Open Mobile API Test Specification for Transport API

Version 0.9

Published by simalliance now Trusted Connectivity Alliance

October 2013

Table of Contents

Table of Tables

1. Terminology

The given terminology is used in this document.

1.1 Abbreviations and Notations

Table 1-1: Abbreviations and Notations

Abbreviation	Description
SE	Secure Element
API	Application Programming Interface
ATR	Answer to Reset (as per ISO/IEC 7816-4)
APDU	Application Protocol Data Unit (as per ISO/IEC 7816-4)
ISO	International Organization for Standardization
ASSD	Advanced Security SD cards (SD memory cards with an embedded security system) as specified by the SD Association.
OS	Operating system
RIL	Radio Interface Layer
SFI	Short File ID
FID	File ID
FCP	File Control Parameters
MF	Master File
DF	Dedicated File
EF	Elementary File
OID	Object Identifier
PPS	Protocol Parameter Selection (as per ISO/IEC 7816-4)
DER	Distinguished Encoding Rules of ASN.1
ASN.1	Abstract Syntax Notation One
DUT	Device under test
CMD	The APDU command sent from the DUT
RESP	The APDU response sent to the DUT

1.2 Terms

Table 1-2: Terms

Term	Description
Secure Element	Smart Card Chip available in the mobile device. For example UICC/SIM, embedded Secure Element, Secure SD card, …
Applet	General term for Secure Element application: An application which is installed in the SE and runs within the SE. For example a JavaCard™ application or a native application
Application	Device/Terminal/Mobile application: An application which is installed in the mobile device and runs within the mobile device

Session	An open connection between an application on the device (e.g. mobile phone) and a SE.
Channel	An open connection between an application on the device (e.g. mobile phone) and an applet on the SE.
restarted	The DUT has been switched off completely and has been started again. No quick start, soft power off, or similar

1.3 Format of the applicability table and table of optional features

The columns in table 4.2 for the optional features have the following meaning:

Column	Meaning
Option	The optional feature supported or not by the DUT
Status	• OP - optional feature
Optional item	The mnemonic identifiers for each optional item

The columns in the applicability table 4.3 have the following meaning:

Column	Meaning
Clause	Reference to the clause index in the document
Test case number and description	The test case description in the document
SUE	The support of the tested feature/method for the Simulated Environment has the following status: • M mandatory - the capability is required to be supported. • OP optional - the capability may be supported or not. In case the support is declared by terminal, the test shall be executed. • N/A not applicable - in the given context, it is impossible to use the capability.
RSE	The support of the tested feature/method for the Real SE Environment has the following status: • M mandatory - the capability is required to be supported. • OP optional - the capability may be supported or not. In case the support is declared by terminal, the test shall be executed. • N/A not applicable - in the given context, it is impossible to use the capability.

2. Informative References

Table 2-1: Informative References

Specification	Description
[1] OMAPI V2.04	SIMalliance Open Mobile API specification V2.04Loli1266
[2] GP 2.2	GlobalPlatform Card Specification 2.2
[3] ISO/IEC 7816-4:2005	Identification cards - Integrated circuit cards - Part 4: Organisation, security and commands for interchange
[4] ISO/IEC 7816-5:2004	Identification cards - Integrated circuit cards - Part 5: Registration of application providers
[5] ISO/IEC 7816-15:2004	Identification cards - Integrated circuit cards with contacts - Part 15: Cryptographic information application
[6] PKCS #11 v2.20	Cryptographic Token Interface Standard Go to following website for PKCS#15 documentation: http://www.rsa.com/rsalabs/node.asp?id=2133
[7] PKCS #15 v1.1	Cryptographic Token Information Syntax Standard
[8] Java™ Cryptography Architecture API Specification & Reference	Go to the following website for JCA documentation: http://download.oracle.com/javase/1.4.2/docs/guide/security/CryptoSpec.html
[9] ISO/IEC 8825-1:2002 | ITU-T Recommendations X.690 (2002)	Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)
[10] GlobalPlatform Secure Element Access Control, V1.0	Specification for controlling access to Secure Elements based on access policies that are stored in the Secure Element

3. Overview

This test specification describes how to test the Transport API part of the Open Mobile API. This is the mandatory part of the Open Mobile API. The other parts of the Open Mobile API shall be tested in a similar way.

The Open Mobile API Test Specification has been released by SIMalliance for a public consultation period between 3 October and 30 November 2013. If you have any comments or suggestions to share on the contents of this document, or if you would like to bring forward any considerations for v1.0, please email SIMalliance at OMAPI@simalliance.org before 30 November 2013.

4. Applicability

4.1 Applicability of the tests

The test cases are categorized in the applicability table to use the test environment as follow:

- **Simulated UICC environment (SUE):** Test method shall be implemented in a simulated environment for the UICC.
- **Real SE environment (RSE):** Test method shall use a real SE environment. The test method shall use the one of type of SE according to implementation in the DUT and the applicability is stated in table as:
    - UICC: test cases executed with real UICC.
    - eSE: test cases executed with eSE.
    - mSD: test cases executed with mSD.
    -

If both test methods are marked as applicable (SUE and RSE), only one of test methods is required for demonstrating device compliance.

4.2 Table of DUT options

The DUT supplier shall state the support of possible options in table 4.1.

Table 4.1: Options

Item	Option	Status	Optional item
1	DUT offers possibility to log APDU communication to eSE or µSD	OP	OP-001
2	access to the basic channel is blocked by the DUT	OP	OP-002
3	access to the basic channel is allowed by the DUT	OP	OP-003
4	the ATR returned by the SE is available	OP	OP-004
5	the ATR returned by the SE is not available	OP	OP-005

4.3 Applicability table

Clause	Test case number and description	SUE	RSE		
			UICC	eSE	mSD
	class SEService				
6.1.1	Constructor: SEService(Context context, SEService.CallBack listener)	M	M	M	M
6.1.2	Method: Reader[] getReaders()	M	M	M	M
6.1.3	Method: boolean isConnected ()	M	M	M	M
6.1.3	Method: void shutdown () ID1	M	M	M	M
6.1.3	Method: void shutdown () ID2, ID3	M	M	OP-001	OP-001
6.2.1	Method: void serviceConnected(SEService service)	M	M	M	M
	class Reader				
6.3.1	Method: String getName()	M	M	M	M
6.3.2	Method SEService getSEService()	M	M	M	M
6.3.3	Method: boolean isSecureElementPresent() ID1	M	M	M	M
6.3.3	Method: boolean isSecureElementPresent() ID2	M	NA	NA	NA
6.3.4	Method: Session openSession()	M	M	M	M
6.3.5	Method: void closeSessions()	M	M	M	M
	class Session				
6.4.1	Method: Reader getReader()	M	M	M	M
6.4.2	Method: byte[] getATR() ID1	M	OP-004	OP-004	OP-004
6.4.2	Method: byte[] getATR() ID2	M	OP-004	OP-004 and OP-001	OP-004 and OP-001
6.4.2	Method: byte[] getATR() ID3	M	OP-005	OP-005	OP-005
6.4.3	Method: void close()	M	M	OP-001	OP-001
6.4.4	Method: boolean isClosed()	M	M	M	M
6.4.5	Method: void closeChannels()	M	M	OP-001	OP-001
6.4.6	Method: Channel openBasicChannel ID1 – ID6, ID8, ID9, ID11 – ID13	OP-003	OP-003	OP-003	M
6.4.6	Method: Channel openBasicChannel ID7	OP-002	OP-002	OP-002	NA
6.4.6	Method: Channel openBasicChannel ID10	OP-002	NA	NA	NA
6.4.7	Method: Channel openLogicalChannel ID1 – ID7, ID09 – ID12	M	M	M	M
6.4.7	Method: Channel openLogicalChannel ID8	M	NA	NA	NA
	class Channel				
6.5.1	Method: void close() ID1 – ID3	M	M	M	M
6.5.1	Method: void close() ID4	M	M	OP-001	OP-001
6.5.2	Method: boolean isBasicChannel() ID1	OP-003	OP-003	OP-003	M
6.5.2	Method: boolean isBasicChannel() ID2	M	M	M	M
6.5.3	Method: boolean isClosed()	M	M	M	M

Clause	Test case number and description	SUE	RSE		
			UICC	eSE	mSD
6.5.4	Method: byte[] getSelectResponse()	M	M	M	M
6.5.5	Method: Session getSession()	M	M	M	M
6.5.6	Method: byte[] transmit(byte[] command) ID1	OP-003	OP-003	OP-003	M
6.5.6	Method: byte[] transmit(byte[] command) ID2 – ID7; ID9 – ID11	M	M	M	M
6.5.6	Method: byte[] transmit(byte[] command) ID8	M	NA	NA	NA
6.5.7	Method: Boolean[] selectNext() ID1 –ID4, ID7	M	M	M	M
6.5.7	Method: Boolean[] selectNext() ID5 – ID6	M	NA	NA	NA

Table 2 specifies the applicability of each test case to the mobile.

Clause	Test case number and description	SUE	RSE		
			UICC	eSE	mSD
	class SEService				
6.1.1	Constructor: SEService(Context context, SEService.CallBack listener)	M	M	M	M
6.1.2	Method: Reader[] getReaders()	M	M	M	M
6.1.3	Method: boolean isConnected ()	M	M	M	M
6.1.3	Method: void shutdown () ID1	M	M	M	M
6.1.3	Method: void shutdown () ID2, ID3	M	M	OP-001	OP-001
6.2.1	Method: void serviceConnected(SEService service)	M	M	M	M
	class Reader				
6.3.1	Method: String getName()	M	M	M	M
6.3.2	Method SEService getSEService()	M	M	M	M
6.3.3	Method: boolean isSecureElementPresent() ID1	M	M	M	M
6.3.3	Method: boolean isSecureElementPresent() ID2	M	NA	NA	NA
6.3.4	Method: Session openSession()	M	M	M	M
6.3.5	Method: void closeSessions()	M	M	M	M
	class Session				
6.4.1	Method: Reader getReader()	M	M	M	M
6.4.2	Method: byte[] getATR() ID1	M	OP-004	OP-004	OP-004
6.4.2	Method: byte[] getATR() ID2	M	OP-004	OP-004 and OP-001	OP-004 and OP-001
6.4.2	Method: byte[] getATR() ID3	M	OP-005	OP-005	OP-005
6.4.3	Method: void close()	M	M	OP-001	OP-001
6.4.4	Method: boolean isClosed()	M	M	M	M
6.4.5	Method: void closeChannels()	M	M	OP-001	OP-001
6.4.6	Method: Channel openBasicChannel ID1 – ID6, ID8, ID9, ID11 – ID13	OP-003	OP-003	OP-003	M
6.4.6	Method: Channel openBasicChannel ID7	OP-002	OP-002	OP-002	NA
6.4.6	Method: Channel openBasicChannel ID10	OP-002	NA	NA	NA
6.4.7	Method: Channel openLogicalChannel ID1 – ID7, ID09 – ID12	M	M	M	M
6.4.7	Method: Channel openLogicalChannel ID8	M	NA	NA	NA

Clause	Test case number and description	SUE	RSE		
			UICC	eSE	mSD
	class Channel				
6.5.1	Method: void close() ID1 – ID3	M	M	M	M
6.5.1	Method: void close() ID4	M	M	OP-001	OP-001
6.5.2	Method: boolean isBasicChannel() ID1	OP-003	OP-003	OP-003	M
6.5.2	Method: boolean isBasicChannel() ID2	M	M	M	M
6.5.3	Method: boolean isClosed()	M	M	M	M
6.5.4	Method: byte[] getSelectResponse()	M	M	M	M
6.5.5	Method: Session getSession()	M	M	M	M
6.5.6	Method: byte[] transmit(byte[] command) ID1	OP-003	OP-003	OP-003	M
6.5.6	Method: byte[] transmit(byte[] command) ID2 – ID7; ID9 – ID11	M	M	M	M
6.5.6	Method: byte[] transmit(byte[] command) ID8	M	NA	NA	NA
6.5.7	Method: Boolean[] selectNext() ID1 –ID4, ID7	M	M	M	M
6.5.7	Method: Boolean[] selectNext() ID5 – ID6	M	NA	NA	NA

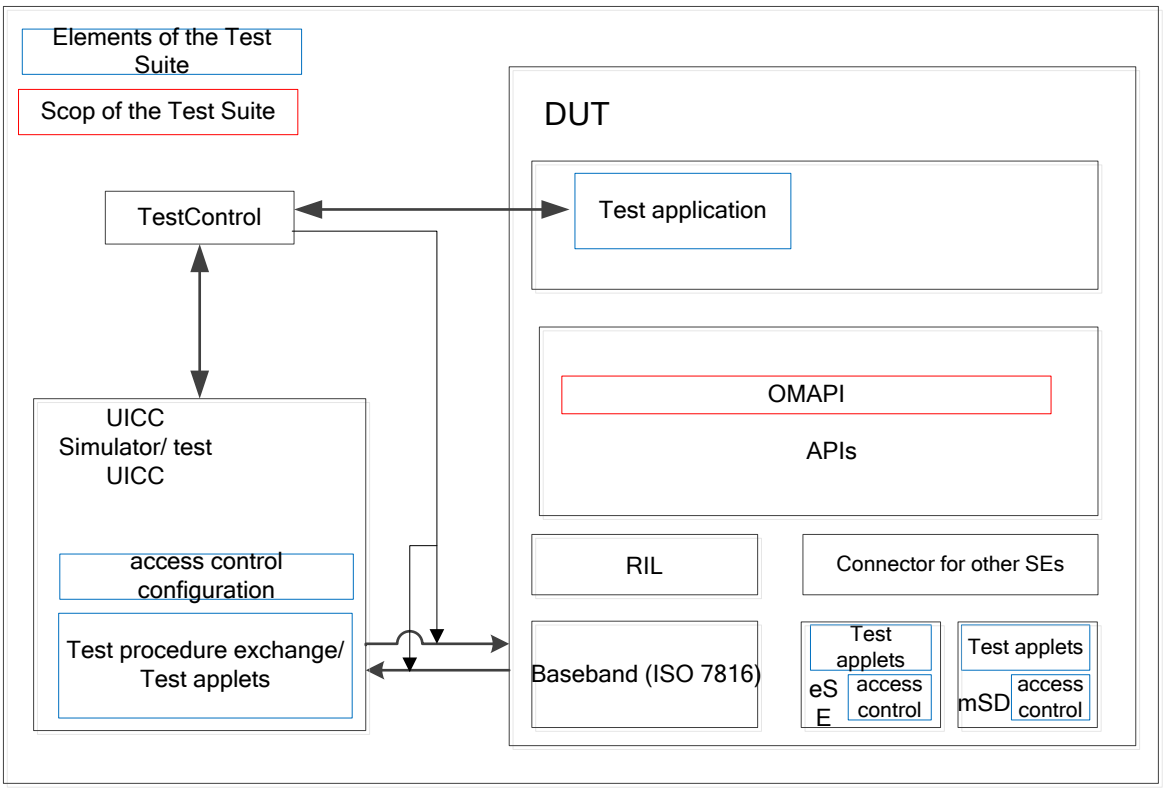
Table 2: Applicability of tests

5. Test environment

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

5.1 Test environment description

The general architecture for the test environment is:



5.2 Test tool

5.2.1 UICC Simulator

The test equipment used for executing this test specification shall meet the following requirements in order to be able to use the OMAPI implementation on a mobile device:

- be able to send and receive the commands correctly on the lower layer; i.e. to use commands as specified in ISO/IEC 7816-4.
- Be able to provide access to Basic and Logical Channels for APDUs transmission and channels can be opened simultaneously.
- the ATR used by the test equipment shall correctly show the minimum capability required to run the tests
- shall be capable to work in multi SEs environment.

- shall be able to provide the access control conditions according to section 5.7.

5.2.2 UICC, eSE and mSD

Unless otherwise specified, the following requirements and configuration shall be met:

- be able to send and receive the commands correctly on the lower layer; i.e. to use commands as specified in ISO/IEC 7816-4.
- be able to provide access to Basic and Logical Channels for APDUs transmission and channels can be opened simultaneously.
- the ATR send by the SE shall correctly show the minimum capability required to run the tests
- shall be capable to work in multi SEs environment.
- shall be able to provide the access control conditions according to section 5.7.
- all the test applets specified in section 5.6 need to be installed on the SE.

5.2.3 Test controller

The following requirements shall be provided by the test controller:

- the result of I/O commands must be presented at the application layer;
- be able to provide the test setup prior to the execution of the test, i.e. install the related application on the mobile and do any further actions required to run the test.
- be able to provide results of the tests
- shall send and/or compare all data specified in test file
- shall be able to automatically execute the tests

5.3 Test format

5.3.1 Conformance requirements

The conformance requirements are expressed in the following way:

- Method prototype as listed in Open Mobile API specification.
- Normal execution:
  - Contains normal execution and correct parameters limit values, each referenced as a Conformance Requirement Normal (CRN).
- Parameters error:
  - Contains parameter errors and incorrect parameter limit values, each referenced as a Conformance Requirement Parameter Error (CRP).
- Context error:
  - Contains errors due to the context the method is used in, each referenced as a Conformance Requirement Context Error (CRC).

5.3.2 Initial conditions

In addition to the general preconditions defined in clause 5.4,.this clause defines the initial conditions prior to the execution of each test case; i.e. for each ID.

5.3.3 Test procedure

Each test procedure contains a table of a number of test cases, each of these tests specified as follows:

Test case					
ID	**API Description**	**ISO Command Expectation DUT → UICC Simulator / SE**	**ISO Response UICC Simulator / SE → DUT**	**API Expectation**	**CRR**
The ID of the Test case	The name of the OMAPI method called by the test application.	The expected ISO command (C-APDU) received by the UICC Simulator / SE. It is sent by the DUT to UICC Simulator / SE as a result of the OMAPI method call.	The ISO response (R-APDU) sent by UICC Simulator / SE to the DUT as a response to the received ISO command.	The expected result of the OMAPI method called. E.g.: 'true' is returned.	The list of the Conformity Requirements which is the scope of the Test case.

5.4 General Initial conditions

The General Initial Conditions are a set of general prerequisites prior to the execution of testing. The following rules apply:

- DUT shall be restarted for each test case and shall be ready for test execution.
- The ISO interface is activated successfully.
- ATR is received and PPS is successfully completed.
- The test application is installed on the DUT.
- The test applets are installed on the SE

5.5 Mobile application

Unless otherwise specified, the test application shall be installed on the DUT.

This application will be available on SIMalliance web page (binary files).

5.6 Secure Element Test applets

Unless otherwise specified, the required test applets shall be installed on the SE.

These applets will be available on SIMalliance web page (binary files).

The following AID-s are used in the present document:

AID_TestApp = xx xx
AID_TestApp_1 = xx xx
AID_TestApp_multiselectable = xx xx
AID_TestApp_accessdeined = xx xx

AID_illegal_2 = xx xx
AID_illegal_1 = xx xx
AID_nonexisting = xx xx
Partial_AID_1 = xx
Partial_AID_2 = xx

Applet may requires C9 installation parameters. Applets shall be designed to return installation parameters from applet selection.
The select response of the AID_TestApp_1 shall be TestApp1_SResp = {b1,b2,b3,90,00}

5.6.1 Test APDU Interface

	Cla	Ins	P1	P2	Lc	Data	Le
Test-APDU1	00	10 (case 4)	01 (for echo)	00	length	Data	00
Test-APDU2	00	10 (case 4)	02 (echo with long delay (more than 1 sec) before return)	00	length	Data	00
Test_APDU3	00	20 (filtered APDU)	00	00	length	Data	00
Test_APDU4	00	30 (case 1)	00	00			
Test_APDU5	00	40 (case 2)	00	00			00
Test_APDU6	00	50 (case 3)	00	00	length	Data	

The length of the data and the data bytes may be adopted by the test controller for different test runs (e.g. run the test cases with different data length during different test runs). The test applet must be able to handle different data length.

5.7 Access Control Configuration

To test security errors two rules shall be defined complying with GP SEAC

- Rule 1: denies access to an specific instance of the test applet. (will be defined once the test applet is available)
- Rule 2: denies sending a specific APDU command: Test_APDU3

4.7.1 Access Control Applet (ARA)

A simple ARA applet that allows full access to any applet instance with the exception of the above denial rules will be provided on SIMalliance web page.

4.7.2 Access Control file system (ARF)

A PKCS#15 structure that allows full access to any applet instance with the exception of the above deny rules will be provided on SIMalliance web page.

.

6. Test Cases

6.1 Class SEService

The SEService realizes the communication to available Secure Elements on the device.
This is the entry point of this API. It is used to connect to the infrastructure and get access to a list of Secure Element Readers.

6.1.1 Constructor: SEService(Context context, SEService.CallBack listener)

(a) Conformance Requirements

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

```
SEService(Context context, SEService.CallBack listener)
```

Normal execution

```
CRN1: Establishes a new connection that can be used to connect to all the
Secure Elements available in the DUT.
CRN2: The isConnected() method returns true after the connection process
is finished.
CRN3: The serviceConnected() method of the listener object is called
within 10 sec.
```

Parameter errors

```
CRP1: IllegalParameterError - if the parameter context are null.
```

Context errors

```
None
```

(b) Initial Conditions

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT → UICC Simulator / SE**	**ISO Response UICC Simulator / SE → DUT**	**API Expectation**	**CRR**
1	**SEService Constructor with 2 Parameters**				
	Constructor: SEService(context, listener)	none	none	serviceConnected() method of the listener object is called within 10 sec.	CRN1 CRN3
2	**SEService Constructor and check with isConnceted**				
	1. Constructor: SEService(context, listener) 2. After seService.serviceConnected() callback received; seService.isConnected()	none	none	2. seService.isConnted() returns true	CRN2

3	**SEService Constructor with missing Context**				
	Constructor: SEService(null, listener)	none	none	`IllegalParameterError expected`	CRP1
4	**SEService Constructor with missing Listener**				
	1. Constructor: SEService(context, null) *2. -- wait 10 sec (not blocking) --* seService.isConnected()	none	none	2. seService.`isConnected() returns true`	CRP1
5	**SEService Constructor without an parameters**				
	Constructor: SEService(null, null)	none	none	`IllegalParameterError expected`	CRP1

6.1.2 Method: Reader[] getReaders()

(a) Conformance Requirements

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

```
Reader[] getReaders()
```

Normal execution

```
CRN1: Reader[] contains the list of available secure element readers.
CRN2: If there is no reader, then the array of readers returned by
        getReaders() method has length 0
CRN3: There must be no duplicated objects in the list of readers
```

Parameter errors

```
None
```

Context errors

```
None
```

(b) Initial Conditions

Supported Readers of device under test must be specified
SEService Object has been created and the isConnected() method has been called and has returned true.

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT → UICC Simulator / SE**	**ISO Response UICC Simulator / SE → DUT**	**API Expectation**	**CRR**
1	**SEService GetReaders with return of multiple readers**				
	seService.getReaders ()	None	None	Returned array contains list with the correct number of the supported readers ; There	CRN1 CRN3

				must be not duplicated entries in the list	
	SEService GetReaders with no readers present				
2	seService.getReaders ()	None	none	If there are no supported readers , the returned array shall be non null and the length of the array must be 0	CRN2

6.1.3 Method: boolean isConnected ()

(a) Conformance Requirements

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

```
boolean isConnected ()
```

Normal execution

```
CRN1: isConnected() returns true if the service is connected
CRN2: isConnected() returns false if the service is not connected
```

Parameter errors

```
None
```

Context errors

```
None
```

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT → UICC Simulator / SE**	**ISO Response UICC Simulator / SE → DUT**	**API Expectation**	**CRR**
1	**SEService isConnected returns true**				
	seService.isConnected()	none	none	Returns true	CRN1
2	**SEService isConnected return false**				
	1. seService.shutdown() 2. seService.isConnected()	none	none	2. Returns false	CRN2

6.1.4 Method: void shutdown ()

(a) Conformance Requirements

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

```
Void shutdown ()
```

Normal execution
`CRN1: Releases all Secure Elements resources allocated by this SEService.`

Parameter errors
`None`

Context errors
`None`

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT → UICC Simulator / SE**	**ISO Response UICC Simulator / SE → DUT**	**API Expectation**	**CRR**
1	**SEService shutdown with no channels open**				
	1. seService.shutdown() 2. seService.isConnected()	none	none	2. seService.isConnected returns false	CRN1
2	**SEService shutdown with one channel open**				
	1. seService.getReaders() 2. seService.openSession(firstReader) 3. seService.openLogicalChannel(AID_TestApp) 4. seService.shutdown() 5. seService.isConnected()	CMD 3-1: MANAGE CHANNEL (P1='00', P2='00') CMD 3-2: SELECT – CLA contains the Channel Number returned by the card in RESP 3-1; Data = 'AID_TestApp' CMD-4-1: MANAGE CHANNEL (P1='80')	RESP 3-1: R-APDU - Data: Channel Number; SW '90 00' RESP 3-2: R-APDU - SW '90 00' RESP 4-1: R-APDU - SW '90 00'	no errors or exceptions are expected 5. seService.isConnected returns false	
3	**SEService shutdown during transmit in different thread**				
	1. seService.getReaders() 2. seService.openSeesi	CMD 3-1:	RESP 3-1:	no errors or exceptions are expected	

	on(firstReader) 3. seService.openLogicalChannel(AID_TestApp) *4.* *-- Start new thread –* *Channel.transmit(***Test-APDU1** *5.* *-- return to first thread right after transmit returned the response* -- seService.shutdown() 6. seService.isConnected()	MANAGE CHANNEL (P1='00', P2='00') CMD 3-2: SELECT – CLA contains the Channel Number returned by the card in RESP 3-1; Data = 'AID_TestApp' CMD 4-1: APDU ('01 10 01 00 04 01 02 03 04 00') CMD 5-1: MANAGE CHANNEL (P1='80')	R-APDU - Data: Channel Number; SW '90 00' RESP 3-2: R-APDU - SW '90 00' RESP 4-1: R-APDU – '01 02 03 04' SW '90 00' RESP 5-1: R-APDU - SW '90 00'	4. byte[ ]= {'01, 02, 03, 04, 90, 00} (transmit executed successfully) 6. seService.isConnected returns false	

6.2 Class (or interface): SEService.CallBack

Interface to receive call-backs when the service is connected.
If the target language and environment allows it, then this shall be an inner interface of the SEService class.

6.2.1 Method: void serviceConnected(SEService service)

(a) Conformance Requirements

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

```
void serviceConnected(SEService service)
```

Normal execution

```
CRN1: The SEService object parameter must be the object that was created as
result of the SEService constructor and must not be null.
```

Parameter errors

```
None
```

Context errors

```
None
```

(b) Initial Conditions

SEService Constructor called

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation**	**ISO Response**	**API Expectation**	**CRR**

		DUT → UICC Simulator / SE	UICC Simulator / SE → DUT		
1	SEService Callback received after constructor				
	1. serviceConnected(SEService service) received; 2. Call seService.isConnected() of received SEService object	none	none	1. SEService object created with constructor and the one received in the callback must be identical 2. seService .isConnected returns true.	CRN1

6.3 Class Reader

The instances of this class represent Secure Element Readers connected to this device. These Readers can be physical devices or virtual devices. They can be removable or not. They can contain one Secure Element that can or cannot be removed.

6.3.1 Method: String getName()

(a) Conformance Requirements

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

```
String getName()
```

Normal execution

```
CRN1: Returns the user-friendly name of this reader. The name begins with:
- "SIM" for a SIM reader
- "SD" for a SD reader
- "eSE" for an embedded SE reader
```

Context errors

```
None
```

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.

The "reader" instance is an element of a Reader[] array, returned by invoking seService.getReaders() method.

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT → UICC Simulator / SE**	**ISO Response UICC Simulator / SE → DUT**	**API Expectation**	**CRR**
1	Get Name				
	reader.getName()	none	none	Returned String is not null and starts with correct prefix. E.g.: "SIM" for a SIM reader. No exception is	CRN1

				expected.	

6.3.2 Method: SEService getSEService()

(a) Conformance Requirements

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

```
SEService getSEService()
```

Normal execution

```
CRN1: Get the SEService that provides this Reader
```

Parameter errors

```
None
```

Context errors

```
None
```

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.
The “reader” instance is an element of a Reader[] array, returned by invoking seService.getReaders() method.

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT → UICC Simulator / SE**	**ISO Response UICC Simulator / SE→DUT**	**API Expectation**	**CRR**
1	Get SEService and compare				
	reader.getSEService ()== service	NONE	None	No exception is expected (SEService object is not null and is equal to the “service” which provides this Reader.) .	CRN1

6.3.3 Method: boolean isSecureElementPresent()

(a) Conformance Requirements

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

```
boolean isSecureElementPresent()
```

Normal execution

```
CRN1: this method checks if a Secure Element is present in the reader, in
case of its presence it returns true
CRN2: this method returns false if the Secure Element is not present in
the reader.
```

Parameter errors

```
None
```

Context errors

```
None
```

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.
The "reader" instance is an element of a Reader[] array, returned by invoking seService.getReaders() method.

Test case ID 1: The SE used for testing is available and accessible
Test case ID 2: The SE that is tested is not inserted

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT →UICC Simulator / SE**	**ISO Response UICC Simulator / SE → DUT**	**API Expectation**	**CRR**
1	Secure Element is present				
	reader.isSecureElementPresent()	None	None	True is returned No exception is expected.	CRN1
2	Secure Element is not present				
	reader.isSecureElementPresent()	None	None	False is returned No exception is expected.	CRN2

6.3.4 Method: Session openSession()

(a) Conformance Requirements

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

```
Session openSession()
```

Normal execution

```
CRN1: this method allows a developer to connect to a secure element in
the reader
CRN2: the Secure Element needs to be prepared (initialized) for
communication (i.e. switched on)
CRN3: There might be multiple sessions opened at the same time on the
same reader. The system ensures the interleaving of APDUs between the
respective sessions.
CRN4: this method returns a Session object to be used to create Channels.
```

Parameter errors

```
None
```

Context errors

```
CRC1: IOError - something went wrong with the communication to the secure
element.
```

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.

The "reader" instance is an element of a Reader[] array, returned by invoking seService.getReaders() method.

```
Test case ID 1: A SE is connected to the Reader. No opened Sessions.
Test case ID 2: A SE is connected to the Reader. One or more Sessions
        already successfully opened.
Test case ID 3: A SE is connected to the Reader.
```

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT →UICC Simulator / SE**	**ISO Response UICC Simulator / SE → DUT**	**API Expectation**	**CRR**
1	First Session opening				
	reader.openSession ()	None	None	Returned Session object is not null. No exception is expected	CRN1 CRN2 CRN4
2	Further Session opening				
	reader.openSession ()	None	None	Returned Session object is not null. No exception is expected	CRN1 CRN2 CRN3 CRN4
3	Second Session opening				
	1. Session s1 = reader.openSession (); **2. Session s2 = reader.openSession ();** **3. s1 != s2;**	None	None	1. No exception is expected. 2. No exception is expected. 3.Session instances s1 and s2 are not equal. No exception is expected.	CRN1 CRN2 CRN3 CRN4

6.3.5 Method: void closeSessions()

(a) Conformance Requirements

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

```
void closeSessions()
```

Normal execution
CRN1: This method closes all the sessions opened on this reader
CRN2: All the channels opened by all this session will be closed.

Parameter errors
None

Context errors
None

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.
The "reader" instance is an element of a Reader[] array, returned by invoking seService.getReaders() method.
Test case ID 1: A SE is connected to the Reader. Session instances s1 and s2 are created.
Test case ID 2: A SE is connected to the Reader. Session instance s1 is created. Three logical channels are opened within 's1'.

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT → UICC Simulator / SE**	**ISO Response UICC Simulator / SE → DUT**	**API Expectation**	**CRR**
1	Close sessions				
	1. reader.closeSessions() **2. s1.isClosed();** **3. s2.isClosed();**	None	None	1. No exception is expected 2. return 'true' 3. return 'true'	CRN1
2	Close sessions and channels				
	reader.closeSessions();	CMD 1-1: MANAGE CHANNEL - P1 = '80'; P2 = the number of the opened channel CMD 1-2: MANAGE CHANNEL - P1 = '80'; P2 = the number of the opened channel CMD 1-3: MANAGE CHANNEL - P1 = '80'; P2 = the number of the opened channel	RESP 1-1: R-APDU - SW '90 00' RESP 1-2: R-APDU - SW '90 00' RESP 1-3: R-APDU - SW '90 00'	No exception is expected.	CRN2

6.4 Class Session

6.4.1 Method: Reader getReader()

(a) Conformance Requirements

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

```
Reader getReader()
```

Normal execution

```
CRN1: Get the reader that provides this session.
```

Parameter errors

```
None
```

Context errors

```
None
```

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance "session" is opened with the selected “reader”.

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT -> UICC Simulator / SE**	**ISO Response UICC Simulator / SE -> DUT**	**API Expectation**	**CRR**
1	**Return the Reader object for a Session instance**				
	session.getReader()	None.	None.	Returned Reader object is not null. No exception is expected.	CRN1
2	**Get the Reader object and compare with the object that provides this session**				
	session.getReader() ==reader	None.	None.	The Reader object returned by getReader() equals to the “reader” instance which provides this session. No exception is expected.	CRN1

6.4.2 Method: byte[] getATR()

(a) Conformance Requirements

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

```
byte[] getATR ()
```

Normal execution

```
CRN1: this method gets the Answer to Reset of this secure
element.
CRN2: if the ATR for this secure element is not available the
returned byte array is null
```

Parameter errors

```
None
```

Context errors

```
None
```

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance "session" is opened with the selected “reader”.

Test case ID2: The UICC Simulator / SE has sent its “ATR” to the DUT.

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT -> UICC Simulator / SE**	**ISO Response UICC Simulator / SE -> DUT**	**API Expectation**	**CRR**
1	**Return the Answer To Reset**				
	session.getATR();	None	None	No exception is expected.	CRN1
2	**Returned Answer To Reset equals to the “ATR” sent during reset**				
	session.getATR()== ATR;	None	None	The Answer to Reset returned by getATR() equals to the “ATR” sent by the UICC Simulator / SE. No exception is expected.	CRN1
3	**Return null in case the Answer To Reset is not available**				
	session.getATR();	None	None	Null is expected to return. No exception is expected.	CRN2

6.4.3 Method: void close()

(a) Conformance Requirements

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

```
void close()
```

Normal execution

```
CRN1: Close the connection with the secure element.
CRN2: This API will close any channels opened by this
application with this secure element.
```

Parameter errors

```
None
```

Context errors

```
None
```

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance "session" is opened with the selected “reader”.

Test case ID 1: One logical channel is opened.
Test case ID 2: Three logical channels are opened

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT -> UICC Simulator / SE**	**ISO Response UICC Simulator / SE -> DUT**	**API Expectation**	**CRR**
1	**Close a session and check the state**				
	session.close();	MANAGE CHANNEL - P1 = ‘80’; P2 = the number of the opened channel	R-APDU - SW ‘90 00’	No exception is expected.	CRN1
2	**Close a session with more logical channels**				
	1. session.close();	CMD 1-1: MANAGE CHANNEL - P1 = ‘80’; P2 = the number of the opened channel CMD 1-2: MANAGE CHANNEL - P1 = ‘80’; P2 = the number of the opened channel CMD 1-3: MANAGE CHANNEL - P1 = ‘80’; P2 = the number of the opened channel	RESP 1-1: R-APDU - SW ‘90 00’ RESP 1-2: R-APDU - SW ‘90 00’ RESP 1-3: R-APDU - SW ‘90 00’	1. No exception is expected.	CRN2

6.4.4 Method: boolean isClosed()

(a) Conformance Requirements

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

```
boolean isClosed()
```

Normal execution

```
CRN1: Tells if this session is closed: if so, isClosed
returns "true"
CRN2: If the session is open it returns false
```

Parameter errors

```
None
```

Context errors

```
None
```

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance "session" is opened with the selected "reader".

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT -> UICC Simulator / SE**	**ISO Response UICC Simulator / SE -> DUT**	**API Expectation**	**CRR**
1	**Check a session already closed**				
	1. session.close(); **2. session.isClosed();**	None	None	1. No exception is expected. 2. "true" is expected to return. No exception is expected.	CRN1
2	**Check an open session**				
	session.isClosed();	None	None	"false" is expected to return. No exception is expected.	CRN2

6.4.5 Method: void closeChannels()

(a) Conformance Requirements

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

```
void closeChannels()
```

Normal execution

```
CRN1: Close any channel opened on this session.
```

Parameter errors

```
None
```

Context errors

```
None
```

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance "session" is opened with the selected “reader”.

Test case ID1: Three logical channels are opened.

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT -> UICC Simulator / SE**	**ISO Response UICC Simulator / SE -> DUT**	**API Expectation**	**CRR**
1	**Close all the channels opened by the session**				
	1. session.closeChannels();	CMD 1-1: MANAGE CHANNEL - P1 = ‘80’; P2 = the number of the opened channel CMD 1-2: MANAGE CHANNEL - P1 = ‘80’; P2 = the number of the opened channel CMD 1-3: MANAGE CHANNEL - P1 = ‘80’; P2 = the number of the opened channel	RESP 1-1: R-APDU - SW ‘90 00’ RESP 1-2: R-APDU - SW ‘90 00’ RESP 1-3: R-APDU - SW ‘90 00’	1. No exception is expected.	CRN1

6.4.6 Method: Channel openBasicChannel(byte[] aid)

(a) Conformance Requirements

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

```
Channel openBasicChannel(byte[] aid)
```

Normal execution

```
CRN1: Get an access to the basic channel, as defined in the
ISO7816-4 specification (the one that has number 0). The
obtained object is an instance of the Channel class.
```

```
CRN2: The AID can be null, which means no SE application is
to be selected on this channel and the default SE application
is used. If the default SE application is not currently
selected on the basic channel then null will be returned.
CRN3: Once this channel has been opened by a device
application, it is considered as "locked" by this device
application, and other calls to this method will return null,
until the channel is closed.
CRN4: Returns null, if the basic channel is locked (e.g. by
the Secure Element or Secure Element drivers).
```

Parameter errors

```
CRP1: IllegalParameterError - if the aid's length is not
within 5 to 16 (inclusive).
```

Context errors

```
CRC1: IOError - if something goes wrong with the
communication to the reader or the secure element.
CRC2: NoSuchElementError – If the AID on the Secure Element
is not available
CRC3: IllegalStateError - if the secure element session is
used after being closed.
CRC4: SecurityError - if the calling application cannot be
granted access to this AID on this session.
```

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance "session" is opened with the selected “reader”.

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT -> UICC Simulator / SE**	**ISO Response UICC Simulator / SE -> DUT**	**API Expectation**	**CRR**
1	**Open a basic channel**				
	1. session.openBasicChannel (AID_TestApp);	CMD 1: SELECT – CLA=’00’; P1=’04’; P2=’00’; Data = ‘AID_TestApp‘	RESP 1: R-APDU - SW ‘90 00’	1. Returned Channel object is not null. No exception is expected.	CRN1
2	**Open a basic channel and check, if the selected SE applet answers**				
	1. session.openBasicChannel (AID_TestApp);	CMD 1: SELECT – CLA=’00’; P1=’04’; P2=’00’; Data = ‘AID_TestApp‘	RESP 1: R-APDU - SW ‘90 00’	1. Returned Channel object is not null. No exception is expected.	CRN1
	2. channel.transmit(Test-APDU1)	CMD 2: C-APDU – any case 4 type apdu – e.g.: ’00 10 01 00 04 01 02 03 04 00’	RESP 2: R-APDU - Data = ’01 02 03 04’; SW ‘90 00’	2. Returned Response equals to ‘R-APDU’ - Data = ’ 01 02 03 04’; SW ‘90 00’. No exception is expected.	
3	**Open a basic channel with the default SE applet**				

	1. session.openBasicChannel (null);	CMD 1: SELECT – CLA='00'; P1='04'; P2='00'	RESP 1: R-APDU – SW '90 00'	1. Returned Channel object is not null. No exception is expected.	CRN2
4	**Open a basic channel with the default SE applet and check, if the applet answers**				
	1. session.openBasicChannel (null);	CMD 1: SELECT – CLA='00'; P1='04'; P2='00';	RESP 1: R-APDU - SW '90 00'	1. Returned Channel object is not null. No exception is expected.	CRN2
	2. channel.transmit(Test-APDU1);	CMD 2: C-APDU – any case 4 type apdu – e.g.: '00 10 01 00 04 01 02 03 04 00'	RESP 2: R-APDU - Data = '01 02 03 04'; SW '90 00'	2. Returned Response equals to 'R-APDU' - Data = '01 02 03 04'; SW '90 00'. No exception is expected.	
5	**Open a basic channel with the default SE applet when the default applet is not currently selectable**				
	1. session.openBasicChannel (AID_TestApp);	CMD 1: SELECT – CLA='00'; P1='04'; P2='00'; Data = 'AID_TestApp'	RESP 1: R-APDU - SW '90 00'	1. Returned Channel object is not null. No exception is expected	CRN2
	2. channel.close();	CMD 2: None	RESP 2: None	2. No exception is expected	
	3. session.openBasicChannel (null);	CMD 3: None	RESP 3: None	3. Returned Channel object is null. No exception is expected.	
6	**Open a basic channel when it is locked by an application**				
	1. session.openBasicChannel (AID_TestApp);	CMD 1: SELECT – CLA='00'; P1='04'; P2='00'; Data = 'AID_TestApp'	RESP 1: R-APDU - SW '90 00'	1. Returned Channel object is not null. No exception is expected.	CRN3
	2. session.openBasicChannel (AID_TestApp_multiselectable);	CMD 2: No ISO command is expected. (the channel is locked by the API)	RESP 2: No Response.	2. Returned Channel object is null. No exception is expected.	
7	**Open a basic channel when it is locked by default**				
	session.openBasicChannel (AID_TestApp);	None	None.	Returned Channel object is null. No exception is expected.	CRN4
8	**The length of the AID is less than 5**				
	session.openBasicChannel (AID_Illegal_1);	None	None	IllegalParameterError is expected.	CRP1
9	**The length of the AID is more than 16**				
	session.openBasicChannel (AID_Illegal_2);	None	None	IllegalParameterError is expected.	CRP1
10	**Communication problem with the Secure Element**				
	session.openBasicChannel (AID_TestApp);	SELECT – CLA='00'; P1='04'; P2='00'; Data = 'AID_TestApp'	No R-APDU is returned.	IOError is expected.	CRC1

11	**The AID is not available on the Secure Element**				
	session.openBasicChannel (AID_nonexisting);	SELECT – CLA='00'; P1='04'; P2='00'; Data = 'AID_nonexisting '	R-APDU – SW '6A 82'	NoSuchElementError is expected.	CRC2
12	**Open a basic channel, when session is already closed**				
	1. session.close(); **2. session.openBasicChannel (AID_TestApp);**	None	None	1. No exception is expected. 2. IllegalStateError is expected.	CRC3
13	**The application opening the basic channel has no access to the selected SE applet**				
	session.openBasicChannel (AID_TestApp_accessdenied);	Out of the scope of the test case.	Out of the scope of the test case.	SecurityError is expected.	CRC4

6.4.7 Method: Channel openLogicalChannel(byte[] aid)

(a) Conformance Requirements

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

```
Channel openLogicalChannel(byte[] aid)
```

Normal execution

```
CRN1: Open a logical channel with the secure element,
selecting the application represented by the given AID.
CRN2: if the AID is null, then the default application shall
be used.
CRN3: It's up to the secure element to choose which logical
channel will be used.
CRN4: return null if secure element is unable to provide a new
logical channel
```

Parameter errors

```
CRP1: IllegalParameterError - if the aid's length is not
within 5 to 16 (inclusive).
```

Context errors

```
CRC1: IOError - if something goes wrong with the communication
to the reader or the secure element. (e.g. if the AID is not
available)
CRC2: NoSuchElementError - if the AID on the Secure Element is
not available or a logical channel is already open to a non-
multiselectable Applet.
CRC3: IllegalStateError - if the secure element session is
used after being closed.
CRC4: SecurityError - if the calling application cannot be
granted access to this AID on this session.
```

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance "session" is opened with the selected “reader”.

Test case ID 5: The maximum number of logical channels supported by the UICC Simulator / SE is already opened.

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT -> UICC Simulator / SE**	**ISO Response UICC Simulator / SE -> DUT**	**API Expectation**	**CRR**
1	**Open a logical channel**				
	1. session.openLogicalChannel (AID_TestApp);	CMD 1-1: MANAGE CHANNEL - P1=’00’, P2=‘00‘, Le=’01’ CMD 1-2: SELECT – CLA	RESP 1-1: R-APDU - Data: Channel Number; SW ‘90 00’ RESP 1-2: R-APDU - SW ‘90 00’	1. Returned Channel object is not null. No exception is expected.	CRN1 CRN3

		contains the Channel Number returned by the card in RESP 1-1; P1='04'; P2='00'; Data = 'AID_TestApp'			
2	**Open a logical channel and check, if the selected SE applet answers**				
	1. session.openLogicalChannel (AID_TestApp); **2. channel.transmit(Test_C-APDU)**	CMD 1-1: MANAGE CHANNEL - P1='00', P2='00', Le='01' CMD 1-2: SELECT – CLA contains the Channel Number returned by the card in RESP 1-1; P1='04'; P2='00'; Data = 'AID_TestApp' CMD 2: C-APDU – any case 4 type apdu – e.g.: '01 10 01 00 04 01 02 03 04 00'	RESP 1-1: R-APDU - Data: Channel Number; SW '90 00' RESP 1-2: R-APDU - SW '90 00' RESP 2: R-APDU - Data = '01 02 03 04'; SW '90 00'	1. Returned Channel object is not null. No exception is expected. 2. Returned Response equals to 'R-APDU' - Data = '01 02 03 04'; SW '90 00'. No exception is expected.	CRN1
3	**Open a logical channel with the default SE applet**				
	1. session.openLogicalChannel (null);	CMD 1: MANAGE CHANNEL - P1='00', P2='00', Le='01'	RESP 1: R-APDU - Data: Channel Number; SW '90 00'	1. Returned Channel object is not null. No exception is expected.	CRN2
4	**Open a logical channel with the default SE applet and check, if the applet answers**				
	1. session.openLogicalChannel (null); **2. channel.transmit(Test-APDU1);**	CMD 1: MANAGE CHANNEL - P1='00', P2='00', Le='01' CMD 2: C-APDU – any case 4 type apdu – e.g.: '01 10 01 00 04 01 02 03 04 00'	RESP 1: R-APDU - Data: Channel Number; SW '90 00' RESP 2: R-APDU - Data = '01 02 03 04'; SW '90 00'	1. Returned Channel object is not null. No exception is expected. 2. Returned Response equals to 'R-APDU' - Data = '01 02 03 04 '; SW '90 00'. No exception is expected.	CRN2
5	**Open a logical channel, when no new logical channel is available**				
	1. session.openLogicalChannel (AID_TestApp);	CMD 1: MANAGE CHANNEL - P1='00', P2='00', Le='01'	RESP 1: R-APDU – SW '68 81'	1. Returned Channel object is null. No exception is expected.	CRN4
6	**The length of the AID is less than 5**				
	session.openLogicalChannel (AID_Illegal_1);	None	None	IllegalParameterError is expected.	CRP1
7	**The length of the AID is more than 16**				
	session.openLogicalChannel (AID_Illegal_2);	None	None	IllegalParameterError is expected.	CRP1
8	**Communication problem with the Secure Element**				
	1. session.openLogicalChannel (AID_TestApp);	CMD 1-1: MANAGE CHANNEL - P1='00', P2='00', Le='01' CMD 1-2: SELECT – CLA contains the Channel Number	RESP 1-1: R-APDU - Data: Channel Number; SW '90 00' RESP 1-2: No R-APDU is returned.	1. IOError is expected.	CRC1

		returned by the card in RESP 1-1; P1='04'; P2='00'; Data = 'AID_TestApp'			
9	**The AID is not available on the Secure Element**				
	1. session.openLogicalChannel (AID_nonexisting);	CMD 1/1: MANAGE CHANNEL - P1='00', P2='00', Le='01' CMD 1/2: SELECT – CLA contains the Channel Number returned by the card in RESP 1; P1='04'; P2='00'; Data = 'AID_nonexisting '	RESP 1-1: R-APDU - Data: Channel Number; SW '90 00' RESP 1-2: R-APDU – SW '6A 82'	1. NoSuchElementError is expected.	CRC2
10	**A logical channel is already open to the non-multiselectable SE Applet**				
	1. session.openLogicalChannel (AID_TestApp); **2. session.openLogicalChannel (AID_TestApp);**	CMD 1-1: MANAGE CHANNEL - P1='00', P2='00', Le='01' CMD 1-2: SELECT – CLA contains the Channel Number returned by the card in RESP 1-1; P1='04'; P2='00'; Data = 'AID_TestApp' CMD 2-1: MANAGE CHANNEL - P1='00', P2='00', Le='01' CMD 2-2: SELECT – CLA contains the Channel Number returned by the card in RESP 2-1; P1='04'; P2='00'; Data = 'AID_TestApp	RESP 1-1: R-APDU - Data: Channel Number; SW '90 00' RESP 1-2: R-APDU - SW '90 00' RESP 2-1: R-APDU - Data: Channel Number; SW '90 00' RESP 2-2: R-APDU - SW '6A 82'	1. No exception is expected. 2. NoSuchElementError is expected.	CRC2
11	**Open a logical channel, when session is already closed**				
	1. session.close(); **2. session.openLogicalChannel (AID_TestApp);**	none	none	1. No exception is expected. 2. IllegalStateError is expected.	CRC3
12	**The application opening the logical channel has no access to the selected SE applet**				
	session.openLogicalChannel (AID_TestApp_accessdenied);	Out of the scope of the test case.	Out of the scope of the test case.	SecurityError is expected.	CRC4

6.5 Class: Channel

Instances of this class represent an ISO7816-4 channel opened to a secure element. It can be either a logical channel or the default channel.
They can be used to send APDUs to the secure element. Channels are opened by calling the Session.openBasicChannel(byte[]) or Session.openLogicalChannel(byte[]) methods.

6.5.1 Method: void close()

(a) Conformance Requirements

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

```
void close()
```

Normal execution

```
CRN1: the close() method closes the channel to the Secure
Element
CRN2: if the channel is the basic channel, then it becomes
available again
CRN3: if the channel is already closed, the method is ignored
CRN4: The close() method shall wait for completion of any
pending transmit(byte[] command) before closing the channel.
```

Parameter errors

```
None
```

Context errors

```
None
```

(b) Initial Conditions

```
SEService Object has been created and the isConnected()
method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance
"session" is opened with the selected "reader".
Test cases ID1,ID3,ID4: a logical channel with "AID_Test_App"
is already open.
Test case ID2: a basic channel with "AID_Test_App" is already
open.
```

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation** **DUT → UICC Simulator/SE**	**ISO Response** **UICC Simulator/SE → DUT**	**API Expectation**	**CRR**
1	**Close an open logical channel**				
	1. Channel.close();	CMD 1-1: MANAGE CHANNEL (P1='80')	RESP 1-1: R-APDU - SW '90 00'	1. No exception is expected.	CRN1

2	**Close an open basic channel**				
	1. Channel.close();	CMD 1-1 None	RESP 1-1 None	1. No exception is expected.	CRN2
3	**Close an already closed channel**				
	1. Channel.close(); **2. Channel.close();**	CMD-1-1: MANAGE CHANNEL (P1='80') CMD 2-1: None	RESP 1-1: R-APDU - SW '90 00' RESP 2-1: None	1. No exception is expected. 2. No exception is expected.	CRN3
4	**'Close' method shall wait for an ongoing 'transmit()'**				
	1. *Thread1: Transmit Test-APDU2* **Channel.transmit(Test-APDU2)** *Thread2 sleep/wait for 1 seconds* *2.Thread2:* **Channel.close();**	CMD 1-1: APDU (01 10 01 00 04 01 02 03 04 00) CMD 2-1: MANAGE CHANNEL (P1='80')	RESP 1-1: R-APDU '01 02 03 04 '- SW '90 00' RESP 2-1: R-APDU - SW '90 00'	1. byte[]= {90,00} 2. close returns after transmit has been completed No exception is expected.	CRN4

6.5.2 Method: boolean isBasicChannel()

(a) Conformance Requirements

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

```
boolean isBasicChannel()
```

Normal execution

```
CRN1: this method returns true if the channel is the basic channel
CRN2: this method returns false if the channel is a logical channel
```

Parameter errors

```
None
```

Context errors

```
None
```

(b) Initial Conditions

```
SEService Object has been created and the isConnected()
method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance
"session" is opened with the selected "reader".
Test case ID1: a basic channel with "AID_Test_App" is already
open.
Test case ID2: a logical channel with "AID_Test_App" is
already open.
```

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT → UICC Simulator/SE**	**UICC Simulator - ISO Response UICC Simulator/SE → DUT**	**API Expectation**	**CRR**
1	**Check for an open basic channel**				
	1. Channel.isBasicChannel();	CMD1-1: None	RESP 1-1: None	1. Return 'true'.	CRN1
2	**Check for an open logical channel**				
	1. Channel.isBasicChannel();	CMD 1-1: None	RESP 1-1: None	1. Return 'false'	CRN2

6.5.3 Method: boolean isClosed()

(a) Conformance Requirements

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

```
boolean isClosed ()
```

Normal execution

```
CRN1: this method returns true if the channel is closed
CRN2: this method returns false if the channel is open
```

Parameter errors

```
None
```

Context errors

```
None
```

(b) Initial Conditions

```
SEService Object has been created and the isConnected()
method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance
"session" is opened with the selected "reader".
For all test cases: a logical channel with "AID_Test_App" is
already open.
```

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT → UICC Simulator/SE**	**UICC Simulator - ISO Response UICC Simulator/SE → DUT**	**API Expectation**	**CRR**
1	**Check for an open channel**				
	1. Channel.isClosed();	CMD 1-1: None	CMD 1-1: None	1. Return 'false'	CRN1
2	**Check for a closed channel**				
	1. Channel.close(); **2. Channel.isClosed();**	CMD 1-1: MANAGE CHANNEL (P1='80') CMD 2-1: None	RESP 1-1: R-APDU - SW '90 00' RESP 2-1: None	1. No exception is expected 2. Return 'true'	CRN2

6.5.4 Method: byte[] getSelectResponse()

(a) Conformance Requirements

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

```
byte[] getSelectResponse()
```

Normal execution

```
CRN1: Returns the data as received from the application
select command inclusively the status word.
CRN2: The returned byte array contains the data bytes in the
following order:
[<first data byte>, …, <last data byte>, <sw1>, <sw2>]
CRN3: The returned byte array contains only the status word
if the application select command has no data returned.
CRN4: Null is returned if the application select command has
not been performed or the selection response can not be
retrieved by the reader implementation.
```

Parameter errors

```
None
```

Context errors

None

(b) Initial Conditions

SEService Object has been created and the isConnected() method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance "session" is opened with the selected "reader".
Test case ID1: a logical channel with "AID_Test_App_1" is already open.
Test cases ID2: a logical channel with "AID_Test_App" is already open.
Test case ID3: a logical channel with "null" AID is already open.

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT → UICC Simulator/SE**	**UICC Simulator - ISO Response UICC Simulator/SE → DUT**	**API Expectation**	**CRR**
1	**Return data and Status Word from an application select command**				
	1. Channel.getSelectResponse()	CMD 1-1: None	RESP 1-1: None	1. byte[ ]= TestApp1_SResp	CRN1, CRN2
2	**Return only the Status Word from an application select command**				
	1. Channel.getSelectResponse()	CMD 1-1: None	RESP 1-1: None	1. byte[ ]= {90,00}	CRN1, CRN3
3	**Return null in case the application select command is not performed**				
	1. Channel.getSelectResponse()	CMD 1-1: None	RESP 1-1:None	1. Return 'null'	CRN1, CRN4

6.5.5 Method: Session getSession()

(a) Conformance Requirements

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

```
Session getSession()
```

Normal execution

```
CRN1: this method returns the session object this channel is
bound to
```

Parameter errors

```
None
```

Context errors

```
None
```

(b) Initial Conditions

```
SEService Object has been created and the isConnected()
method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance
"session" is opened with the selected "reader".
A logical channel with "AID_Test_App" is already open.
```

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT → UICC Simulator/SE**	**UICC Simulator - ISO Response UICC Simulator/SE → DUT**	**API Expectation**	**CRR**
1	**Return the Session object for a Channel instance**				
	1. Session == Channel.getSession ()	CMD 1-1: None	RESP 1-1: None	1. The Session object returned by getSession() is not null and equals to the "session" instance created in initial conditions.	CRN1,

6.5.6 Method: byte[] transmit(byte[] command)

(a) Conformance Requirements

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

```
byte[] transmit(byte[] command)
```

Normal execution

```
CRN1: Transmit an APDU command as per ISO7816-4 to the secure
element. The underlying layers can generate as many TPDUs as
necessary to transport this APDU. The transport part is
invisible from the application.
CRN2: The system ensures the synchronization between all the
concurrent calls to this method. The entire APDU
communication to this SE is locked to the APDU.
CRN3: The system ensures that only one APDU will be sent at a
time, irrespective of the number of TPDUs that might be
required to transport it to the SE.
CRN4: The channel information in the class byte in the APDU
will be ignored: the system will add any required information
to ensure the APDU is transported on this channel.
```

Parameter errors

```
CRP1: IllegalParameterError – if the parameter command is
null.
CRP2: IllegalParameterError – if a MANAGE_CHANNEL command is
sent to the SE
CRP3: IllegalParameterError – if a SELECT by DF Name (p1=04)
command is sent to the SE
```

Context errors

```
CRC1: IOError - if there is a communication problem to the
reader or the Secure Element.
CRC2: IllegalStateError - if the channel is closed at the
time of invocation of this method
CRC3: IllegalParameterError – if the command byte array is
less than 4 bytes long
CRC4: SecurityError - if the command is filtered by the
security policy.
```

(b) Initial Conditions

```
SEService Object has been created and the isConnected()
method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance
"session" is opened with the selected "reader".
Test case ID1: a basic channel with "AID_Test_App" is already
open.
Test cases ID2 and ID5 to ID11: a logical channel with
"AID_Test_App" is already open.
Test cases ID3: a logical channel with "AID_Test_App" is
already open. SE shall return logical channel number 1.

Test case ID 4: The two channels are created in two different
sessions
```

(c) Test Procedure

Test case

ID	API Description	ISO Command Expectation DUT → UICC Simulator/SE	UICC Simulator - ISO Response UICC Simulator/SE → DUT	API Expectation	CRR
1	**Transmit an APDU on Basic Channel**				
	1. Channel.transmit(Test-APDU1); **2. Channel.transmit(Test-APDU4);** **3. Channel.transmit(Test-APDU5);** **4. Channel.transmit(Test-APDU6);**	CMD 1-1: APDU (’00 10 01 00 04 01 02 03 04 00’) CMD 2-1: APDU (’00 30 00 00’) CMD 3-1: APDU (’00 40 00 00 00’) CMD 4-1: APDU (’00 50 01 00 04 01 02 03 04’)	RESP 1-1: R-APDU – ’01 02 03 04’ SW ‘90 00’ RESP 2-1: R-APDU – SW ‘90 00’ RESP 3-1: R-APDU – ’01 02 03 04’ SW ‘90 00’ RESP 4-1: R-APDU SW ‘90 00’	1. byte[ ]= {’01, 02, 03, 04, 90,00} 2. byte[ ]= {’ 90,00} 3. byte[ ]= {’01, 02, 03, 04, 90,00} 4. byte[ ]= {’ 90,00}	CRN1
2	**Transmit an APDU on Logical Channel**				
	1. Channel.transmit(Test-APDU1); **2. Channel.transmit(Test-APDU4);** **3. Channel.transmit(Test-APDU5);** **4. Channel.transmit(Test-APDU6);**	CMD 1-1: APDU (’01 10 01 00 04 01 02 03 04 00’) CMD 2-1: APDU (’01 30 00 00’) CMD 3-1: APDU (’01 40 00 00 00’) CMD 4-1: APDU (’01 50 01 00 04 01 02 03 04’)	RESP 1-1: R-APDU – ’01 02 03 04’ SW ‘90 00’ RESP 2-1: R-APDU – SW ‘90 00’ RESP 3-1: R-APDU – ’01 02 03 04’ SW ‘90 00’ RESP 4-1: R-APDU SW ‘90 00’	1. byte[ ]= {’01, 02, 03, 04, 90, 00} 2. byte[ ]= {90,00} 3. byte[ ]= {01, 02, 03, 04, 90,00} 4. byte[ ]= {90,00}	CRN1
3	**Transmit an APDU with a wrong channel number**				

	Send an Test_APDU1 with different channel number. E.g. with number channel = 2 → CLA = '02': *1.***Channel.transmit('02100100040102030 4 00');**	CMD 1-1: APDU ('01 10 01 00 04 01 02 03 04 00')	RESP 1-1: R-APDU - '01 02 03 04' SW '90 00'	1. byte[ ]= {'01, 02, 03, 04, 90,00}	CRN1, CRN4
4	**Synchronization between concurrent calls**				
	1. Channel1 = Session1.openLogicalChannel (AID_TestApp1); *start new Thread2:* **2. Channel2 = Session2.openLogicalChannel (AID_TestApp2);** *Thread 1:* **3. Channel1.transmit(Test_APDU2);** *Thread2: wait – 0,5 s* **4. Channel2.transmit(Test_APDU2);**	CMD 1-1: MANAGE CHANNEL (P1='00', P2='00' ,Le='01') CMD 1-2: SELECT – CLA with Channel Number =1 (returned by the card in RESP 2-1); Data = 'AID_TestApp1' CMD 2-1: MANAGE CHANNEL (P1='00', P2='00' ,Le='01') CMD 2-2: SELECT – CLA with Channel Number =2 (returned by the card in RESP 2-1); Data = 'AID_TestApp2' Both APDUs are transmitted: CMD 3-1: APDU ('01 10 02 00 04 01 02 03 04 00') CMD 4-1: APDU ('02 10 02 00 04 05 06 07 08 00')	RESP 2-1: R-APDU - Data: Channel Number=1; SW '90 00' RESP 2-2: R-APDU - SW '90 00' RESP 2-1: R-APDU - Data: Channel Number=2; SW '90 00' RESP 2-2: R-APDU - SW '90 00' RESP 3-1: R-APDU – '01 02 03 04' SW '90 00' RESP 4-1: R-APDU – '05 06 07 08' SW '90 00'	1. Returned Channel1 object is not null. No exception is expected. 2. Returned Channel2 objects is not null. No exception is expected. 3. byte[ ]= {01, 02, 03, 04, 90,00} 4. byte[ ]= {05, 06, 07, 08, 90,00}	CRN2, CRN3
5	**Null parameter command**				
	1. Channel.transmit(null);	CMD 1-1:None	RESP 1-1: None	`1.IllegalParameterError`	CRP1

6	**MANAGE CHANNEL as parameter command**				
	1. Channel.transmit(MANAGE CHANNEL APDU);	CMD 1-1: None	RESP 1-1: None	1.IllegalParameterError	CRP2
7	**SELECT BY DF NAME as parameter command**				
	1. Channel.transmit(SELECT by DF name);	CMD 1-1: None	RESP 1-1: None	1. IllegalParameterError	CRP3
8	**Communication problem with the Secure Element**				
	Deactivate ISO communication with the Device *1.* **Channel.transmit(Test_APDU1);**	CMD 1-1: : APDU ('01 10 02 00 04 01 02 03 04 00')	RESP 1-1: None	1. IOError	CRC1
9	**Transmit an APDU when the channel is closed**				
	1. Channel.close(); **2.Channel.transmit(Test_APDU1);**	CMD 1-1: MANAGE CHANNEL (P1='80') CMD 2-1: None	RESP 1-1: R-APDU - SW '90 00' RESP 2-1: None	1. No exception is expected. 2. IllegalStateError	CRC2
10	**Command parameter shorter than 4 bytes**				
	1. Channel.close(); *Transmit a dummy command to the application with only 3 bytes:* ***2.*** **Channel.transmit('011500');**	CMD 1-1: MANAGE CHANNEL (P1='80') CMD 2-1: None	RESP 1-1: R-APDU - SW '90 00' RESP 2-1 None	1. No exception is expected. 2. IllegalParameterError	CRC3
11	**Access Control rule does not allow the sending of this APDU**				
		CMD 1-1: None			CRC4

	1. Channel.transmit(Test-APDU3);		RESP 1-1: None	1. SecurityError is expected.	

6.5.7 Method: boolean[] selectNext()

(a) Conformance Requirements

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

```
boolean selectNext()
```

Normal execution

```
CRN1: Performs a selection of the next Applet on this channel
that matches to the partial AID specified in the
openBasicChannel(byte[] aid) or openLogicalChannel(byte[]
aid) method. This mechanism can be used by a device
application to iterate through all Applets matching to the
same partial AID. If selectNext() returns true a new Applet
was successfully selected on this channel.
CRN2: The implementation of the underlying SELECT command
within this method shall use the same values as the
corresponding openBasicChannel(byte[] aid) or
openLogicalChannel(byte[] aid)  command with the option:
P2='02'  (Next occurrence)
CRN3: If no further Applet exists with matches to the partial
AID this method returns false and the already selected Applet
stays selected.
CRN4: The select response stored in the Channel object shall
be updated with the APDU response of the SELECT command.
```

Context errors

```
CRC1: IOError - if there is a communication problem to the
reader or the Secure Element.
CRC2: OperationNotSupportedError - if this operation is not
supported.
CRC3: IllegalStateError - if the Secure Element is used after
being closed.
```

(b) Initial Conditions

```
SEService Object has been created and the isConnected()
method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance
"session" is opened with the selected "reader".
Test cases ID1,ID3,ID5,ID6,ID7: a logical channel with
"Partial_AID_1" is already open.
Test cases ID2,ID4: a logical channel with "Partial_AID_2" is
already open.
```

(c) Test Procedure

Test case					
ID	**API Description**	**ISO Command Expectation DUT → UICC Simulator/SE**	**UICC Simulator - ISO Response UICC Simulator/SE → DUT**	**API Expectation**	**CRR**
1	**Next Applet matches with partial AID**				
	1. Channel.selectNext();	CMD 1-1: SELECT – CLA with Channel Number =1 ; P2='02' (Next occurrence); Data = 'Partial_AID_1'	RESP 1-1: R-APDU - SW '90 00'	1. Return 'true'	CRN1, CRN2
2	**No other Applet does not match with partial AID**				
	1. Channel.selectNext();	CMD 1-1: SELECT – CLA with Channel Number =1 ; P2='02' (Next occurrence) ; Data = 'Partial_AID_2'	RESP 1-1: R-APDU - SW '6A 82'	1. Return 'false'	CRN2, CRN3
3	**Check select response is updated**				

	1. response1 = Channel.getSelectResponse() **2. Channel.selectNext();** **3. response2 = Channel.getSelectResponse()**	CMD 1-1: None CMD 2-1: SELECT – CLA with Channel Number =1 ; P2='02' (Next occurrence); Data = 'Partial_AID_1' CMD 3-1: None	RESP 1-1: None RESP 2-1: R-APDU -AID SW '90 00' RESP 3-1: None	1. response1 = {AID 90. 00} 2. Return 'true' 3. response2 = { AID 90. 00}	CRN1, CRN2, CRN4
4	**Check select response is updated in case slectNext() fails**				
	1. response1 = Channel.getSelectResponse() **2. Channel.selectNext();** **3. response2 = Channel.getSelectResponse()**	CMD 1-1: None CMD 2-1: SELECT – CLA with Channel Number =1 ; P2='02' (Next occurrence); Data = 'Partial_AID_2' CMD 3-1: None	RESP 1-1: None RESP 2-1: R-APDU - SW '6A 82' RESP 3-1: None	1. response1 = {AID 90. 00} 2. Return 'false' 3. response2 = null	CRN1, CRN2, CRN4
5	**Communication problem with the Secure Element**				
	Deactivate ISO communication with the Device **1. Channel.selectNext();**	CMD 1-1: SELECT – CLA with Channel Number =1 ; P2='02' (Next occurrence); Data = 'Partial_AID_1'	RESP 1-1: None	`1. IOError`	CRC1
6	**Operation not supported**				
	1. Channel.selectNext();	CMD 1-1: SELECT – CLA with Channel Number =1 ; P2='02' (Next occurrence); Data = 'Partial_AID_1'	RESP 1-1: R-APDU - SW '6A 81'	1. `OperationNotSupportedError`	CRC2

7	**selectNext() when the channel is closed**				
	1. Channel.close(); **2. Channel.selectNext();**	CMD 1-1: MANAGE CHANNEL (P1='80') CMD 2-1: None	RESP 1-1: R-APDU - SW '90 00' RESP 2-1: None	2. IllegalStateError	CRC3

7. History

Table 7-1: History

Version	Date	Author	Comment
0.9	13.09.2013	SIMalliance	Initial Release 0.9

Annex A (normative): None tested requirements

The requirements that are not tested in the current version of the specification listed in table A.1. The section index referenced in table A.1 is the index used in this specification.

Table A.1

Requirement	Class	Index	Method
CRC1: IOError - something went wrong with the communication to the secure element. (e.g. no SE connected or no more Session available)	Reader	6.3.4	Session openSession()
CRN1: This method closes all the sessions opened on this reader	Reader	6.3.5	void closeSessions()