proactive command to the terminal with the provided data.

CRRN2: If Duration is zero, the invocation is a NOP that returns CAT_SUCCESS.

6.3.8.1.2 Parameter errors

6.3.8.1.3 Context errors

6.3.8.2 Test Procedure

```
SET Test TO CatPlayTone.

/* Test Case 1: Duration positive, text non-null */
IF   {CatPlayTone(4, "Beep", GENERAL_BEEP, GSM_SECONDS, 5, NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, playTone1, sizeof(playTone1)) == 0}.

/* Test Case 2: Duration positive, text null */
IF   {CatPlayTone(0, NULL, GENERAL_BEEP, GSM_SECONDS, 5, NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, playTone2, sizeof(playTone2)) == 0}.

/* Test Case 3: Duration zero */
IF   {CatPlayTone(4, "Beep", GENERAL_BEEP, GSM_SECONDS, 0, NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, playTone3, sizeof(playTone3)) == 0}.
```

6.3.8.3 Test Coverage

CRR number	Test Case Number
N1	1, 2
N2	3

6.4 Timers

6.4.1 CatGetTimer

Test Area Reference: API_CatGetTimer

6.4.1.1 Conformance Requirements

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

```
BYTE CatGetTimer(void);
```

6.4.1.1.1 Normal execution

CRRN1: Send a GET TIMER proactive command to the terminal; return a timer index.

6.4.1.1.2 Parameter errors

6.4.1.1.3 Context errors

6.4.1.2 Test Procedure

```
SET Test TO CatGetTimer.

/* Test Case 1: Acquire a timer */
IF   {(CatGetTimer(),CAT_COMMAND_SUCCESSFUL)  == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getTimer, sizeof(getTimer)) == 0}.
```

6.4.1.3 Test Coverage

CRR number	Test Case Number
N1	1

6.4.2 CatFreeTimer

Test Area Reference: API_CatFreeTimer

6.4.2.1 Conformance Requirements

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

```
void CatFreeTimer(BYTE TimerID);
```

6.4.2.1.1 Normal execution

CRRN1: Send a FREE TIMER proactive command to the terminal with the provided timer index.

6.4.2.1.2 Parameter errors

6.4.2.1.3 Context errors

6.4.2.2 Test Procedure

```
SET Test TO CatFreeTimer.

/* Test Case 1: Free a timer */
IF   {(bytes = CatGetTimer(),CAT_COMMAND_SUCCESSFUL)  == CAT_COMMAND_SUCCESSFUL}
AND  {(CatFreeTimer((BYTE)bytes),CAT_COMMAND_SUCCESSFUL)  == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, freeTimer, sizeof(freeTimer)) == 0}.
```

6.4.2.3 Test Coverage

CRR number	Test Case Number
N1	1

6.4.3 CatStartTimer

Test Area Reference: API_CatStartTimer

6.4.3.1 Conformance Requirements

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

```
void CatStartTimer(BYTE TimerID, CatTimerValue *TimerValue);
```

6.4.3.1.1 Normal execution

CRRN1: Send a START TIMER proactive command to the terminal with the provided data.

CRRN2: If TimerValue is null, the invocation is a NOP.

6.4.3.1.2 Parameter errors

6.4.3.1.3 Context errors

6.4.3.2 Test Procedure

```
SET Test TO CatStartTimer.

/* Test Case 1: Start a timer */
IF   {(bytes = CatGetTimer(),CAT_COMMAND_SUCCESSFUL)  == CAT_COMMAND_SUCCESSFUL}
AND  {(CatStartTimer((BYTE)bytes, &duration),CAT_COMMAND_SUCCESSFUL)  == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, startTimer, sizeof(startTimer)) == 0}.

/* Test Case 2: Null duration */
IF   {(bytes = CatGetTimer(),CAT_COMMAND_SUCCESSFUL)  == CAT_COMMAND_SUCCESSFUL}
AND  {(CatStartTimer((BYTE)bytes, NULL),CAT_COMMAND_SUCCESSFUL)  == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.
```

6.4.3.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.4.4 CatGetTimerValue

Test Area Reference: API_CatGetTimerValue

6.4.4.1 Conformance Requirements

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

```
void CatGetTimerValue(BYTE TimerID, CatTimerValue *TimerValue);
```

6.4.4.1.1 Normal execution

CRRN1: Send a GET TIMER VALUE to the terminal; return a timer value in *TimerValue if TimerValue is non-null.

CRRN2: If TimerValue is null, the invocation is a NOP.

6.4.4.1.2 Parameter errors

6.4.4.1.3 Context errors

6.4.4.2 Test Procedure

```
SET Test TO CatGetTimerValue.

/* Test Case 1: Get a timer value */
IF   {(bytes = CatGetTimer(),CAT_COMMAND_SUCCESSFUL)  == CAT_COMMAND_SUCCESSFUL}
AND  {(CatGetTimerValue((BYTE)bytes, &duration),CAT_COMMAND_SUCCESSFUL)  == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getTimerValue, sizeof(getTimerValue)) == 0}.

/* Test Case 2: Null duration */
IF   {(bytes = CatGetTimer(),CAT_COMMAND_SUCCESSFUL)  == CAT_COMMAND_SUCCESSFUL}
AND  {(CatGetTimerValue((BYTE)bytes, NULL),CAT_COMMAND_SUCCESSFUL)  == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.
```

6.4.4.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.5 Supplementary Card Reader Management

6.5.1 CatPowerOnCard

Test Area Reference: API_CatPowerOnCard

6.5.1.1 Conformance Requirements

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

```
CatGeneralResult CatPowerOnCard(CatDevice DeviceID, BYTE *ATRLength, void *ATR);
```

6.5.1.1.1 Normal execution

CRRN1: Send a POWER ON CARD proactive command to the terminal with the provided data; return the ATR of the card in the indicated card reader in ATR.

CRRN2: If either ATRLength or ATR is null, then the proactive command is issued but the ATR is not returned.

6.5.1.1.2 Parameter errors

6.5.1.1.3 Context errors

6.5.1.2 Test Procedure

```
SET Test TO CatPowerOnCard.

/* Test Case 1: Power on a card */
IF   {CatPowerOnCard(DEVICE_CARD_READER_7, (BYTE *)&bytes, buffer) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, powerOnCard, sizeof(powerOnCard)) == 0}.

/* Test Case 2: ATR null */
IF   {CatPowerOnCard(DEVICE_CARD_READER_7, (BYTE *)&bytes, NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, powerOnCard, sizeof(powerOnCard)) == 0}.

/* Test Case 3: ATR length null */
IF   {CatPowerOnCard(DEVICE_CARD_READER_7, NULL, buffer) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, powerOnCard, sizeof(powerOnCard)) == 0}.
```

6.5.1.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2, 3

6.5.2 CatPowerOffCard

Test Area Reference: API_CatPowerOffCard

6.5.2.1 Conformance Requirements

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

```
CatGeneralResult CatPowerOffCard(CatDevice DeviceID);
```

6.5.2.1.1 Normal execution

CRRN1: Send a POWER OFF CARD proactive command to the terminal.

6.5.2.1.2 Parameter errors

6.5.2.1.3 Context errors

6.5.2.2 Test Procedure

```
SET Test TO CatPowerOffCard.

/* Test Case 1: Power off a card */
IF   {CatPowerOffCard(DEVICE_CARD_READER_7) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, powerOffCard, sizeof(powerOffCard)) == 0}.
```

6.5.2.3 Test Coverage

CRR number	Test Case Number
N1	1

6.5.3 CatPerformCardAPDU

Test Area Reference: API_CatPerformCardAPDU

6.5.3.1 Conformance Requirements

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

```
CatGeneralResult CatPerformCardAPDU(CatDevice DeviceID,
                                    BYTE CAPDULength, const void *CAPDU,
                                    BYTE *RAPDULength, void *RAPDU);
```

6.5.3.1.1 Normal execution

CRRN1: Send a PERFORM CARD APDU command to the terminal with the provided data; return the response of the card at RAPDU.

CRRN2: If RAPDULength or RAPDU is null, the response of the card in the indicated card reader is not returned.

6.5.3.1.2 Parameter errors

6.5.3.1.3 Context errors

6.5.3.2 Test Procedure

```
SET Test TO CatPerformCardAPDU
AND CAPDU TO {0x00, 0xA4, 0x00, 0x00, 0x02, 0x3F, 0x00}.

/* Test Case 1: Perform card APDU */
IF   {CatPerformCardAPDU(DEVICE_CARD_READER_7,
          sizeof(CAPDU), CAPDU, (BYTE *)(&bytes), buffer) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, performCardAPDU, sizeof(performCardAPDU)) == 0}.

/* Test Case 2: Null RAPDU length */
IF   {CatPerformCardAPDU(DEVICE_CARD_READER_7,
          sizeof(CAPDU), CAPDU, NULL, buffer) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, performCardAPDU, sizeof(performCardAPDU)) == 0}.

/* Test Case 3: NULL RAPDU */
IF   {CatPerformCardAPDU(DEVICE_CARD_READER_7,
          sizeof(CAPDU), CAPDU, (BYTE *)(&bytes), NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, performCardAPDU, sizeof(performCardAPDU)) == 0}.
```

6.5.3.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2, 3

6.5.4 CatGetReaderStatus

Test Area Reference: API_CatGetReaderStatus

6.5.4.1 Conformance Requirements

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

```
CatGeneralResult CatGetReaderStatus(CatDevice DeviceID, CatReaderStatusOptions Options,
                                    BYTE *Status);
```

6.5.4.1.1 Normal execution

CRRN1: Send a GET READER STATUS proactive command with the provided data to the terminal; return the status of the indicated device at Status.

CRRN2: If Status is null then the invocation is a NOP that returns CAT_SUCCESS.

6.5.4.1.2 Parameter errors

6.5.4.1.3 Context errors

6.5.4.2 Test Procedure

```
SET Test TO CatGetReaderStatus.

/* Test Case 1: Perform card APDU */
IF   {CatGetReaderStatus(DEVICE_CARD_READER_7, 0, buffer) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getReaderStatus, sizeof(getReaderStatus)) == 0}.

/* Test Case 2: Null status */
IF   {CatGetReaderStatus(DEVICE_CARD_READER_7, 0, NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getReaderStatus, sizeof(getReaderStatus)) == 0}.
```

6.5.4.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.6 Network Services

6.6.1 CatGetLocationInformation

Test Area Reference: API_CatGetLocationInformation

6.6.1.1 Conformance Requirements

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

```
CatGeneralResult CatGetLocationInformation(CatLocationInformation *LocationInformation);
```

6.6.1.1.1 Normal execution

CRRN1: Send a PROVIDE LOCAL INFORMATION proactive command to the terminal; return the retrieved local information in LocationInformation.

CRRN2: If LocationInformation is null, the invocation is a NOP that returns CAT_SUCCESS.

6.6.1.1.2 Parameter errors

6.6.1.1.3 Context errors

6.6.1.2 Test Procedure

```
SET Test TO CatGetLocationInformation.

/* Test Case 1: Get location information */
IF   {CatGetLocationInformation( &locationInformation) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getLocationInformation, sizeof(getLocationInformation)) == 0}.

/* Test Case 2: Null location information */
IF   {CatGetLocationInformation(NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.
```

6.6.1.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.6.2 CatGetTimingAdvance

Test Area Reference: API_CatGetTimingAdvance

6.6.2.1 Conformance Requirements

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

```
CatGeneralResult CatGetTimingAdvance(CatTimingAdvance *TimingAdvance);
```

6.6.2.1.1 Normal execution

CRRN1: Send a PROVIDE LOCAL INFORMATION proactive command to the terminal to retrieve the timing advance information; return the retrieved timing advance information at TimingAdvance.

CRRN2: If TimingAdvance is null, the invocation is a NOP that returns CAT_SUCCESS.

6.6.2.1.2 Parameter errors

6.6.2.1.3 Context errors

6.6.2.2 Test Procedure

```
SET Test TO CatGetTimingAdvance.

/* Test Case 1: Get timing advance */
IF   {CatGetTimingAdvance(&timingAdvance) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getTimingAdvance, sizeof(getTimingAdvance)) == 0}.

/* Test Case 2: Null timing advance */
IF   {CatGetTimingAdvance(NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.
```

6.6.2.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.6.3 CatGetIMEI

Test Area Reference: API_CatGetIMEI

6.6.3.1 Conformance Requirements

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

```
CatGeneralResult CatGetIMEI(BYTE IMEI[8]);
```

6.6.3.1.1 Normal execution

CRRN1: Send a PROVIDE LOCAL INFORMATION proactive command to the termina to retrieve the IMEI; return the retrieved IMEI in IMEI.

CRRN2: If IMEI is null, the invocation is a NOP that returns CAT_SUCCESS.

6.6.3.1.2 Parameter errors

6.6.3.1.3 Context errors

6.6.3.2 Test Procedure

```
SET Test TO CatGetIMEI.

/* Test Case 1: Get IMEI */
IF   {CatGetIMEI(buffer) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getIMEI, sizeof(getIMEI)) == 0}.

/* Test Case 2: Null IMEI */
IF   {CatGetIMEI(NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.
```

6.6.3.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.6.4 CatGetNetworkMeasurementResults

Test Area Reference: API_CatGetNetworkMeasurementResults

6.6.4.1 Conformance Requirements

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

```
CatGeneralResult CatGetNetworkMeasurementResults(BYTE MeasurementResults[10]);
```

6.6.4.1.1 Normal execution

CRRN1: Send a PROVIDE LOCAL INFORMATION proactive command to the terminal to retrieve the measurement results; return the retrieved measurement results in MeasurementResults.

CRRN2: If MeasurementResults is null, the invocation is a NOP that returns CAT_SUCCESS.

6.6.4.1.2 Parameter errors

6.6.4.1.3 Context errors

6.6.4.2 Test Procedure

```
SET Test TO CatGetNetworkMeasurementResults.

/* Test Case 1: Get network measurement results */
IF   {CatGetNetworkMeasurementResults(buffer) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getNetworkMeasurementResults,
sizeof(getNetworkMeasurementResults)) == 0}.

/* Test Case 2: Null network measurement results */
IF   {CatGetNetworkMeasurementResults(NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.
```

6.6.4.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.6.5 CatGetDateTimeAndTimeZone

Test Area Reference: API_CatGetDateTimeAndTimeZone

6.6.5.1 Conformance Requirements

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

```
CatGeneralResult CatGetDateTimeAndTimeZone(BYTE DateTimeAndTimeZone[7]);
```

6.6.5.1.1 Normal execution

CRRN1: Send a PROVIDE LOCAL INFORMATION proactive command to the terminal to retrieve the date, time and time zone; return the retrieved date, time and time zone in DateTimeAndTimeZone.

CRRN2: If DateTimeAndTimeZone is null, the invocation is a NOP that returns CAT_SUCCESS.

6.6.5.1.2 Parameter errors

6.6.5.1.3 Context errors

6.6.5.2 Test Procedure

```
SET Test TO CatGetDateTimeAndTimeZone.
```

```
/* Test Case 1: Get date, time and time zone */
IF   {CatGetDateTimeAndTimeZone(buffer) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getDateTimeAndTimeZone, sizeof(getDateTimeAndTimeZone)) == 0}.

/* Test Case 2: Null date, time and time zone */
IF   {CatGetDateTimeAndTimeZone(NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.
```

6.6.5.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.6.6 CatGetLanguage

Test Area Reference: API_CatGetLanguage

6.6.6.1 Conformance Requirements

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

```
CatGeneralResult CatGetLanguage(BYTE Language[2]);
```

6.6.6.1.1 Normal execution

CRRN1: Send a PROVIDE LOCAL INFORMATION proactive command to the terminal to retrieve the current language selection; return the retrieved language selection in Language.

CRRN2: If Language is null, the invocation is a NOP that returns CAT_SUCCESS.

6.6.6.1.2 Parameter errors

6.6.6.1.3 Context errors

6.6.6.2 Test Procedure

```
SET Test TO CatGetLanguage.

/* Test Case 1: Get language */
IF   {CatGetLanguage(buffer) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, getLanguage, sizeof(getLanguage)) == 0}.

/* Test Case 2: Null language */
IF   {CatGetLanguage(NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}
```

6.6.6.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.6.7 CatSetupCall

Test Area Reference: API_CatSetupCall

6.6.7.1 Conformance Requirements

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

```
CatGeneralResult CatSetupCall(BYTE UserConfirmationMessageLength,
          const void *UserConfirmationMessage,
          CatTypeOfNumberAndNumberingPlanIdentifier TONandNPI,
          BYTE DiallingNumberLength, const void *DiallingNumber,
          CatSetupCallOptions Options,
          const CatIconIdentifier *UserConfirmationIconIdentifier,
          BYTE CallSetupMessageLength, const void *CallSetupMessage,
          const CatIconIdentifier *CallSeupIconIdentifier);
```

6.6.7.1.1 Normal execution

CRRN1: Send a SETUP CALL proactive command to the terminal with the provided data.

6.6.7.1.2 Parameter errors

- CRRP1: If DiallingNumberLength is zero or Dialling Number is null, the status word CAT_REQUIRED_VALUES_MISSING is returned.

6.6.7.1.3 Context errors

6.6.7.2 Test Procedure

```
SET Test TO CatSetUpCall.

/* Test Case 1: Set up call */
IF   {CatSetupCall(3, "AOK", TON_UNKNOWN_AND_NPI_UNKNOWN, 12, "+155512345678",
             0, NULL, 10, "Calling...", NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, setUpCall, sizeof(setUpCall)) == 0}.

/* Test Case 2: Null dialing number */
IF   {CatSetupCall(3, "AOK", TON_UNKNOWN_AND_NPI_UNKNOWN, 0, NULL,
             0, NULL, 10, "Calling...", NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.
```

6.6.7.3 Test Coverage

CRR number	Test Case Number
N1	1
P1	2

6.6.8 CatSendShortMessage

Test Area Reference: API_CatSendShortMessage

6.6.8.1 Conformance Requirements

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

```
CatGeneralResult CatSendShortMessage(BYTE TitleLength, const void *Title,
                 CatTypeOfNumberAndNumberingPlanIdentifier TONandNPI,
                 BYTE AddressLength, const void *Address,
                 BYTE SmsTPDULength, const void *SmsTPDU,
                 CatSendShortMessageOptions Options,
                 const CatIconIdentifier *IconIdentifier);
```

6.6.8.1.1 Normal execution

CRRN1: Send a SEND SHORT MESSAGE proactive command to the terminal with the provided data.

6.6.8.1.2 Parameter errors

- CRRP1: If AddressLength is zero or Address is null, the value CAT_REQUIRED_VALUES_MISSING is returned.

- CRRP2: If SmsTPDULength is zero or SmsTPDU is null, the value CAT_REQUIRED_VALUES_MISSING is returned.

6.6.8.1.3 Context errors

6.6.8.2 Test Procedure

```
SET Test TO CatSendShortMessage
AND SmsTPDU TO {0x00, 0x01, 0x02, 0x03}.

/* Test Case 1: Send short message */
IF   {CatSendShortMessage(3, "SMS", TON_UNKNOWN_AND_NPI_UNKNOWN, 12, "+155512345678",
                          (BYTE)sizeof(SmsTPDU), SmsTPDU, 0, NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, sendShortMessage, sizeof(sendShortMessage)) == 0}.

/* Test Case 2: Null dialing number */
IF   {CatSendShortMessage(3, "SMS", TON_UNKNOWN_AND_NPI_UNKNOWN, 0, NULL,
                          (BYTE)sizeof(SmsTPDU), SmsTPDU, 0, NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.

/* Test Case 3: Null SMS */
IF   {CatSendShortMessage(3, "SMS", TON_UNKNOWN_AND_NPI_UNKNOWN, 12, "+155512345678",
                          0, NULL, 0, NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, sendShortMessage, sizeof(sendShortMessage)) == 0}.
```

6.6.8.3 Test Coverage

CRR number	Test Case Number
N1	1
P1	2
P2	3

6.6.9 CatSendSS

Test Area Reference: API_CatSendSS

6.6.9.1 Conformance Requirements

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

```
CatGeneralResult CatSendSS(BYTE TitleLength, const void *Title,
       CatTypeOfNumberAndNumberingPlanIdentifier TONandNPI,
       BYTE SSStringLength, const void *SSString,
       const CatIconIdentifier *IconIdentifier);
```

6.6.9.1.1 Normal execution

CRRN1: Send a SEND SS proactive command to the terminal with the provided data.

CRRN2: If SSStringLength is zero or SSString is null, the invocation is a NOP that returns CAT_SUCCESS.

6.6.9.1.2 Parameter errors

6.6.9.1.3 Context errors

6.6.9.2 Test Procedure

```
SET Test TO CatSendSS
AND SSString TO {0x00, 0x01, 0x02, 0x03}.

/* Test Case 1: Send SS */
IF   {CatSendSS(2, "SS", TON_UNKNOWN_AND_NPI_UNKNOWN,
                         (BYTE)sizeof(SSString), SSString, NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, sendSS, sizeof(sendSS)) == 0}.

/* Test Case 2: Null SS */
IF   {CatSendSS(2, "SS", TON_UNKNOWN_AND_NPI_UNKNOWN,
                         0, NULL, NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.
```

6.6.9.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.6.10 CatSendUSSD

Test Area Reference: API_CatSendUSSD

6.6.10.1 Conformance Requirements

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

```
    CatGeneralResult CatSendUSSD(BYTE TitleLength, const void *Title,
             CatDCSValue MessageDCS, BYTE MessageLength, const void *Message,
             CatDCSValue *MsgOutDCS, BYTE *MsgOutLength, void *MsgOut,
             const CatIconIdentifier *IconIdentifier);
```

6.6.10.1.1 Normal execution

CRRN1: Send a SEND USSD proactive command to the terminal with the provided data.

CRRN2: If MessageLength is zero or Message is null, the invocation is a NOP that returns CAT_SUCCESS.

CRRN3: If MsgOutDCS, MsgOutLength or MsgOut is null, only the CatGeneralResult return value of execution the function is returned.

6.6.10.1.2 Parameter errors

6.6.10.1.3 Context errors

6.6.10.2 Test Procedure

```
SET Test TO CatSendUSSD
AND SSString TO {0x00, 0x01, 0x02, 0x03}.

/* Test Case 1: Send SS */
IF   {CatSendUSSD(4, "USSD", DCS_UCS2, 4, "Time", &dcs, (BYTE*)&bytes, buffer, NULL) ==
CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, sendUSSD1, sizeof(sendUSSD1)) == 0}.
```

```
/* Test Case 2: Null USSD */
IF   {CatSendUSSD(0, NULL, DCS_UCS2, 4, "Time", &dcs, (BYTE*)&bytes, buffer, NULL) ==
CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.

/* Test Case 3: Null response */
IF   {CatSendUSSD(4, "USSD", DCS_UCS2, 4, "Time", NULL, NULL, NULL, NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, sendUSSD2, sizeof(sendUSSD2)) == 0}.
```

6.6.10.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2
N3	3

6.6.11 CatOpenCSChannel

Test Area Reference: API_CatOpenCSChannel

6.6.11.1 Conformance Requirements

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

```
CatGeneralResult CatOpenCSChannel(CatOpenChannelOptions Options,
             BYTE UserConfirmationLength, const void *UserConfirmation,
             const CatIconIdentifier *UserConfimationIconIdentifier,
             CatTypeOfNumberAndNumberingPlanIdentifier TONandNPI,
             BYTE DiallingNumberLength, const void *DiallingNumber,
             BYTE BearerDescription[3], WORD *BufferSize, CatDevice *ChannelIdentifier);
```

6.6.11.1.1 Normal execution

CRRN1: Send an OPEN CHANNEL proactive command to the terminal that opens a circuit switched channel.

6.6.11.1.2 Parameter errors

- CRRP1: If DiallingNumberLength is zero or Dialling Number is null, the status word CAT_REQUIRED_VALUES_MISSING is returned.

6.6.11.1.3 Context errors

6.6.11.2 Test Procedure

```
SET Test TO CatOpenCSChannel
AND bearer TO {0x00, 0x01, 0x02}.

/* Test Case 1: Open circuit channel */
IF   {CatOpenCSChannel(OPEN_CHANNEL_ON_DEMAND, 2, "OK", NULL, TON_UNKNOWN_AND_NPI_UNKNOWN,
                  12, "+155512345678", bearer, (WORD *)&bytes, &device) ==
CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, openCSChannel, sizeof(openCSChannel)) == 0}.

/* Test Case 2: Null dialing number */
IF   {CatOpenCSChannel(OPEN_CHANNEL_ON_DEMAND, 2, "OK", NULL, TON_UNKNOWN_AND_NPI_UNKNOWN,
                  12, "+155512345678", bearer, (WORD *)&bytes, &device) ==
CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.
```

6.6.11.3 Test Coverage

CRR number	Test Case Number
N1	1
P1	2

6.6.12 CatOpenGPRSChannel

Test Area Reference: API_CatOpenGPRSChannel

6.6.12.1 Conformance Requirements

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

```
CatGeneralResult CatOpenGPRSChannel(CatOpenChannelOptions Options,
              BYTE UserConfirmationLength, const void *UserConfirmation,
              const CatIconIdentifier *UserConfirmationIconIdentifier,
              BYTE BearerDescription[8], WORD *BufferSize, CatDevice *ChannelIdentifier);
```

6.6.12.1.1 Normal execution

CRRN1: Send an OPEN CHANNEL proactive command with the provided data to the terminal that opens a GPRS channel.

6.6.12.1.2 Parameter errors

6.6.12.1.3 Context errors

6.6.12.2 Test Procedure

```
SET Test TO CatOpenGPRSChannel
AND bearer TO {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07}.

/* Test Case 1: Open GPRS channel */
IF   {CatOpenGPRSChannel(OPEN_CHANNEL_ON_DEMAND, 2, "OK", NULL,
                  bearer, (WORD *)&bytes, &device) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, openGPRSChannel, sizeof(openGPRSChannel)) == 0}.
```

6.6.12.3 Test Coverage

CRR number	Test Case Number
N1	1

6.6.13 CatCloseChannel

Test Area Reference: API_CatCloseChannel

6.6.13.1 Conformance Requirements

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

```
CatGeneralResult CatCloseChannel(CatDevice ChannelIdentifier,
            BYTE TitleLength, const void *Title,
            const CatIconIdentifier *IconIdentifier);
```

6.6.13.1.1 Normal execution

CRRN1: Send a CLOSE CHANNEL proactive command with the provided data to the terminal.

6.6.13.1.2 Parameter errors

6.6.13.1.3 Context errors

6.6.13.2 Test Procedure

```
SET Test TO CatCloseChannel.

/* Test Case 1: Close channel */
IF   {CatCloseChannel(DEVICE_CHANNEL_1, 3, "Bye", NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, closeChannel, sizeof(closeChannel)) == 0}.
```

6.6.13.3 Test Coverage

CRR number	Test Case Number
N1	1

6.6.14 CatReceiveData

Test Area Reference: API_CatReceiveData

6.6.14.1 Conformance Requirements

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

```
CatGeneralResult CatReceiveData(CatDevice ChannelIdentifier,
            BYTE TitleLength, const void *Title, BYTE RequestedChannelDataLength,
            const CatIconIdentifier *IconIdentifier,
            BYTE *ChannelData, BYTE *NumChannelBytesRead, BYTE *NumChannelBytesLeft);
```

6.6.14.1.1 Normal execution

CRRN1: Send a RECEIVE DATA proactive command to the terminal with the provided data.

CRRN2: If ChannelData or NumChannelBytesRead is null, the number of bytes available for reception on the channel is returned at NumChannelBytesLeft.

CRRN3: If NumChannelBytesLeft is null, the invocation is a NOP that returns CAT_SUCCESS.

6.6.14.1.2 Parameter errors

6.6.14.1.3 Context errors

6.6.14.2 Test Procedure

```
SET Test TO CatReceiveData.

/* Test Case 1: Receive data */
IF   {CatReceiveData(DEVICE_CHANNEL_1, 8, "Incoming", 10, NULL,
            buffer, (BYTE *)(&bytes), (BYTE *)(&bytes)) == CAT_COMMAND_SUCCESSFUL}
```

```
THEN {memcmp(proactiveCommandBuffer, receiveData1, sizeof(receiveData1)) == 0}.

/* Test Case 2: Null data buffer */
IF   {CatReceiveData(DEVICE_CHANNEL_1, 8, "Incoming", 10, NULL,
               NULL, (BYTE *)(&bytes), (BYTE *)(&bytes)) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, receiveData2, sizeof(receiveData2)) == 0}.

/* Test Case 3: Null bytes left */
IF   {CatReceiveData(DEVICE_CHANNEL_1, 8, "Incoming", 10, NULL,
               buffer, (BYTE *)(&bytes), NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, receiveData3, sizeof(receiveData3)) == 0}.
```

6.6.14.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2
N3	3

6.6.15 CatSendData

Test Area Reference: API_CatSendData

6.6.15.1 Conformance Requirements

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

```
CatGeneralResult CatSendData(CatDevice ChannelIdentifier,
         CatSendDataOptions Options, BYTE TitleLength, const void *Title,
         BYTE ChannelDataLength, const void *ChannelData,
         const CatIconIdentifier *IconIdentifier, BYTE *ActualBytesSent);
```

6.6.15.1.1 Normal execution

CRRN1: Send a SEND DATA proactive command to the terminal with the provided data; if ActualBytesSent is non-null, the number of bytes actually sent is returnedin ActualBytesSent.

CRRN2: If ChannelDataLength is zero or ChannelData is null, the invocation is a NOP that returns CAT_SUCCESS.

6.6.15.1.2 Parameter errors

6.6.15.1.3 Context errors

6.6.15.2 Test Procedure

```
SET Test TO CatSendData.

/* Test Case 1: Receive data */
IF   {CatSendData(DEVICE_CHANNEL_1, 0, 8, "Outgoing",
            10, buffer, NULL, (BYTE *)&bytes) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, sendData1, sizeof(sendData1)) == 0}.

/* Test Case 2: Null data buffer */
IF   {CatSendData(DEVICE_CHANNEL_1, 0, 8, "Outgoing",
            10, NULL, NULL, (BYTE *)&bytes) == CAT_COMMAND_SUCCESSFUL}
THEN {NOP}.

/* Test Case 3: Null bytes sent */
IF   {CatSendData(DEVICE_CHANNEL_1, 0, 8, "Outgoing",
            10, buffer, NULL, NULL) == CAT_COMMAND_SUCCESSFUL}
THEN {memcmp(proactiveCommandBuffer, sendData2, sizeof(sendData2)) == 0}.
```

6.6.15.3 Test Coverage

CRR number	Test Case Number
N1	1,2
N2	3

6.6.16 CatGetChannelStatus

Test Area Reference: API_CatGetChannelStatus

6.6.16.1 Conformance Requirements

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

```
CatGeneralResult CatGetChannelStatus(CatDevice ChannelIdentifier, void *ChannelStatus);
```

6.6.16.1.1 Normal execution

CRRN1: Send a GET CHANNEL STATUS proactive command to the terminal; if ChannelStatus is non-null return the channel status at ChannelStatus.

CRRN2: If ChannelStatus is null, invocation is a NOP that returns CAT_SUCCESS.

6.6.16.1.2 Parameter errors

6.6.16.1.3 Context errors

6.6.16.2 Test Procedure

SET Test TO CatGetChannelStatus.

/* Test Case 1: Get channel status */

IF {CatGetChannelStatus(DEVICE_CHANNEL_1, buffer) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, sendData1, sizeof(sendData1)) == 0}.

/* Test Case 2: Null status buffer */

IF {CatGetChannelStatus(DEVICE_CHANNEL_1, NULL) == CAT_COMMAND_SUCCESSFUL}

THEN {NOP}.

6.6.16.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.6.17 CatServiceSearch

Test Area Reference: API_CatServiceSearch

6.6.17.1 Conformance Requirements

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

```
CatGeneralResult CatServiceSearch(CatBearer BearerId,
            BYTE AttributeLength, void *Attributes, void *ServiceAvailability);
```

6.6.17.1.1 Normal execution

CRRN1: Send a SERVICE SEARCH proactive command to the terminal with the provided data.

CRRN2: If AttributeLength is zero, Attributes is null or ServiceAvailability is null, invocation is a NOP that returns CAT_SUCCESS.

6.6.17.1.2 Parameter errors

6.6.17.1.3 Context errors

6.6.17.2 Test Procedure

SET Test TO CatServiceSearch.

/* Test Case 1: Service search */

IF {CatServiceSearch(BEARER_GPRS, 20, buffer, &bytes) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, serviceSearch, sizeof(serviceSearch)) == 0}.

/* Test Case 2: Attributes null */

IF {CatServiceSearch(BEARER_GPRS, 20, buffer, &bytes) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, serviceSearch, sizeof(serviceSearch)) == 0}.

/* Test Case 3: Availability null */

IF {CatServiceSearch(BEARER_GPRS, 20, buffer, &bytes) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, serviceSearch, sizeof(serviceSearch)) == 0}.

6.6.17.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2, 3

6.6.18 CatGetServiceInformation

Test Area Reference: API_CatGetServiceInformation

6.6.18.1 Conformance Requirements

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

```
CatGeneralResult CatGetServiceInformation(BYTE TitleLength, const BYTE *Title,
            const CatIconIdentifier *IconIdentifier,
            BYTE BearerId, BYTE *AttributeLength, void *Attributes,
            void *ServiceInformation);
```

6.6.18.1.1 Normal execution

CRRN1: Send a GET SERVICE INFORMATION proactive command to the terminal with the provided data.

CRNN2: If AttributesLength, Attributes or ServiceInformation is null, the invocation is a NOP that returns CAT_SUCCESS.

6.6.18.1.2 Parameter errors

6.6.18.1.3 Context errors

6.6.18.2 Test Procedure

SET Test TO CatGetServiceInformation.

/* Test Case 1: Get service information */

IF {CatGetServiceInformation(7, "Whither", NULL, BEARER_GPRS,

(BYTE *)&bytes, buffer, buffer) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, getServiceInformation, sizeof(getServiceInformation)) == 0}.

/* Test Case 2: Attributes null */

IF {CatGetServiceInformation(7, "Whither", NULL, BEARER_GPRS,

(BYTE *)&bytes, NULL, buffer) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, getServiceInformation, sizeof(getServiceInformation)) == 0}.

/* Test Case 3: Service information null */

IF {CatGetServiceInformation(7, "Whither", NULL, BEARER_GPRS,

(BYTE *)&bytes, buffer, NULL) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, getServiceInformation, sizeof(getServiceInformation)) == 0}.

Id	Description	API Expectation	APDU Expectation
0	**UICC Initialisation**	Responses ignored.	

1	**All arguments non-null**
	`CatGeneralResult rv;`
	`Char *title="Title";`
	`BYTE attr[] = {1,2,3,4}, service[16];`
	`1 – rv =`
	`CatGetServiceInformation(strlen(title),`
	`title,NULL, BEARER_SMS, sizeof(attr),`
	`attr, service);`
		1 – rv = CAT_SUCCESS	1 – SERVICE INFORMATION proactive command

2	**ServiceInformation nul**
	`CatGeneralResult rv;`
	`Char *title="Title";`
	`BYTE attr[] = {1,2,3,4}, service[16];`
	`1 – rv =`
	`CatGetServiceInformation(strlen(title),`
	`title,NULL, BEARER_SMS, sizeof(attr),`
	`attr, NULL);`
		1 –rv = CAT_SUCCESS	

6.6.18.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.6.19 CatDeclareService

Test Area Reference: API_CatDeclareService

6.6.19.1 Conformance Requirements

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

```
CatGeneralResult CatDeclareService(BYTE BearerId, BYTE ServiceId,
            CatTransportProtocol TransportProtocol,
            WORD *PortNumber, BYTE ServiceRecordLength, void *ServiceRecord);
```

6.6.19.1.1 Normal execution

CRRN1: Send a DECLARE SERVICE proactive command to the terminal with the provided data.

CRRN2: If ServiceRecordLength is zero or PortNumber or ServiceRecord is null, the invocation is a NOP that returns CAT_SUCCESS.

6.6.19.1.2 Parameter errors

6.6.19.1.3 Context errors

6.6.19.2 Test Procedure

SET Test TO CatDeclareService.

/* Test Case 1: Declare service */

IF {CatDeclareService(BEARER_GPRS, 1, TRANSPORT_PROTOCOL_TCP,

(WORD *)(&bytes), 10, buffer) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, declareService, sizeof(declareService)) == 0}.

/* Test Case 2: Service record null */

IF {CatDeclareService(BEARER_GPRS, 1, TRANSPORT_PROTOCOL_TCP,

(WORD *)(&bytes), 10, buffer) == CAT_COMMAND_SUCCESSFUL}

THEN {NOP}.

6.6.19.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.6.20 CatRunATCommand

Test Area Reference: API_CatRunATCommand

6.6.20.1 Conformance Requirements

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

```
CatGeneralResult CatRunATCommand(BYTE TitleLength, const void *Title,
            BYTE CommandLength, const void *Command,
            const CatIconIdentifier *IconIdentifier,
            void *Response, BYTE *ResponseLength);
```

6.6.20.1.1 Normal execution

CRRN1: Send a RUN AT COMMAND proactive command to the terminal with the provided data;
*ResponseLength bytes of the result of the command are returned in Response if both are non-null.

CRNN2: If ResponseLength or Response is null, no results are returned.

CRNN3: If CommandLength is zero or Command is null, invocation is a NOP that returns CAT_SUCCESS.

6.6.20.1.2 Parameter errors

6.6.20.1.3 Context errors

6.6.20.2 Test Procedure

SET Test TO CatRunATCommand.

/* Test Case 1: Run AT command */

IF {CatRunATCommand(2, "AT", 10, buffer, NULL, buffer, (BYTE *)(&bytes)) ==
CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, runATCommand, sizeof(runATCommand)) == 0}.

/* Test Case 2: Command null */

IF {CatRunATCommand(2, "AT", 0, NULL, NULL, buffer, (BYTE *)(&bytes)) == CAT_COMMAND_SUCCESSFUL}

THEN {NOP}.

/* Test Case 3: Response null */

IF {CatRunATCommand(2, "AT", 10, buffer, NULL, NULL, NULL) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, runATCommand, sizeof(runATCommand)) == 0}.

6.6.20.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2
N3	3

6.6.21 CatSendDTMFCommad

Test Area Reference: API_CatSendDTMFCommand

6.6.21.1 Conformance Requirements

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

```
CatGeneralResult CatSendDTMFCommand(BYTE TitleLength, const void *Title,
                BYTE DTMFCodeLength, const void *DTMFCode,
                const CatIconIdentifier *IconIdentifier);
```

6.6.21.1.1 Normal execution

CRRN1: Send a SEND DTMF COMMAND proactive command to the terminal with the provided data.

CRRN2: If DTMFCodeLength is zero or DTMFCode is null, the invocation is a NOP that returns CAT_SUCCESS.

6.6.21.1.2 Parameter errors

6.6.21.1.3 Context errors

6.6.21.2 Test Procedure

SET Test TO CatSendDTMFCommand.

/* Test Case 1: Send DTMF command */

IF {CatSendDTMFCommand(4, "DTMF", 10, buffer, NULL) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, sendDTMFCommand, sizeof(sendDTMFCommand)) == 0}.

/* Test Case 2: Command null */

IF {CatSendDTMFCommand(4, "DTMF", 0, NULL, NULL) == CAT_COMMAND_SUCCESSFUL}

THEN {NOP}.

6.6.21.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.7 Toolkit Application

6.7.1 main

Test Area Reference: API_main

6.7.1.1 Conformance Requirements

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

```
void main(void)
```

6.7.1.1.1 Normal execution

CRRN1: Load and execute the null application: main(){}.

6.7.1.1.2 Parameter errors

6.7.1.1.3 Context errors

6.7.1.2 Test Procedure

SET Test TO main.

/* Test Case 1: main */

IF {(CatExit(), SW_OK) == SW_OK}

THEN {NOP}.

6.7.1.3 Test Coverage

CRR number	Test Case Number
N1	1

6.7.2 CatGetFrameworkEvent

Test Area Reference: API_CatGetFrameworkEvent

6.7.2.1 Conformance Requirements

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

```
CatFrameworkEventType CatGetFrameworkEvent(void);
```

6.7.2.1.1 Normal execution

CRRN1: Return the current framework event.

6.7.2.1.2 Parameter errors

6.7.2.1.3 Context errors

6.7.2.2 Test Procedure

SET Test TO CatGetFrameworkEvent.

/* Test Case 1: Get framework event */

IF {CatGetFrameworkEvent()}

THEN {NOP}.

6.7.2.3 Test Coverage

CRR number	Test Case Number
N1	1

6.7.3 CatExit

Test Area Reference: API_CatExit

6.7.3.1 Conformance Requirements

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

```
void CatExit(void);
```

6.7.3.1.1 Normal execution

CRRN1: Load and execute the exit application: main(){catExit();}

6.7.3.1.2 Parameter errors

6.7.3.1.3 Context errors

6.7.3.2 Test Procedure

SET Test TO CatExit.

/* Test Case 1: CatExit */

IF {(CatExit(), SW_OK) == SW_OK}

THEN {NOP}.

6.7.3.3 Test Coverage

CRR number	Test Case Number
N1	1

6.8 Miscellaneous

6.8.1 CatGetTerminalProfile

Test Area Reference: API_CatGetTerminalProfile

6.8.1.1 Conformance Requirements

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

```
void CatGetTerminalProfile(BYTE *ProfileOutLength, BYTE *Profile);
```

6.8.1.1.1 Normal execution

CRRN1: Retrieve the current terminal profile.

CRRN2: If ProfileOutLength or Profile is null, the invocation is a NOP.

6.8.1.1.2 Parameter errors

6.8.1.1.3 Context errors

6.8.1.2 Test Procedure

```
SET Test TO CatGetTerminalProfile.

/* Test Case 1: Get terminal profile */
IF   {(CatGetTerminalProfile((BYTE *)(&bytes), buffer), SW_OK) == SW_OK}
THEN {NOP}.

/* Test Case 2: Null return */
IF   {(CatGetTerminalProfile(NULL, NULL), SW_OK) == SW_OK}
THEN {NOP}.
```

6.8.1.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.8.2 CatMoreTime

Test Area Reference: API_CatMoreTime

6.8.2.1 Conformance Requirements

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

```
CatGeneralResult CatMoreTime(void);
```

6.8.2.1.1 Normal execution

CRRN1: Issue a MORE TIME request to the terminal.

6.8.2.1.2 Parameter errors

6.8.2.1.3 Context errors

Test Application: API_ CatMoreTime.c

6.8.2.2 Test Procedure

SET Test TO CatMoreTime.

/* Test Case 1: Get more time */

IF {CatMoreTime() == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, moreTime, sizeof(moreTime)) == 0}.

6.8.2.3 Test Coverage

CRR number	Test Case Number
N1	1

6.8.3 CatPollingOff

Test Area Reference: API_CatPollingOff

6.8.3.1 Conformance Requirements

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

```
CatGeneralResult CatPollingOff(void);
```

6.8.3.1.1 Normal execution

CRRN1: Send the POLLING OFF proactive command to the terminal.

6.8.3.1.2 Parameter errors

6.8.3.1.3 Context errors

6.8.3.2 Test Procedure

SET Test TO CatPollingOff.

/* Test Case 1: Turn polling off */

IF {CatPollingOff() == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, pollingOff, sizeof(pollingOff)) == 0}.

6.8.3.3 Test Coverage

CRR number	Test Case Number
N1	1

6.8.4 CatPollInterval

Test Area Reference: API_CatPollInterval

6.8.4.1 Conformance Requirements

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

```
CatGeneralResult CatPollInterval(CatTimeUnit Unit, BYTE Interval,
            CatTimeInterval *ActualIntervalOut);
```

6.8.4.1.1 Normal execution

CRRN1: Send the POLL INTERVAL proactive command to the terminal.

CRRN2: If ActualIntervalOut is null, the actual interval set by the terminal is not returned.

6.8.4.1.2 Parameter errors

6.8.4.1.3 Context errors

6.8.4.2 Test Procedure

SET Test TO CatPollInterval.

/* Test Case 1: Set polling interval */

IF {CatPollInterval(GSM_SECONDS, 5, &timeInterval) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, pollInterval, sizeof(pollInterval)) == 0}.

/* Test Case 2: Null time interval return */

IF {CatPollInterval(GSM_SECONDS, 5, &timeInterval) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, pollInterval, sizeof(pollInterval)) == 0}.

6.8.4.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.8.5 CatRefresh

Test Area Reference: API_CatRefreshWithFileList

6.8.5.1 Conformance Requirements

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

```
CatGeneralResult CatRefreshWithFileList(CatRefreshOptions Options,
                    BYTE FileListLength, const void *FileList);
```

6.8.5.1.1 Normal execution

CRRN1: Send a REFRESH proactive command to the terminal with the provided data.

CRRN2: If FileListLength is zero or FileList is null, send a REFRESH proactive command that refreshes all files.

6.8.5.1.2 Parameter errors

6.8.5.1.3 Context errors

6.8.5.2 Test Procedure

AND fileList TO {0x10, 0x00, 0x20, 0x00}.

/* Test Case 1: Refresh file list */

IF {CatRefreshWithFileList(REFRESH_SIM_RESET, 2, fileList) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, refreshWithFileList1, sizeof(refreshWithFileList1)) == 0}.

/* Test Case 2: Refresh all files */

IF {CatRefreshWithFileList(REFRESH_SIM_RESET, 0, NULL) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, refreshWithFileList2, sizeof(refreshWithFileList2)) == 0}.

6.8.5.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.8.6 CatLanguageNotification

Test Area Reference: API_CatLanguageNotification

6.8.6.1 Conformance Requirements

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

```
void CatLanguageNotification(CatLanguageNotificationOptions Options,
                             const void *Language);
```

6.8.6.1.1 Normal execution

CRRN1: Send a LANGUAGE NOTIFICATION proactive command to the terminal with the provided data.

CRRN2: If Langauge is null, the invocation is a NOP.

6.8.6.1.2 Parameter errors

6.8.6.1.3 Context errors

6.8.6.2 Test Procedure

SET Test TO CatLanguageNotification.

/* Test Case 1: Refresh file list */

IF {(CatLanguageNotification(LANGUAGE_SPECIFIC_NOTIFICATION, buffer), SW_OK) == SW_OK}

THEN {memcmp(proactiveCommandBuffer, languageNotification, sizeof(languageNotification)) == 0}.

/* Test Case 2: Null buffer */

IF {(CatLanguageNotification(LANGUAGE_SPECIFIC_NOTIFICATION, NULL), SW_OK) == SW_OK}

THEN {NOP}.

6.8.6.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.8.7 CatLaunchBrowser

Test Area Reference: API_CatLaunchBrowser

6.8.7.1 Conformance Requirements

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

```
CatGeneralResult CatLaunchBrowser(CatLaunchBrowserOptions Options,
             BYTE TitleLength, const void *Title,
             BYTE URLLength, const void *URL,
             const CatIconIdentifier *IconIdentifier);
```

6.8.7.1.1 Normal execution

CRRN1: Send a LAUNCH BROWSER proactive command to the terminal with the provided data.

6.8.7.1.2 Parameter errors

6.8.7.1.3 Context errors

6.8.7.2 Test Procedure

SET Test TO CatLanguageNotification.

/* Test Case 1: Launch browser */

IF {CatLaunchBrowser(LAUNCH_USE_EXISTING_BROWSER, 6, "Google",

21, "http://www.google.com", NULL) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, launchBrowser, sizeof(launchBrowser)) == 0}.

6.8.7.3 Test Coverage

CRR number	Test Case Number
N1	1

6.9 Low-Level Interface

6.9.1 CatResetBuffer

Test Area Reference: API_CatResetBuffer

6.9.1.1 Conformance Requirements

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

```
void CatResetBuffer(void);
```

6.9.1.1.1 Normal execution

CRRN1: Clear the buffer that is used to construct proactive commands.

6.9.1.1.2 Parameter errors

6.9.1.1.3 Context errors

6.9.1.2 Test Procedure

SET Test TO CatResetBuffer.

/* Test Case 1: Reset proactive command buffer */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

THEN {NOP}.

6.9.1.3 Test Coverage

CRR number	Test Case Number
N1	1

6.9.2 CatStartProactiveCommand

Test Area Reference: API_CatStartProactiveCommand

6.9.2.1 Conformance Requirements

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

```
void CatStartProactiveCommand(BYTE Command, BYTE Options, CatDevice To);
```

6.9.2.1.1 Normal execution

CRRN1: Initialize the constructed proactive command buffer to contain the proactive command header that includes the provided data.

6.9.2.1.2 Parameter errors

6.9.2.1.3 Context errors

6.9.2.2 Test Procedure

/*

** Test Procedure for CatStartProactiveCommand

*/

SET Test TO CatStartProactiveCommand.

/* Test Case 1: Start construction of a proactive command */

IF {(CatStartProactiveCommand(SEND_SS_COMMAND, 0, DEVICE_NETWORK), SW_OK) == SW_OK}

THEN {NOP}.

6.9.2.3 Test Coverage

CRR number	Test Case Number
N1	1

6.9.3 CatSendProactiveCommand

Test Area Reference: API_CatSendProactiveCommand

6.9.3.1 Conformance Requirements

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

```
CatGeneralResult CatSendProactiveCommand(BYTE *Length);
```

6.9.3.1.1 Normal execution

CRRN1: Send the proactive command in the constructed proactive command to the terminal and return the result of the terminal's execution of the command.

CRRN2: If Length is null then the invocation is for the general result only.

6.9.3.1.2 Parameter errors

6.9.3.1.3 Context errors

6.9.3.2 Test Procedure

SET Test TO CatSendProactiveCommand.

/* Test Case 1: Send a proactive command */

IF {(CatStartProactiveCommand(SEND_SS_COMMAND, 0, DEVICE_NETWORK), SW_OK) == SW_OK}

AND {CatSendProactiveCommand((BYTE *)&bytes) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, sendProactiveCommand, sizeof(sendProactiveCommand)) == 0}.

/* Test Case 2: Null argument */

IF {(CatStartProactiveCommand(SEND_SS_COMMAND, 0, DEVICE_NETWORK), SW_OK) == SW_OK}

AND {CatSendProactiveCommand(NULL) == CAT_COMMAND_SUCCESSFUL}

THEN {memcmp(proactiveCommandBuffer, sendProactiveCommand, sizeof(sendProactiveCommand)) == 0}.

6.9.3.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.9.4 CatOpenEnvelope

Test Area Reference: API_CatOpenEnvelope

6.9.4.1 Conformance Requirements

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

```
CatEnvelopeTagType CatOpenEnvelope(BYTE *Length);
```

6.9.4.1.1 Normal execution

CRRN1: Return the tag and the length of the envelope command that arrived for the application.

CRRN2: If Length is null, on the tag is returned.

6.9.4.1.2 Parameter errors

6.9.4.1.3 Context errors

6.9.4.2 Test Procedure

SET Test TO CatOpenEnvelope.

/* Test Case 1: Open an incoming envelope */

IF {CatOpenEnvelope((BYTE *)&bytes) == EVENT_DOWNLOAD_TAG}

THEN {NOP}.

/* Test Case 2: Null argument */

IF {CatOpenEnvelope(NULL) == EVENT_DOWNLOAD_TAG}

THEN {NOP}.

6.9.4.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.9.5 CatSendEnvelopeResponse

Test Area Reference: API_CatSendEnvelopeResponse

6.9.5.1 Conformance Requirements

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

```
void CatSendEnvelopeResponse(void);
```

6.9.5.1.1 Normal execution

CRRN1: Send the contents of the constructed proactive command buffer to the terminal as a successful response to a envelope command.

6.9.5.1.2 Parameter errors

6.9.5.1.3 Context errors

6.9.5.2 Test Procedure

SET Test TO CatSendEnvelopeResponse.

/* Test Case 1: Send a response to an envelope APDU */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutByte(0x77), SW_OK) == SW_OK}

AND {(CatSendEnvelopeResponse(), SW_OK) == SW_OK}

THEN {memcmp(proactiveCommandBuffer, sendEnvelopeResponse, sizeof(sendEnvelopeResponse)) == 0}.

6.9.5.3 Test Coverage

CRR number	Test Case Number
N1	1

6.9.6 CatSendEnvelopeErrorResponse

Test Area Reference: API_CatSendEnvelopeErrorResponse

6.9.6.1 Conformance Requirements

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

```
void CatSendEnvelopeErrorResponse(void);
```

6.9.6.1.1 Normal execution

CRRN1: Send the contents of the constructed proactive command buffer to the terminal as an unsuccessful response to a envelope command.

6.9.6.1.2 Parameter errors

6.9.6.1.3 Context errors

6.9.6.2 Test Procedure

SET Test TO CatSendEnvelopeErrorResponse

AND error TO {0x6D, 0x00}.

/* Test Case 1: Send an error response to an envelope APDU */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutData(sizeof(error), error), SW_OK) == SW_OK}

AND {(CatSendEnvelopeErrorResponse(), SW_OK) == SW_OK}

THEN {memcmp(proactiveCommandBuffer, sendEnvelopeErrorResponse, sizeof(sendEnvelopeErrorResponse)) == 0}.

6.9.6.3 Test Coverage

CRR number	Test Case Number
N1	1

6.9.7 CatPutData

Test Area Reference: API_CatPutData

6.9.7.1 Conformance Requirements

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

```
void CatPutData(BYTE Length, const void *Data);
```

6.9.7.1.1 Normal execution

CRRN1: Append the provided data to the data in the constructed proactive command buffer.

CRRN2: If Length is zero or Data is null, the invocation is a NOP.

6.9.7.1.2 Parameter errors

6.9.7.1.3 Context errors

6.9.7.2 Test Procedure

SET Test TO CatPutData

AND data TO {0x01, 0x02, 0x03, 0x04}.

/* Test Case 1: Put some data into the proactive command buffer */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutData(sizeof(data), data), SW_OK) == SW_OK}

THEN {memcmp(proactiveCommandBuffer, putData, sizeof(putData)) == 0}.

/* Test Case 2: Null arguments */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutData(0, NULL), SW_OK) == SW_OK}

THEN {NOP}.

6.9.7.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.9.8 CatPutByte

Test Area Reference: API_CatPutByte

6.9.8.1 Conformance Requirements

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

```
void CatPutByte(BYTE Data)
```

6.9.8.1.1 Normal execution

CRRN1: Append the provided byte to the data in the constructed proactive command buffer.

6.9.8.1.2 Parameter errors

6.9.8.1.3 Context errors

6.9.8.2 Test Procedure

SET Test TO CatPutByte.

/* Test Case 1: Put a byte into the proactive command buffer */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutByte(0x77), SW_OK) == SW_OK}

THEN {memcmp(proactiveCommandBuffer, putByte, sizeof(putByte)) == 0}.

6.9.8.3 Test Coverage

CRR number	Test Case Number
N1	1

6.9.9 CatPutTLV

Test Area Reference: API_CatPutTLV

6.9.9.1 Conformance Requirements

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

```
void CatPutTLV(BYTE Tag, BYTE Length, const void *Value);
```

6.9.9.1.1 Normal execution

CRRN1: Append the provided TLV to the data in the constructed proactive command buffer.

CRRN2: If Value is null, the invocation is a NOP.

6.9.9.1.2 Parameter errors

6.9.9.1.3 Context errors

6.9.9.2 Test Procedure

SET Test TO CatPutTLV

AND value TO {0x01, 0x02, 0x03, 0x04}.

/* Test Case 1: Put a TLV into the proactive command buffer */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutTLV(R_APDU_TAG, sizeof(value), value), SW_OK) == SW_OK}

THEN {memcmp(proactiveCommandBuffer, putTLV, sizeof(putTLV)) == 0}.

/* Test Case 21: Null argument */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutTLV(0, 0, NULL), SW_OK) == SW_OK}

THEN {NOP}.

6.9.9.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.9.10 CatPutBytePrefixedTLV

Test Area Reference: API_CatPutBytePrefixedTLV

6.9.10.1 Conformance Requirements

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

```
void CatPutBytePrefixedTLV(BYTE Tag, BYTE Prefix, BYTE Length, const void *Value);
```

6.9.10.1.1 Normal execution

CRRN1: Append the provided data to the data in the constructed proactive command buffer.

CRRN2: If Value is null the invocation is a NOP.

6.9.10.1.2 Parameter errors

6.9.10.1.3 Context errors

6.9.10.2 Test Procedure

SET Test TO CatPutBytePrefixedTLV

AND value TO {0x01, 0x02, 0x03, 0x04}.

/* Test Case 1: Put a TLV into the proactive command buffer */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutBytePrefixedTLV(R_APDU_TAG, 0x77, sizeof(value), value), SW_OK) == SW_OK}

THEN {memcmp(proactiveCommandBuffer, putBytePrefixedTLV, sizeof(putBytePrefixedTLV)) == 0}.

/* Test Case 21: Null argument */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutBytePrefixedTLV(0, 0, 0, NULL), SW_OK) == SW_OK}

THEN {NOP}.

6.9.10.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2

6.9.11 CatPutOneByteTLV

Test Area Reference: API_CatPutOneByteTLV

6.9.11.1 Conformance Requirements

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

```
void CatPutOneByteTLV(BYTE Tag, BYTE Value);
```

6.9.11.1.1 Normal execution

CRRN1: Append the provided data to the data in the constructed proactive command buffer.

6.9.11.1.2 Parameter errors

6.9.11.1.3 Context errors

6.9.11.2 Test Procedure

SET Test TO CatPutOneByteTLV.

/* Test Case 1: Put a TLV into the proactive command buffer */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutOneByteTLV(R_APDU_TAG, 0x77), SW_OK) == SW_OK}

THEN {memcmp(proactiveCommandBuffer, putOneByteTLV, sizeof(putOneByteTLV)) == 0}.

6.9.11.3 Test Coverage

CRR number	Test Case Number
N1	1

6.9.12 CatPutTwoByteTLV

Test Area Reference: API_CatPutTwoByteTLV

6.9.12.1 Conformance Requirements

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

```
Void CatPutTwoByteTLV(BYTE Tag, BYTE Value1, BYTE Value2);
```

6.9.12.1.1 Normal execution

CRRN1: Append the provided data to the data in the constructed proactive command buffer.

6.9.12.1.2 Parameter errors

6.9.12.1.3 Context errors

6.9.12.2 Test Procedure

SET Test TO CatPutTwoByteTLV.

/* Test Case 1: Put a TLV into the proactive command buffer */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutTwoByteTLV(R_APDU_TAG, 0x77, 0x88), SW_OK) == SW_OK}

THEN {memcmp(proactiveCommandBuffer, putTwoByteTLV, sizeof(putTwoByteTLV)) == 0}.

6.9.12.3 Test Coverage

CRR number	Test Case Number
N1	1

6.9.13 CatGetByte

Test Area Reference: API_CatGetByte

6.9.13.1 Conformance Requirements

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

```
BYTE CatGetByte(void);
```

6.9.13.1.1 Normal execution

CRRN1: Return the byte at the current buffer pointer and advance the pointer by one byte.

6.9.13.1.2 Parameter errors

6.9.13.1.3 Context errors

6.9.13.2 Test Procedure

SET Test TO CatGetByte.

/* Test Case 1: Get a byte from the proactive command buffer */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutTwoByteTLV(R_APDU_TAG, 0x77, 0x88), SW_OK) == SW_OK}

AND {(CatResetBuffer(), SW_OK) == SW_OK}

THEN {CatGetByte() == R_APDU_TAG}.

6.9.13.3 Test Coverage

CRR number	Test Case Number
N1	1

6.9.14 CatGetData

Test Area Reference: API_CatGetData

6.9.14.1 Conformance Requirements

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

```
const void *CatGetData(BYTE Length);
```

6.9.14.1.1 Normal execution

CRRN1: Return the current data pointer and then increment the pointer by Length.

6.9.14.1.2 Parameter errors

6.9.14.1.3 Context errors

6.9.14.2 Test Procedure

SET Test TO CatGetData

AND data TO {R_APDU_TAG, 0x02, 0x77, 0x88}.

/* Test Case 1: Get some data from the proactive command buffer */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutTwoByteTLV(R_APDU_TAG, 0x77, 0x88), SW_OK) == SW_OK}

AND {(CatResetBuffer(), SW_OK) == SW_OK}

THEN {memcmp((BYTE *)CatGetByte(), data, sizeof(data)) == 0}.

6.9.14.3 Test Coverage

CRR number	Test Case Number
N1	1

6.9.15 CatFindNthTLV

Test Area Reference: API_CatFindNthTLV

6.9.15.1 Conformance Requirements

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

```
const void *CatFindNthTLV(BYTE Tag, BYTE Occurrence, BYTE *Length);
```

6.9.15.1.1 Normal execution

CRRN1: Return a pointer to the Occurance appearance of a TLV with the provided Tag in the data buffer and the length of this TLV at Length.

CRRN2: If there are less than Occurance appearances of TLVs with the provided Tag, return null.

CRRN3: If Length is null, do not return the Length of the TLV if one is found.

6.9.15.1.2 Parameter errors

6.9.15.1.3 Context errors

6.9.15.2 Test Procedure

SET Test TO CatFindNthTLV

AND tlvs TO {0x80, 0x01, 0x00, 0x81, 0x01, 0x01, 0x81, 0x02, 0x01, 0x02}.

/* Test Case 1: Find a TLV */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutData(sizeof(tlvs), tlvs), SW_OK) == SW_OK}

AND {(CatResetBuffer(), SW_OK) == SW_OK}

THEN {memcmp((BYTE *)CatFindNthTLV(0x81, 2, buffer), tlvs+6, buffer[0]) == 0}.

/* Test Case 2: Too few appearances */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutData(sizeof(tlvs), tlvs), SW_OK) == SW_OK}

AND {(CatResetBuffer(), SW_OK) == SW_OK}

THEN {(BYTE *)CatFindNthTLV(0x81, 3, buffer) == NULL}.

/* Test Case 3: Null argument */

IF {(CatResetBuffer(), SW_OK) == SW_OK}

AND {(CatPutData(sizeof(tlvs), tlvs), SW_OK) == SW_OK}

AND {(CatResetBuffer(), SW_OK) == SW_OK}

THEN {memcmp((BYTE *)CatFindNthTLV(0x81, 2, NULL), tlvs+6, 4) == 0}.

6.9.15.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2
N3	3

6.9.16 CatFindNthTLVInUserBuffer

Test Area Reference: API_CatFindNthTLVInUserBuffer

6.9.16.1 Conformance Requirements

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

```
const void *CatFindNthTLVInUserBuffer(BYTE BufferLen,
                      const void *Buffer, BYTE Tag,
                      BYTE Occurrence, BYTE *Length);
```

6.9.16.1.1 Normal execution

CRRN1: Return a pointer to the Occurance appearance of a TLV with the provided Tag in the provided buffer and the length of this TLV at Length.

CRRN2: If there are less than Occurance appearances of TLVs with the provided Tag, return null.

CRRN3: If Length is null, do not return the Length of the TLV if one is found.

CRNN4: If BufferLen is zero or Buffer is null, the invocation is a NOP that returns null.

6.9.16.1.2 Parameter errors

6.9.16.1.3 Context errors

6.9.16.2 Test Procedure

SET Test TO CatFindNthTLVInUserBuffer

AND tlvs TO {0x80, 0x01, 0x00, 0x81, 0x01, 0x01, 0x81, 0x02, 0x01, 0x02}.

/* Test Case 1: Find a TLV */

IF {OK}

THEN {memcmp((BYTE *)CatFindNthTLVInUserBuffer(sizeof(tlvs), tlvs, 0x81, 2, buffer), tlvs+6, buffer[0]) == 0}.

/* Test Case 2: Too few appearances */

IF {OK}

THEN {(BYTE *)CatFindNthTLVInUserBuffer(sizeof(tlvs), tlvs, 0x81, 3, buffer) == NULL}.

/* Test Case 3: Length null */

IF {OK}

THEN {CatFindNthTLVInUserBuffer(sizeof(tlvs), tlvs, 0x81, 2, NULL) == tlvs+6}.

/* Test Case 4: Buffer null */

IF {OK}

THEN {CatFindNthTLVInUserBuffer(0, NULL, 0x81, 2, buffer) == 0}.

6.9.16.3 Test Coverage

CRR number	Test Case Number
N1	1
N2	2
N3	3
N4	4

Annex A (normative): Script file syntax and format description

A.1 Syntax description

Following is a syntax description in BNF.

<statement list> ::= [<statement> **\n**] +

<statement> ::= <simple> | <switch> | <*blank line*>

<simple> ::= <reset> | <init> | <command> | <remark>

<reset> ::= **RST**

<init> ::= **INI** <*hexdata*>

<command> ::= **CMD** <*hexdata*> [<response>] (<status>)

<response> ::= **[** <*hexdata*> **]**

<status> ::= **(** <*hexdata*> **)**

<remark> ::= **REM** <*text line*>

<switch> ::= **SWI {** [<labelled list>] + **}**

<labelled list> ::= <label> **: \n** <statement list>

Description of syntax metalanguage :

\n represents a linebreak

[x] means x can appear optionally

[x] + means 1 or more appearances of x

x | y means x or y

[]{}: (bold) these are characters that appear literally in the script files

<*text line*> any character until the end of the line

<*blank line*> a line containing no text is acceptable

<*hexdata*> data written in hexadecimal, each byte separated from the following by a whitespace

Each simple statement beginning with 3 characters different than the ones defined indicates another tool command, and shall be ignored by the parser if not recognised.

`' ', '\t'` : Can be used as separator
A long statement can be broken into several lines by using the character '****' at the end of each line which is not the last one in the statement.

For more details refer to the examples in A.3.

A.2 Semantics

Following is the meaning of each of the statements:

CMD : Sends an APDU Command to the card, including (optionally) the expected response data and also (optionally) the expected status words SW1, SW2.
RST : Resets and powers on the card
INI : Performs the terminal profile with the following data. Afterwards, it shall perform all the fetch and terminal response commands until there is no proactive session in progress.
REM : Used for comments
SWI : Activates a switch condition. Every labelled list represents a list of statements to be executed, if the label matches the SW resulting from the previously executed command.

Evaluation of expected response and status in the case of a CMD:
<response> data within **[**…**]** has to be checked, it needs to be present for an outgoing command. Bytes written as XX shall not be checked by the APDU tool.
<status> status contained within **(**…**)** has to be checked; when several status are valid they shall be separated by commas. Bytes written as XX shall not be checked by the APDU tool.

A.3 Example

```
REM this is an example

RST
INI FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
REM Case 1 example
CMD A0 C2 00 00 00 (91 33 , 69 XX)

REM Case 2 example
CMD A0 B6 00 00 07 \
    [XX XX XX 55 55 XX 55] \
    (91 33 , 67 XX)


CMD A0 B6 00 00 07 \
    (91 33 , 67 XX)

CMD A0 C0 00 00 1F \
    [10 A0 00 00 00 09 00 02 FF FF FF FF 89 28 A4 05 \
    02 0D CC CC CC CC CC CC CC CC CC CC CC CC ] \
    (90 00)

REM Case 3 example
CMD A0 C2 00 00 33 \
    D1 31 82 02 83 81 06 05 80 11 22 33 44 8B 24 40 \
    08 00 24 23 85 18 41 04 51 10 10 00 00 00 00 13 \
    02 70 00 00 0E 0D 00 00 00 00 28 A4 05 00 00 00 \
    00 00 00 \
    (90 00)

REM Case 4 example with switch statement
CMD 00 A4 04 00 10 \
    A0 00 00 00 09 00 02 FF FF FF FF 89 41 04 44 02 \
    (61 XX, 6A 82)

SWI {
61 XX:
CMD 00 C0 00 00 14 \
    [10 A0 00 00 00 09 00 02 FF FF FF FF 89 41 04 44 \
    02 02 CC CC] \
    (90 00)

CMD A0 A4 00 00 02 \
    3F 00

6A 82:
RST

}
```

A.4 Style and formatting

In order to show a common appearance all the scripts shall follow those format rules:

- start always with a 'RST' followed by an 'INI' command.

– The command, data to be checked and status to be checked shall be presented in the following order:

  CMD COMMAND [EXPECTED DATA] (EXPECTED STATUS)

– APDU shall be presented with command (CLA INS P1 P2 P3) in one line and data (if present) in next line grouped 16 bytes per line (see example above).

– The expected data (if present) shall be presented in 16 bytes groups per line (see example above).

Annex B (normative): Default Prepersonalisation

B.1 General Default Prepersonalisation

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

Name	Identifier	Default Value	Special Features
EF_{ICCID}	2FE2	0F FF FF FF FF FF FF FF FF FF	This value is not compliant with GSM 11.11
EF_{IMSI}	6F07	FF FF FF FF FF FF FF FF FF	This value is not compliant with GSM 11.11
EF_{LP}	6F05	01 FF FF FF	
EF_{Kc}	6F20	FF FF FF FF FF FF FF FF 07	
$EF_{PLMNsel}$	6F30	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	
EF_{HPLMN}	6F31	05	
EF_{ACMmax}	6F37	00 00 00	Access condition UPDATE: CHV1
EF_{SST}	6F38	FF 3F C3 03 0C 00 FF 0F 00 33	
EF_{ACM}	6F39	00 00 00	Access condition UPDATE: CHV1
EF_{PUCT}	6F41	FF FF FF 00 00	Access condition UPDATE: CHV1
EF_{BCCH}	6F74	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	
EF_{ACC}	6F78	00 00	
EF_{FPLMN}	6F7B	FF FF FF FF FF FF FF FF FF FF FF FF	
EF_{LOCI}	6F7E	FF FF FF FF 00 F0 00 00 00 FF 01	
EF_{AD}	6FAD	00 FF FF	
EF_{Phase}	6FAE	03	
EF_{FDN}	6F3B	Default value in all the records: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	Records: 5
EF_{SMSP}	6F42	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	Records: 1
EF_{LND}	6F44	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	Records: 1
EF_{SMSS}	6F43	FF FF	
EF_{SMS}	6F3C	1st record: 00 FF … FF(length 176) 2nd record:00 FF … FF(length 176) 3rd record: 00 FF … FF(length 176)	Records: 3
EF_{ADN}	6F3A	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	Records: 1
EF_{CCP}	6F3D	FF FF FF FF FF FF FF FF FF FF FF FF FF FF	
EF_{MSISDN}	6F40	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	Records: 1
EF_{SDN}	6F49	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	Records: 1
EF_{SUME}	6F54	85 0C 54 4F 4F 4C 4B 49 54 20 54 45 53 54 FF FF FF FF	
EF_{CBMI}	6F45	FF FF	
EF_{IM}	4F20	FF FF FF FF FF FF FF FF FF FF FF FF	

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

B.2 File System Access Default Prepersonalisation

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

Identifier: '3113'

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

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

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

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

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

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

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

Identifier: '6F04'	Structure: cyclic			Mandatory
Record length: 3 bytes		Update activity: high		
Access Conditions: READ NEVER UPDATE ALWAYS INCREASE ALWAYS INVALIDATE ALWAYS REHABILITATE ALWAYS				
Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes
2	Test Data	00 00 00	M	3 bytes

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

Identifier: '6F05'	Structure: cyclic			Mandatory
Record length: 3 bytes		Update activity: high		
Access Conditions: READ ALWAYS UPDATE NEVER INCREASE NEVER INVALIDATE ALWAYS REHABILITATE ALWAYS				
Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes
2	Test Data	00 00 00	M	3 bytes

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

Identifier: '6F06	Structure: cyclic			Mandatory
Record length: 3 bytes		Update activity: high		
Access Conditions: READ ALWAYS UPDATE ALWAYS INCREASE NEVER INVALIDATE ALWAYS REHABILITATE ALWAYS				
Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes
2	Test Data	00 00 00	M	3 bytes

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

Identifier: '6F07'	Structure: cyclic			Mandatory
Record length: 3 bytes		Update activity: high		
Access Conditions: READ ALWAYS UPDATE ALWAYS INCREASE ALWAYS INVALIDATE NEVER REHABILITATE ALWAYS				
Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes
2	Test Data	00 00 00	M	3 bytes

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

Identifier: '6F08'	Structure: cyclic			Mandatory
Record length: 3 bytes		Update activity: high		
Access Conditions: READ ALWAYS UPDATE ALWAYS INCREASE ALWAYS INVALIDATE ALWAYS REHABILITATE NEVER				
Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	00 00 00	M	3 bytes
2	Test Data	00 00 00	M	3 bytes

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

Identifier: '6F09'	Structure: cyclic			Mandatory
Record length: 3 bytes		Update activity: high		
Access Conditions: READ ALWAYS UPDATE ALWAYS INCREASE ALWAYS INVALIDATE ALWAYS REHABILITATE ALWAYS				
Logical Record Number	Description	Default Value	M/O	Length
1	Test Data	55 55 55	M	3 bytes
2	Test Data	AA AA AA	M	3 bytes

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

Identifier: '6F0A'	Structure: linear fixed	Mandatory
Record length: 4 bytes	Update activity: low	
Access Conditions: READ NEVER UPDATE ALWAYS INVALIDATE ALWAYS REHABILITATE ALWAYS		

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

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

Identifier: '6F0B'	Structure: linear fixed	Mandatory
Record length: 4 bytes	Update activity: low	
Access Conditions: READ ALWAYS UPDATE NEVER INVALIDATE ALWAYS REHABILITATE ALWAYS		

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

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

Identifier: '6F0C'	Structure: linear fixed	Mandatory
Record length: 4 bytes	Update activity: low	
Access Conditions: READ ALWAYS UPDATE ALWAYS INVALIDATE ALWAYS REHABILITATE ALWAYS		

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

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

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

Access Conditions:
READ ALWAYS
UPDATE ALWAYS
INCREASE ALWAYS (see note 1)
INVALIDATE ALWAYS
REHABILITATE ALWAYS

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

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

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

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

Access Conditions:
READ CHV2
UPDATE ALWAYS
INCREASE ALWAYS
INVALIDATE ALWAYS
REHABILITATE ALWAYS

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

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

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

Access Conditions:
READ ALWAYS
UPDATE ALWAYS
INCREASE ALWAYS
INVALIDATE CHV1
REHABILITATE ALWAYS

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

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

Identifier: '6F10'	Structure: cyclic	Mandatory
Record length: 3 bytes	Update activity: low	
Access Conditions: READ ALWAYS UPDATE ALWAYS INCREASE CHV2 INVALIDATE ALWAYS REHABILITATE ALWAYS		

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

B.2.18 EF_{CIAA} (Cyclic Increase Access Condition ADM)

Identifier: '6F11'	Structure: cyclic	Mandatory
Record length: 3 bytes	Update activity: low	
Access Conditions: READ ALWAYS UPDATE ALWAYS INCREASE ADM INVALIDATE ALWAYS REHABILITATE ALWAYS		

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

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

Identifier: '6F12'	Structure: cyclic	Mandatory
Record length: 3 bytes	Update activity: low	
Access Conditions: READ ALWAYS UPDATE ALWAYS INCREASE ALWAYS INVALIDATE ALWAYS REHABILITATE NEVER		

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

The file status shall be invalidated as defined in [12]

Annex C (informative): Change history

Change history								
Date	**TSG #**	**TSG Doc**	**CR**	**Rev**	**Cat**	**Subject/Comment**	**Old**	**New**
2003-03	TP-19	-	-	-	-	approved	2.0.0	6.0.0
2005-01	-	-	-	-	-	History box updated to show approval; TS title expanded to show abbreviations.	6.0.0	6.0.1
2007-06	-	-	-	-	-	Update to Rel-7 version (MCC)	6.0.1	7.0.0

History

Document history		
V7.0.0	June 2007	Publication