INTERNATIONAL STANDARD

ISO/IEC 19784-2

First edition
2007-02-01

Information technology — Biometric application programming interface —

Part 2: Biometric archive function provider interface

Technologies de l'information — Interface de programmation d'applications biométriques —

Partie 2: Interface du fournisseur de fonction d'archives biométriques

Reference number
ISO/IEC 19784-2:2007(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 19784-2 was prepared by Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 37, *Biometrics*.

ISO/IEC 19784 consists of the following parts, under the general title *Information technology — Biometric application programming interface*:

— *Part 1: BioAPI specification*

— *Part 2: Biometric archive function provider interface*

Introduction

This part of ISO/IEC 19784 specifies the interface to an archive BFP. An archive BFP is responsible for the storage and management of BIRs. The archive BFP interface covers functions to attach, detach and operate related functional units by the biometric service provider, to store BIRs, to retrieve information about stored BIRs and to retrieve formerly stored BIRs for further processing by a BSP.

ISO/IEC 19784-1 provides a high-level generic biometric authentication model. This part of ISO/IEC 19784 specifies a BSP archive interface. An archive is responsible for the management of biometric data storage. Because of the application-specific requirements of biometric data storage, an archive interface is standardized.

The BioAPI Unit where the BIRs are stored is completely under control of the archive BFP. The general model for the handling of BIRs is as if they were data records in a database.

Information technology— Biometric application programming interface —

Part 2:
Biometric archive function provider interface

1 Scope

This part of ISO/IEC 19784 specifies the interface to an archive biometric function provider.

NOTE The interface assumes that the archive will be handled as a database, regardless of its physical realization. (Smartcards, tokens, memory sticks, files on hard drives and any other kind of memory can be handled via an abstraction layer presenting a database interface.)

This part of ISO/IEC 19784 enables any third party to create biometric archive function providers, which may be plugged into any biometric service provider supporting this interface.

It is not in the scope of this part of ISO/IEC 19784 to define security and privacy requirements for storage and management of BIRs.

2 Conformance

A conformant BAFP is required to support all functions and parameters specified in this part of ISO/IEC 19784. No subsets of conformant BAFP functions are defined.

NOTE BSPs may require any of those functions and parameters.

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

NOTE Function names and data element names are not included here, but are defined within the body of this part of ISO/IEC 19784.

3.1
biometric archive function provider
BAFP
BioAPI component, attached to a BSP via its interface, for storage, management and interchange of BIRs

3.2
biometric archive function provider interface
BAFPI
BAFP-to-BSP interface which supports the functions to manage the BAFP itself, its databases or BIRs

4 Interface architecture

ISO/IEC 19784-1:2006 specifies the interface at the top of the BioAPI Framework, which a biometric application uses to interact with BSPs, and through that to biometric units either directly or through BFPs (see Figure 1 and Figure 2 of ISO/IEC 19784-1:2006).

The BAFPI contains the following types of functions.

- General BFP management functions, which are called by the BSP to manage the BFP and its associated BioAPI Units. These functions are not directly related to SPI functions:
  - BioAFPI_BAFPLoad;
  - BioAFPI_BAFPUnload;
  - BioAFPI_UnitAttach;
  - BioAFPI_UnitDetach;
  - BioAFPI_QueryUnits;
  - BioAFPI_Free (the relation of this function to the respective SPI call depends on the behaviour of the BSP – when the BSP copies the data the BFP has allocated memory for then it can immediately send BioAFPI_Free to the BFP, otherwise the BSP will call this function when itself is requested to free the memory for these data).
- Unit management functions:
  - BioAFPI_ControlUnit (this function is directly related to the SPI function);
  - BioAFPI_Cancel (this function can be called in relation to the respective calls of the SPI or the application, but can be called for other reasons from the BSP also when there is no respective request from the BioAPI framework or application;
  - BioAFPI_SetPowerMode (this function is directly related to the SPI function).
- Database management functions. These functions are directly related to the respective functions of the SPI:
  - BioAFPI_DbOpen;
  - BioAFPI_DbClose;
  - BioAFPI_DbCreate;
  - BioAFPI_DbDelete.
- Record management functions. These functions are directly related to the respective functions of the SPI:
  - BioAFPI_DbSetMarker;
  - BioAFPI_DbFreeMarker.
- Data management functions. These functions are directly related to the respective functions of the SPI:
  - BioAFPI_DbStoreBIR;
  - BioAFPI_DbGetBIR;
  - BioAFPI_DbGetNextBIR;
  - BioAFPI_DbQueryBIR;
  - BioAFPI_DbDeleteBIR.

5 BAFPI Definition

5.1 BAFPI data structures

5.1.1 BioAFPI_EventHandler

This defines the event handler interface to receive asynchronous notification of events of type BioAPI_ EVENT from a BioAPI Unit. Example events include insertion or removal of a BioAPI Unit (e.g. insertion or withdrawal of a Smartcard storing biometric templates for off card or on card matching).

This event handler is passed to the BAFP during BioAFPI_BFPLoad. This is the single event handler that all BioAPI Units managed by this BAFP should use to notify the BSP of event types that occur in a loaded BAFP.

```
typedef BioAPI_RETURN (BioAPI *BioAFPI_EventHandler)
       (const BioAPI_UUID *BAFPUuid,
       BioAPI_UNIT_ID UnitID,
       BioAPI_UNIT_SCHEMA *UnitSchema,
       BioAPI_ EVENT EventType);
```

BAFPUuid (input) — The UUID of the BAFP raising the event.

UnitID (input) — The unit ID of the BioAPI Unit (biometric archive) associated with the event.

UnitSchema (input) — The schema of the BioAPI Unit raising the event.

EventType (input) — The BioAPI_ EVENT that has occurred.

5.1.2 BioAFPI_BAFPPropertyID

Defines the UUID of the format of the BioAFPI_BAFPPropertySchema. When a BAFP claims to be conformant to this part of ISO/IEC 19784, it has to use a PropertySchema as defined within this part of ISO/IEC 19784.

```
#define (BioAFPI_BAFPPropertyID, 0x561403e5, 0x7d2d, 0x489a, 0x89, 0x41, 0xe7, 0xa9, 0x35,
                                 0x99, 0xcf, 0x9c);
```

5.1.3 BioAFPI_BAFPPropertySchema

The BioAFPI_BAFPPropertySchema contains information about the function provider's capabilities, which are presented to the BioAPI component registry and in the related function calls. The function provider capabilities can differ from the capabilities of a biometric archive unit. The capabilities usuable by a BSP or BioAPI application are the intersection of the BAFP and biometric archive unit capabilities.

```
typedef struct _bioafpi_bafp_property_schema {
        uint32_t                        MaxRecordLength;
        uint32_t                        MaxRecordNumber;
        uint32_t                        MaxDatabases;
        uint32_t                        MaxResponseTime;
        BioAPI_POWER_MODE               SupportedPowerModes;
    }BioAFPI_BAFP_PROPERTY_SCHEMA, *BioAFPI_BAFP_PROPERTY_SCHEMA_PTR;
```

MaxRecordLength – The value defines the maximum length in bytes of a record. A record presents a BIR. A Biometric Unit shall use a value smaller or equal to this value in its UnitSchema. A '0' means that no information about the maximum record length is given.

MaxRecordNumber – The value defines the maximum number of records. A Biometric Unit shall use a value smaller or equal to this value in its UnitSchema. A '0' means that no information about the maximum number of records is given.

MaxDatabases – The value defines the maximum number of databases, which can be managed by the AFP at a time. A '0' means that no information about the maximum number of databases is given.

MaxResponseTime – The value defines the maximum time in milliseconds a caller of any BAFP function has to wait until the function returns. A Biometric Unit shall use a value smaller or equal to this value in its UnitSchema. A '0' means that no information about the maximum time until the BioAPI Unit returns is given.

SupportedPowerModes – The bit mask defining, which power modes can be handled by the BFP. Handling of a power mode means to pass the related function call to the BioAPI Unit, when this supports the mode and to return the error code related to the result of the reaction of the BioAPI Unit.

5.1.4 BioAFPI_UnitPropertyID

Defines the UUID of the format of the BioAFPI_UnitPropertySchema. When a BAFP claims to be conformant to this part of ISO/IEC 19784, it has to use a PropertySchema as defined within this part of ISO/IEC 19784.

```
#define (BioAFPI_UnitPropertyID, 0x94cae58f, 0x6b26, 0x41fc, 0x9d, 0xdb, 0xd9, 0x8e, 0x9a,
        0xd5, 0x6f, 0xa0);
```

5.1.5 BioAFPI_UnitPropertySchema

The BioAFPI_UnitPropertySchema contains the information to be presented in the related function calls. Presenting a particular biometric archive unit this schema contains information to identify the unit itself (DbUuid and DbName) and capability information as well.

```
typedef struct _bioafpi_unit_property_schema {
       BioAPI_UUID                  DbUuid;
       BioAPI_STRING                DbName;
       uint32_t                     MaxRecordLength;
       uint32_t                     MaxRecordNumber;
       uint32_t                     MaxDatabases;
       uint32_t                     MaxResponseTime;
       BioAPI_POWER_MODE            SupportedPowerModes;
   }BioAFPI_UNIT_PROPERTY_SCHEMA, *BioAFPI_UNIT_PROPERTY_SCHEMA_PTR;
```

DbUuid – The UUID of the archive unit related database.

DbName – The descriptional name of the database.

MaxRecordLength – The value defines the maximum length in bytes of a record. A record presents a BIR. A '0' means that no information about the maximum record length is given.

MaxRecordNumber – The value defines the maximum number of records. A '0' means that no information about the maximum number of records is given.

MaxDatabases – The value defines the maximum number of databases, which can be managed by the AFP at a time. A '0' means that no information about the maximum number of databases is given.

MaxResponseTime – The value defines the maximum time in milliseconds a caller of any function to the BioAPI Unit has to wait until the function returns. A '0' means that no no information about the maximum time until the BioAPI Unit returns is given.

SupportedPowerModes – The bit mask defining, which power modes can be handled by the BFP. Handling of a power mode means to pass the related function call to the BioAPI Unit, when this supports the mode and to return the error code related to the result of the reaction of the BioAPI Unit.

5.2 BAFP Functions

5.2.1 BioAFPI_BAFPLoad

```
BioAPI_RETURN BioAPI BioAFPI_BAFPLoad
    (const BioAPI_UUID *BAFPUuid,
    BioAFPI_EventHandler BioAFPINotifyCallback);
```

Description
This function loads an Archive BFP.

The *BAFPUuid* identifies the invoked.

The *BioAFPINotifyCallback* defines a callback used to notify the BSP of events of type BioAPI_EVENT. The BAFP shall retain this information for later use.

Parameters

BAFPUuid (input) – The UUID of the invoked BAFP. Used to locate the component's directory entry.

BioAFPINotifyCallback (input) – A function pointer for the event handler that manages events of type BioAPI_ EVENT.

Return Value

A BioAPI_RETURN value indicating success or specifying a particular error condition. The value BioAPI_OK indicates success. All other values represent an error condition.

Errors

```
BioAPIERR_MODULE_LOAD_FAIL
BioAPIERR_INVALID_UUID
```

5.2.2 BioAFPI_BAFPUnload

```
BioAPI_RETURN BioAPI BioAFPI_BAFPUnload
    (const BioAPI_UUID *BAFPUuid,
    BioAFPI_EventHandler BioAFPINotifyCallback);
```

Description

This function disables events and de-registers the BSP event-notification function. The BAFP shall perform cleanup operations, reversing the initialization performed in *BioAFPI_BAFPLoad*.

Parameters

BAFPUuid (input) – The UUID of the invoked BAFP.

BioAFPINotifyCallback (input) – A function pointer for the event handler that manages events of type BioAPI_ EVENT.

Return Value

A BioAPI_RETURN value indicating success or specifying a particular error condition. The value BioAPI_OK indicates success. All other values represent an error condition.

Errors

`BioAPIERR_INVALID_UUID`

5.2.3 BioAFPI_UnitAttach

```
BioAPI_RETURN BioAPI BioAFPI_UnitAttach
    (BioAPI_UNIT_ID UnitID);
```

Description
This function creates an Attach Session.

There is no requirement that the unit ID value provided by a BSP as input to this function be the same unit ID value that the framework provided to the BSP in the original call to BioSPI_BSPAttach (if any), provided that the two unit ID values identify the same BioAPI unit (see 5.2.5).

Parameters

UnitID (input) – The ID of that BioAPI Unit defined by a prior call to BioAFPI_QueryUnits. The UnitID has been filled into the UnitId field of each element of the BioAPI_UNIT_SCHEMA_ARRAY by the BAFP.

Return Value

A BioAPI_RETURN value indicating success or specifying a particular error condition. The value BioAPI_OK indicates success. All other values represent an error condition.

Errors

```
BioAPIERR_INVALID_UNIT_ID
BioAPIERR_UNIT_IN_USE
```

5.2.4 BioAFPI_UnitDetach

```
BioAPI_RETURN BioAPI BioAFPI_UnitDetach
    (BioAPI_UNIT_ID UnitID);
```

Description
This function terminates an Attach Session.

There is no requirement that the unit ID value provided by a BSP as input to this function be the same unit ID value that the framework provided to the BSP in the original call to BioSPI_BSPAttach (if any), provided that the two unit ID values identify the same BioAPI unit (see 5.2.5).

Parameters

UnitID (input) – The ID of that BioAPI Unit defined by a prior call to BioAFPI_QueryUnits. The UnitID has been filled into the UnitId field of each element of the BioAPI_UNIT_SCHEMA_ARRAY by the BAFP.

Return Value

A BioAPI_RETURN value indicating success or specifying a particular error condition. The value BioAPI_OK indicates success. All other values represent an error condition.

Errors

`BioAPIERR_INVALID_UNIT_ID`

5.2.5 BioAFPI_QueryUnits

```
BioAPI_RETURN BioAPI BioAFPI_QueryUnits
   (const BioAPI_UUID *BafpUuid,
    BioAPI_UNIT_SCHEMA **UnitSchemaArray,
    uint32_t *NumberOfElements);
```

Description

This function returns an array of BioAPI Unit schemas (see 7.55 of ISO/IEC 19784-1:2006), which are managed by the given BAFP and are currently in the inserted state.

NOTE When the BSP calls the function ***BioAFPI_QueryUnits*** of a BFP, the BFP allocates memory for the data to be returned to the BSP. In some implementation architectures, the BSP will simply pass to the framework the data and the addresses exactly as returned by the BFP because the framework will interpret the addresses in the same way as the BFP and will be able to access the data that the BFP has placed at those addresses. In other implementation architectures, the BSP will need to move all the data returned by the BFP to newly allocated memory blocks accessible to the framework, and will call ***BioAFPI_Free*** after copying each memory block but before returning from the ***BioSPI_QueryUnits*** call. In the former case, when the framework calls ***BioSPI_Free***, the BSP will make a corresponding call to ***BioAFPI_Free***. In the latter case, the calls to ***BioSPI_Free*** will be handled internally by the BSP. However, such differences in the behavior of the BSP are not visible to the Framework.

The memory block containing the array shall be freed by the BSP via a call to ***BioAFPI_Free*** when it is no longer needed by the BSP. The memory block pointed to by the *UnitProperty* member within each element of the array shall also be freed by the BSP via a call to ***BioAFPI_Free*** when it is no longer needed by the BSP.

This function shall only be called after ***BioAFPI_Load*** has been called for the specified BFP, and shall not be called after ***BioAFPI_Unload*** has been called for the BFP.

There is no requirement that a unit ID, returned by this function for a given BioAPI unit, be made available with the same unit ID value by the BSP to the framework. A BSP is free to translate any unit ID value (provided by a BFP) into a different unit ID value before providing it to the framework. The purpose of such a translation would be to avoid the existence of duplicate unit IDs within the scope of the BSP, which might happen when a BSP is using two or more BFPs of the same category, or when a BSP is using an archive BFP while also directly managing archive units.

Parameters

BafpUuid (input) – The unique identifier for the BAFP for which the information is to be returned.

UnitSchemaArray (output) – A pointer to an address of the array of elements of type BioAPI_UNIT_SCHEMA (allocated by the BSP - but see note above) containing the unit schema information.

NumberOfElements (output) – A pointer to the number of elements in the array, which are managed by the given BAFP and are currently in the inserted state.

Return Value

A BioAPI_RETURN value indicating success or specifying a particular error condition. The value BioAPI_OK indicates success. All other values represent an error condition.

Errors

`BioAPIERR_INVALID_UUID`
`BioAPIERR_MEMORY_ERROR`

5.2.6 BioAFPI_Free

```
BioAPI_RETURN BioAPI BioAFPI_Free
    (void *Ptr);
```

Description

This function causes the space pointed to by *Ptr* to be deallocated. If *Ptr* is NULL, no action occurs. Otherwise, if *Ptr* does not match a pointer earlier returned by the BioAFPI functions, or if the space already has been deallocated by a call to ***BioAFPI_Free***, the behavior is undefined.

Parameters

Ptr (*input*) – A pointer to the memory to free.

Return Value

A BioAPI_RETURN value indicating success or specifying a particular error condition. The value BioAPI_OK indicates success. All other values represent an error condition.

Errors

5.2.7 BioAFPI_ControlUnit

```
BioAPI_RETURN BioAPI BioAFPI_ControlUnit
    (BioAPI_UNIT_ID   UnitID,
     uint32_t ControlCode,
     const BioAPI_DATA *InputData,
    BioAPI_DATA *OutputData);
```

Description

This function sends control data to the BioAPI Unit and receives status or operation data from there. The content of the *ControlCode*, the send (input) data, and the receive (output) data will be specified in the related interface specification for this BioAPI Unit.

The function allocates a memory block large enough to contain the output data that are to be returned to the application, fills the memory block with the data, and sets the fields *Length* and *Data* of the *OutputData* structure to the size and address of the memory block (respectively).

The memory block returned by the BioAFPI function call shall be freed by the application using ***BioAFPI_Free***.

This function is a transparent channel to the BioAPI Unit and can be used for any purposes, which cannot be realized by standardized functions (e.g. archive maintenance, index recreation etc.).

There is no requirement that the unit ID value provided by a BSP as input to this function be the same unit ID value that the framework provided to the BSP in the original call to BioSPI_BSPAttach (if any), provided that the two unit ID values identify the same BioAPI unit (see 5.2.5).

Parameters

UnitID (input) – The ID of that BioAPI Unit previously defined by the BAFP.

ControlCode (input) – The function code in the BioAPI unit to be called.

InputData (input) – Address and length of a buffer containing the data to be send to the BioAPI unit related to the given *ControlCode*.

OutputData (output) – Pointer to a BioAPI_DATA structure. On output, this shall contain the address and length of a data buffer containing the data received from the BioAPI unit after processing the function indicated by the *ControlCode*. If no memory block has been allocated by the function, the address shall be set to NULL and the length shall be set to zero.

Return Value

A BioAPI_RETURN value indicating success or specifying a particular error condition. The value BioAPI_OK indicates success. All other values represent an error condition.

Errors

```
BioAPIERR_MEMORY_ERROR
BioAPIERR_INVALID_POINTER
BioAPIERR_FUNCTION_FAILED
```

5.2.8 BioAFPI_Cancel

```
BioAPI_RETURN BioAPI BioAFPI_Cancel
    (BioAPI_UNIT_ID   UnitID);
```

Description
This function shall cancel the presently blocked call to the BioAPI Unit. The function shall not return until the blocking call has been cancelled.

There is no requirement that the unit ID value provided by a BSP as input to this function be the same unit ID value that the framework provided to the BSP in the original call to BioSPI_BSPAttach (if any), provided that the two unit ID values identify the same BioAPI unit (see 5.2.5).

Parameters
UnitID (input) – The ID of that BioAPI Unit where the blocked call shall be cancelled.

Return Value
A BioAPI_RETURN value indicating success or specifying a particular error condition. The value BioAPI_OK indicates success. All other values represent an error condition.

Errors

5.2.9 BioAFPI_SetPowerMode

```
BioAPI_RETURN BioAPI BioAFPI_SetPowerMode
    (BioAPI_UNIT_ID  UnitId,
     BioAPI_POWER_MODE  PowerMode);
```

Description

This function sets the currently attached BioAPI Unit to the requested power mode if the BioAPI Unit supports it.

There is no requirement that the unit ID value provided by a BSP as input to this function be the same unit ID value that the framework provided to the BSP in the original call to BioSPI_BSPAttach (if any), provided that the two unit ID values identify the same BioAPI unit (see 5.2.5).

Parameters

UnitId (input) – The ID of the BioAPI Unit for which the power mode is to be set. The BioAPI_DONT_CARE option is not valid for this function.

PowerMode (input) – A 32-bit value indicating the power mode to set the BioAPI Unit to.

Return Value

A BioAPI_RETURN value indicating success or specifying a particular error condition. The value BioAPI_OK indicates success. All other values represent an error condition.

Errors

`BioAPIERR_FUNCTION_NOT_SUPPORTED`

5.2.10 BioAFPI_DbOpen

```
BioAPI_RETURN BioAPI BioAFPI_DbOpen
    (BioAPI_UNIT_ID   UnitID,
    BioAPI_DB_ACCESS_TYPE   AccessRequest,
    BioAPI_DB_MARKER  *Marker);
```

Description

This function opens a BIR database maintained by the currently attached archive of the identified BioAPI Unit invocation, using the access mode specified by the AccessRequest. A database *Marker* is set to point to the first record in the BIR database.

NOTE Every call to this function should be followed at some point by a call to *BioAFPI_DbFreeMarker*, to allow the marker resource to be released. BSPs should be aware that every call to *BioAFPI_DbOpen* results in the allocation of some resources that cannot be automatically freed (otherwise it would be impossible to iterate through the database using the returned marker handle). If the BSP doesn't want to iterate through the database, it should call *BioAFPI_DbFreeMarker* immediately, otherwise it should do so after terminating the iteration.

There is no requirement that the unit ID value provided by a BSP as input to this function be the same unit ID value that the framework provided to the BSP in the original call to BioSPI_BSPAttach (if any), provided that the two unit ID values identify the same BioAPI unit (see 5.2.5).

Parameters

UnitID (input) – The ID of that BioAPI Unit previously defined by the BAFP.

AccessRequest (input) – An indicator of the requested access mode for the database, **read** or **write**. On a single system, only one application at a time is allowed to open a target database in BioAPI_DB_ACCESS_WRITE mode. This is to be managed by the BSP.

Marker (output) – A pointer that can be used to iterate through the database.

Return Value

A BioAPI_RETURN value indicating success or specifying a particular error condition. The value BioAPI_OK indicates success. All other values represent an error condition.

Errors

```
BioAPIERR_INVALID_UNIT_ID
BioAPIERR_UNABLE_TO_OPEN_DATABASE
BioAPIERR_DATABASE_IS_LOCKED
BioAPIERR_DATABASE_DOES_NOT_EXIST
BioAPIERR_INVALID_ACCESS_REQUEST
BioAPIERR_DATABASE_IS_CORRUPT
```

5.2.11 BioAFPI_DbClose

```
BioAPI_RETURN BioAPI BioAFPI_DbClose
    (BioAPI_UNIT_ID   UnitID);
```

Description
This function closes an open BIR database supported by the identified BioAPI Unit invocation. All markers currently set for records in the database are freed.

NOTE A database opened in DB_ACCESS_WRITE mode can be damaged if it is left unclosed.

There is no requirement that the unit ID value provided by a BSP as input to this function be the same unit ID value that the framework provided to the BSP in the original call to BioSPI_BSPAttach (if any), provided that the two unit ID values identify the same BioAPI unit (see 5.2.5).

Parameters
UnitID (input) – The ID of that BioAPI Unit previously defined by the BAFP.

Return Value
A BioAPI_RETURN value indicating success or specifying a particular error condition. The value BioAPI_OK indicates success. All other values represent an error condition.

Errors
BioAPIERR_INVALID_UNIT_ID
BioAPIERR_UNABLE_TO_CLOSE_DATABASE
BioAPIERR_DATABASE_IS_CORRUPT

5.2.12 BioAFPI_DbCreate

```
BioAPI_RETURN BioAPI BioAFPI_DbCreate
    (const BioAPI_UUID  *DbUuid,
    uint32_t  NbrOfRecords);
```

Description

This function creates a new BIR database supported by the currently attached BAFP. The identification of the new database is specified by the input parameter *DbUuid,* which shall be created by the biometric application, and shall be distinct from any current database. The BAFP will not open this database hence there is no BioAPI Unit created managing the database. After successful creation of the database the BAFP will first write the information into the component registry including maximum record size, maximum record number and then raise an insert event to the BSP and thus presents a new archive unit in inserted state to the system.

Parameters

DbUuid (input) – A pointer to a UUID identifying the BIR database to be created.

NbrOfRecords (input) – The maximum number of records to be stored at in the BIR database.

Return Value

A BioAPI_RETURN value indicating success or specifying a particular error condition. The value BioAPI_OK indicates success. All other values represent an error condition.

Errors

```
BioAPIERR_UNABLE_TO_CREATE_DATABASE
BioAPIERR_DATABASE_ALREADY_EXISTS
BioAPIERR_INVALID_UUID
BioAPIERR_INVALID_UNIT_ID
```

5.2.13 BioAFPI_DbDelete

```
BioAPI_RETURN BioAPI BioAFPI_DbDelete
    (BioAPI_UNIT_ID   UnitID);
```

Description

This function deletes all records from the specified BIR database and removes all state information associated with that database. The BAFP will also delete all information out of the component registry. After successful deletion of the database the BAFP will raise a remove event.

There is no requirement that the unit ID value provided by a BSP as input to this function be the same unit ID value that the framework provided to the BSP in the original call to BioSPI_BSPAttach (if any), provided that the two unit ID values identify the same BioAPI unit (see 5.2.5).

Parameters

UnitID (input) – The ID of the archive unit, which manages the database to be deleted.

Return Value

A BioAPI_RETURN value indicating success or specifying a particular error condition. The value BioAPI_OK indicates success. All other values represent an error condition.

Errors

`BioAPIERR_DATABASE_IS_OPEN`
`BioAPIERR_INVALID_UNIT_ID`

5.2.14 BioAFPI_DbSetMarker

```
BioAPI_RETURN BioAPI BioAFPI_DbSetMarker
    (BioAPI_UNIT_ID   UnitID,
    const BioAPI_UUID  *KeyValue,
    BioAPI_DB_MARKER  *Marker);
```

Description

The *Marker* is set to point to the record indicated by the *KeyValue* in the BIR database identified by the *UnitID*. A NULL value will set the marker to the first record in the database.

NOTE When an error occurs, the position of the marker does not change.

There is no requirement that the unit ID value provided by a BSP as input to this function be the same unit ID value that the framework provided to the BSP in the original call to BioSPI_BSPAttach (if any), provided that the two unit ID values identify the same BioAPI unit (see 5.2.5).

Parameters

UnitID (input) – Indicates the archive unit managing the database.

KeyValue (input) – The key into the database of the BIR to which the *Marker* is to be set.

Marker (input/output) – A pointer that can be used to iterate through the database from the retrieved record.

Return Value

A BioAPI_RETURN value indicating success or specifying a particular error condition. The value BioAPI_OK indicates success. All other values represent an error condition.

Errors

```
BioAPIERR_RECORD_NOT_FOUND
BioAPIERR_INVALID_UNIT_ID
```