

# **Interoperability Specification for ICCs and Personal Computer Systems**

*Part 10 IFDs with Secure PIN Entry Capabilities  
Supplement - IFDs with Feature Capabilities*

*Gemalto*

*HID Global*

*Identive*

*Oracle America*

*Vasco Data Security*

**Revision 2.02.01**

**Nov 2012**

**Copyright © 1996–2012 Apple, Gemalto, Hewlett-Packard, IBM, Infineon, Ingenico, Microsoft, HID Global, NXP Semiconductors, Oracle, Siemens, Sun Microsystems, Toshiba, Vasco Data Security. All rights reserved.**

**INTELLECTUAL PROPERTY DISCLAIMER**

**THIS SPECIFICATION IS PROVIDED “AS IS” WITH NO WARRANTIES WHATSOEVER INCLUDING ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION, OR SAMPLE. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED OR INTENDED HEREBY.**

**GEMALTO, HEWLETT-PACKARD, HID OMNIKEY, IBM, INFINEON, INGENICO, MICROSOFT, NXP SEMICINDUCTORS, SIEMENS, SUN MICROSYSTEMS AND TOSHIBA DISCLAIM ALL LIABILITY, INCLUDING LIABILITY FOR INFRINGEMENT OF PROPRIETARY RIGHTS, RELATING TO IMPLEMENTATION OF INFORMATION IN THIS SPECIFICATION. GEMALTO, HEWLETT-PACKARD, HID GLOBAL, IBM, INFINEON, INGENICO, MICROSOFT, NXP SEMICINDUCTORS, ORACLE, SIEMENS, SUN MICROSYSTEMS, TOSHIBA AND VASCO DATA SECURITY DO NOT WARRANT OR REPRESENT THAT SUCH IMPLEMENTATION(S) WILL NOT INFRINGE SUCH RIGHTS.**

Windows, Windows NT, Windows 2000, Windows 2003, Windows 2008, Windows XP, Windows Vista and Windows 7 are trademarks and Microsoft and Win32 are registered trademarks of Microsoft Corporation. PS/2 is a registered trademark of IBM Corp. JAVA is a registered trademark of Oracle Corporation. All other product names are trademarks, registered trademarks, or servicemarks of their respective owners.

## **Revision History**

<b>Revision</b>	<b>Issue Date</b>	<b>Comments</b>
02.02.01	Nov, 2012	First Draft

---

## Contents

---

<b>1</b>	<b>SYSTEM ARCHITECTURE</b>	<b>1</b>
<b>2</b>	<b>DEFINITION OF FEATURES</b>	<b>2</b>
2.1	General Description	2
2.2	Feature Execution	2
2.2.1	Feature Execution by Pseudo-APDU	3
2.3	Get List Of Features (GET_FEATURE_REQUEST)	4
2.3.1	GET_FEATURE_REQUEST by Pseudo-APDU	4
<b>3</b>	<b>FEATURES</b>	<b>5</b>
3.1	FEATURE_VERIFY_PIN_START	5
3.2	FEATURE_VERIFY_PIN_FINISH	5
3.3	FEATURE_MODIFY_PIN_START	5
3.4	FEATURE_MODIFY_PIN_FINISH	6
3.5	FEATURE_GET_KEY_PRESSED	6
3.6	FEATURE_VERIFY_PIN_DIRECT	6
3.7	FEATURE_MODIFY_PIN_DIRECT	6
3.8	FEATURE_MCT_READER_DIRECT	7
3.9	FEATURE_MCT_UNIVERSAL	7
3.10	FEATURE_IFD_PIN_PROPERTIES	7
3.11	FEATURE_ABORT	8
3.12	FEATURE_SET_SPE_MESSAGE	8
3.13	FEATURE_VERIFY_PIN_DIRECT_APP_ID	8
3.14	FEATURE_MODIFY_PIN_DIRECT_APP_ID	9
3.15	FEATURE_WRITE_DISPLAY	9
3.16	FEATURE_GET_KEY	9
3.17	FEATURE_IFD_DISPLAY_PROPERTIES	9

<b>3.18</b>	<b>FEATURE_GET_TLV_PROPERTIES</b>	<b>10</b>
<b>3.19</b>	<b>FEATURE_CCID_ESC_COMMAND</b>	<b>10</b>
<b>3.20</b>	<b>FEATURE_EXECUTE_PACE</b>	<b>10</b>
	<b>ABBREVIATIONS</b>	<b>11</b>
<b>4</b>	<b>REFERENCES</b>	<b>12</b>

# 1 System Architecture

This documents deals with feature readers and their integration into the PC/SC architecture.

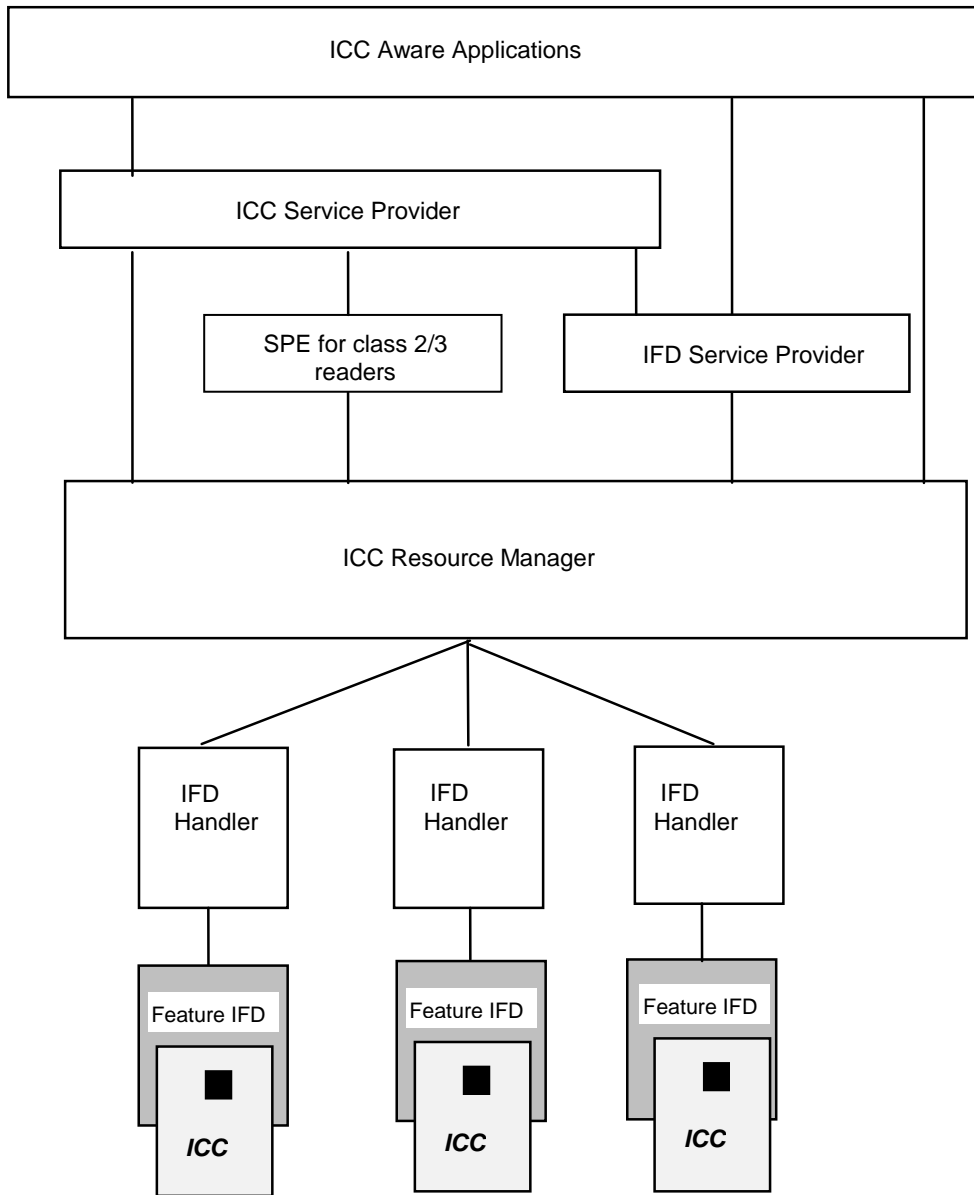


Figure 1- General Architecture

## 2 Definition of Features

### 2.1 General Description

Chipcard readers are becoming more intelligent: features as secure PIN entry are becoming very important. This part of the PC/SC specifications defines general features of the subsystem.

A feature is defined by its *Feature Number* and the accompanying *Feature Command Data* and *Feature Response Data*.

An application can query the subsystem which features are supported. In the response, the application receives a list of *Feature Numbers*; this list represents all supported features on the subsystem.

A feature is executed on the IFD by commanding it through a Pseudo-APDU (PPDU)

### 2.2 Feature Execution

A certain feature is represented by its *Feature Number*, the input data for a feature is presented in the accompanying *Feature Command Data*.

The result of the feature's execution is presented in the *Feature Response Data*.



## 2.2.1 Feature Execution by Pseudo-APDU

A feature is commanded by special APDU's, called Pseudo APDU (PPDU). The Pseudo-APDU command is in a data format which has much resemblance with an APDU for cards:

command header				command body	
CLA	INS	P1	P2	Lc	Command
'FF'	'C2'	'01'	<i>Feature Number</i>	Lc	<i>Feature Command Data</i>

This Pseudo-APDU is defined as a command header (CLA/INS/P1/P2) and an optional command body, according [5], chapter 12.

Any valid Pseudo-APDU command will always generate a response:

response data	response status SW1/SW2	description
<i>Feature Response Data</i>	90 00	Feature executed successful, <i>Feature Response Data</i> is present
-empty-	6A 86	Incorrect value for P2 (requested feature not present)

This response is defined as an optional 'response data' part plus a 2-byte status code, in line with [5], chapter 12.

This Pseudo-APDU is represented by a number of sequential bytes (a buffer), this shall be exchanged by means of the Transmit method (see [6]), as follows:

```
RESPONSECODE Transmit(
    IN SCARD_IO_HEADER SendPci           // Send protocol structure
    IN BYTE[] SendBuffer                 // Data buffer for send data
    IN OUT SCARD_IO_HEADER RecvPci       // Receive protocol structure
    IN OUT BYTE[] RecvBuffer             // Data buffer for receive data
    OUT DWORD RecvLength                 // Length of received data
)
```

The `SendPci` must contain the protocol structure of the current inserted card.

The `SendBuffer` contains the *Feature Command Data*.

The `RecvPci` contains the protocol structure used to communicate.

The `RecvBuffer` will contain the *Feature Response Data* when the subsystem has successfully executed this command. A successful execution is marked when the response status SW1/SW2 has value '90 00'.

A note on endianness.

For a successful use of the Pseudo-APDU interface, values in the structures of [ref 7] representing an integer greater than 1 byte are defined to be ordered low-byte first (little-endian byte-order). Remind that in [ref 7] the byte ordering is decided by machine architecture.



## 2.3 Get List Of Features (GET\_FEATURE\_REQUEST)

A reader (subsystem) may contain a certain number of features. The application shall be able to request the actual supported feature(s) of the current subsystem.

The following features are currently defined:

Feature	Feature Number
FEATURE_VERIFY_PIN_START	0x01
FEATURE_VERIFY_PIN_FINISH	0x02
FEATURE_MODIFY_PIN_START	0x03
FEATURE_MODIFY_PIN_FINISH	0x04
FEATURE_GET_KEY_PRESSED	0x05
FEATURE_VERIFY_PIN_DIRECT	0x06
FEATURE_MODIFY_PIN_DIRECT	0x07
FEATURE_MCT_READER_DIRECT	0x08
FEATURE_MCT_UNIVERSAL	0x09
FEATURE_IFD_PIN_PROPERTIES	0x0A
FEATURE_ABORT	0x0B
FEATURE_SET_SPE_MESSAGE	0x0C
FEATURE_VERIFY_PIN_DIRECT_APP_ID	0x0D
FEATURE_MODIFY_PIN_DIRECT_APP_ID	0x0E
FEATURE_WRITE_DISPLAY	0x0F
FEATURE_GET_KEY	0x10
FEATURE_IFD_DISPLAY_PROPERTIES	0x11
FEATURE_GET_TLV_PROPERTIES	0x12
FEATURE_CCID_ESC_COMMAND	0x13
FEATURE_EXECUTE_PACE	0x20

**Table 1**

### 2.3.1 GET\_FEATURE\_REQUEST by Pseudo-APDU

The GET\_FEATURE\_REQUEST returns all features in the IFD (see also [7]).

This PPDU feature shall execute (see ch 2.2.1) as follows:

- The *FeatureNumber* is 0x00.
- The *FeatureCommandData* is empty ( $L_c = 0$ ).
- the *FeatureResponseData* is a byte array: each byte in this array represents a feature number present in this reader. Table 1 defines the FeatureNumbers.

## 3 Features

### 3.1 FEATURE\_VERIFY\_PIN\_START

The FEATURE\_VERIFY\_PIN\_START starts an indirect PIN procedure in the IFD (see also [7]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x01.
- The *FeatureCommandData* is according the PIN\_VERIFY structure, [7] chapter 2.5.2
- the *FeatureResponseData* is empty.

See also the other indirect PIN features FEATURE\_GET\_KEY\_PRESSED, FEATURE\_VERIFY\_PIN\_FINISH and FEATURE\_ABORT

### 3.2 FEATURE\_VERIFY\_PIN\_FINISH

The FEATURE\_VERIFY\_PIN\_FINISH ends an indirect PIN procedure in the IFD (see also [7]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x02.
- The *FeatureCommandData* is empty.
- the *FeatureResponseData* is a 2-byte status according [7] chapter 2.6.3.

See also the other indirect PIN features FEATURE\_VERIFY\_PIN\_START, FEATURE\_ABORT, FEATURE\_GET\_KEY\_PRESSED

### 3.3 FEATURE\_MODIFY\_PIN\_START

The FEATURE\_MODIFY\_PIN\_FINISH starts an indirect PIN procedure in the IFD (see also [7]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x03.
- The *FeatureCommandData* is according the PIN\_MODIFY structure, [7] chapter 2.5.3
- the *FeatureResponseData* is empty.

See also the other indirect PIN features FEATURE\_MODIFY\_PIN\_FINISH, FEATURE\_ABORT, FEATURE\_GET\_KEY\_PRESSED

### 3.4 FEATURE\_MODIFY\_PIN\_FINISH

The FEATURE\_MODIFY\_PIN\_FINISH ends an indirect PIN procedure in the IFD (see also [7]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x04.
- The *FeatureCommandData* is empty.
- the *FeatureResponseData* is a 2-byte status according [7] chapter 2.6.3.

See also the other indirect PIN features FEATURE\_MODIFY\_PIN\_START, FEATURE\_ABORT, FEATURE\_GET\_KEY\_PRESSED

### 3.5 FEATURE\_GET\_KEY\_PRESSED

The FEATURE\_GET\_KEY\_PRESSED can be used at an indirect PIN procedure in the IFD (see also [7]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x05.
- The *FeatureCommandData* is empty.
- the *FeatureResponseData* is a single byte according [7] chapter 2.6.2

See also the other indirect PIN features FEATURE\_VERIFY\_PIN\_START, FEATURE\_VERIFY\_PIN\_FINISH, FEATURE\_MODIFY\_PIN\_START, FEATURE\_MODIFY\_PIN\_FINISH, FEATURE\_ABORT

### 3.6 FEATURE\_VERIFY\_PIN\_DIRECT

The FEATURE\_VERIFY\_PIN\_DIRECT performs a complete (direct) PIN procedure in the IFD (see also [7]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x06.
- The *FeatureCommandData* is according the PIN\_VERIFY structure, [7] chapter 2.5.2
- the *FeatureResponseData* is a 2-byte status according [7] chapter 2.6.4.

### 3.7 FEATURE\_MODIFY\_PIN\_DIRECT

The FEATURE\_MODIFY\_PIN\_DIRECT performs a complete (direct) PIN procedure in the IFD (see also [7]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x07.
- The *FeatureCommandData* is according the PIN\_MODIFY structure, [7] chapter 2.5.3
- the *FeatureResponseData* is a 2-byte status according [7] chapter 2.6.4.

### 3.8 FEATURE\_MCT\_READER\_DIRECT

The FEATURE\_MCT\_READER\_DIRECT can be used to transmit a command to the IFD (see also [7]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x08.
- The *FeatureCommandData* is vendor specific, see ref [7] chapter 2.6.6.
- the *FeatureResponseData* is a buffer containing vendor specific data (data size can be null), see ref [7] chapter 2.6.6.

### 3.9 FEATURE\_MCT\_UNIVERSAL

The FEATURE\_MCT\_UNIVERSAL can be used to transmit a command to the IFD or the ICC (see also [7]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x09.
- The *FeatureCommandData* is according the MCT\_UNIVERSAL structure, see [7] chapter 2.5.4
- the *FeatureResponseData* will contain data according the MCT\_UNIVERSAL structure with SAD and DAD fields containing values concerning to table 7 of [2].

### 3.10 FEATURE\_IFD\_PIN\_PROPERTIES

The FEATURE\_IFD\_PIN\_PROPERTIES can be used to retrieve the properties of the IFD regarding PIN handling (see also [7]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x0A.
- The *FeatureCommandData* is empty.
- the *FeatureResponseData* is according the PIN\_PROPERTIES structure (see [7] chapter 2.5.5),

### 3.11 FEATURE\_ABORT

The FEATURE\_ABORT aborts an indirect PIN procedure in the IFD (see also [7]). This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x0B.
- The *FeatureCommandData* is empty.
- the *FeatureResponseData* is a 2-byte status according [7] chapter 2.6.5.

See also the other indirect PIN features FEATURE\_GET\_KEY\_PRESSED, FEATURE\_VERIFY\_PIN\_START, FEATURE\_VERIFY\_PIN\_FINISH, FEATURE\_MODIFY\_PIN\_START, FEATURE\_MODIFY\_PIN\_FINISH

### 3.12 FEATURE\_SET\_SPE\_MESSAGE

The FEATURE\_SET\_SPE\_MESSAGE can be used to define a message which should be displayed during an SPE operation in the IFD (see also [7]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x0C.
- The *FeatureCommandData* is according the SET\_SPE\_MESSAGE structure, [7] chapter 2.5.7
- the *FeatureResponseData* contains a 2-byte status according [7] chapter 2.6.10

See also the SPE related features FEATURE\_VERIFY\_PIN\_DIRECT, FEATURE\_VERIFY\_PIN\_DIRECT\_APP\_ID, FEATURE\_MODIFY\_PIN\_DIRECT, FEATURE\_MODIFY\_PIN\_DIRECT\_APP\_ID

### 3.13 FEATURE\_VERIFY\_PIN\_DIRECT\_APP\_ID

The FEATURE\_VERIFY\_PIN\_DIRECT\_APP\_ID performs a complete (direct) PIN procedure in the IFD (see also [7]), based on specific SPE messages.

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x0D.
- The *FeatureCommandData* is according the PIN\_VERIFYAPP\_ID structure, [7] chapter 2.5.8
- the *FeatureResponseData* is a 2-byte status according [7] chapter 2.6.3.

See also the SPE related features FEATURE\_SET\_SPE\_MESSAGE, FEATURE\_MODIFY\_PIN\_DIRECT\_APP\_ID

### 3.14 FEATURE\_MODIFY\_PIN\_DIRECT\_APP\_ID

The FEATURE\_MODIFY\_PIN\_DIRECT\_APP\_ID performs a complete (direct) PIN procedure in the IFD (see also [7]), based on specific SPE messages.

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x0E.
- The *FeatureCommandData* is according the PIN\_MODIFY\_APP\_ID structure, [7] chapter 2.5.9
- the *FeatureResponseData* is a 2-byte status according [7] chapter 2.6.3

See also the SPE related features FEATURE\_SET\_SPE\_MESSAGE, FEATURE\_VERIFY\_PIN\_DIRECT\_APP\_ID

### 3.15 FEATURE\_WRITE\_DISPLAY

The FEATURE\_WRITE\_DISPLAY writes any UTF-8 based message on the display of the IFD (see also [7]), if SPE is not active.

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x0F.
- The *FeatureCommandData* is according the WRITE\_DISPLAY structure, [7] chapter 2.5.2
- the *FeatureResponseData* is empty.

### 3.16 FEATURE\_GET\_KEY

The FEATURE\_GET\_KEY retrieves the value of a pressed key on the keypad of the IFD (see also [7]), if SPE is not active.

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x10.
- The *FeatureCommandData* is according the GET\_KEY structure [7] chapter 2.5.11
- the *FeatureResponseData* is a single byte according [7] chapter 2.6.13

### 3.17 FEATURE\_IFD\_DISPLAY\_PROPERTIES

The FEATURE\_IFD\_DISPLAY\_PROPERTIES returns a structure with the properties of the display of the IFD (see also [7]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x11.
- The *FeatureCommandData* is empty.
- the *FeatureResponseData* is according the DISPLAY\_PROPERTIES structure, [7] chapter 2.5.6

### 3.18 FEATURE\_GET\_TLV\_PROPERTIES

The FEATURE\_GET\_TLV\_PROPERTIES returns a TLV list of the properties of the IFD (see also [7]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x12.
- The *FeatureCommandData* is empty.
- the *FeatureResponseData* is a TLV structure according [7] chapter 2.6.14

### 3.19 FEATURE\_CCID\_ESC\_COMMAND

The FEATURE\_CCID\_ESC\_COMMAND is used to exchange vendor proprietary information with the reader (see also [7]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x13.
- The *FeatureCommandData* is vendor specific.
- the *FeatureResponseData* is vendor specific

### 3.20 FEATURE\_EXECUTE\_PACE

The FEATURE\_EXECUTE\_PACE is used to command the PACE functionality within the reader (see also [8]).

This PPDU feature shall execute (see ch 2.2) as follows:

- The *FeatureNumber* is 0x20.
- The *FeatureCommandData* is according the InBuffer structure of ch 2.5.12 in ref [8].
- the *FeatureResponseData* is according the OutBuffer structure of ch 2.5.12 in ref [8].

## **Abbreviations**

IFD	Interface Device
MCT	Multifunctional Card Terminal
PIN	Personal Identification Number
SPE	Secure PIN Entry
TLV	Tag Length Value
APDU	Application Protocol Data Unit
PPDU	Peripheral Processor Data Unit (Pseudo-APDU)



## **4 References**

- [1] International technology – Identification cards – Integrated circuit(s) cards with contacts – Part 4: Interindustry commands for interchange; International Standard ISO/IEC 7816-4:1995(E)
- [2] Multifunctional Card Terminals, Part 3 - Application Independent Card Terminal Application Programming Interface for ICC Applications (CT-API 1.1); Deutsche Telekom AG / PZ Telesec, SIT Fraunhofer Institut für Sichere Telekooperation, TÜV Informationstechnik GmbH, TeleTrust Deutschland e.V.; 2002
- [3] Multifunctional Card Terminals, Part 4 - CT-BCS - Application Independent Card Terminal Basic Command Set; Deutsche Telekom AG / PZ Telesec, SIT Fraunhofer Institut für Sichere Telekooperation, TÜV Informationstechnik GmbH, TeleTrust Deutschland e.V.; 2002
- [4] USB Serial Bus Device Class Spec of USB Chip/Smart Card Interface Devices, Revision 1.1
- [5] ISO/IEC 7816: Identification cards — Integrated circuit cards — Part 3: Cards with contacts — Electrical interface and transmission protocols, ISO/IEC 7816-3 Third edition 2006-11-01
- [6] Interoperability Specification for ICCs and Personal Computer Systems, Part 5. ICC Resource Manager Definition, Revision 2.01.01 September 2005
- [7] Interoperability Specification for ICCs and Personal Computer Systems Part 10 IFDs with Secure PIN Entry Capabilities, v2.02.09 January 2012
- [8] Interoperability Specification for ICCs and Personal Computer Systems Part 10 IFDs with Secure PIN Entry Capabilities, Revision 2.02.08, April 2010, AMENDMENT 1, 2011-06-03