Federal Office
for Information Security

BSI Technical Guideline TR-03121-2

Biometrics for Public Sector Applications

Part 2: Software Architecture

Volume 1: BioAPI (Deprecated)

Version 5.0

Federal Office for Information Security
P.O. Box 20 03 63
53133 Bonn, Germany
E-mail: TRBiometrics@bsi.bund.de
Web: https://bsi.bund.de

1. Introduction 1
2. Software Architecture 2
2.1. General Architecture 2
2.1.1. Working Example 3
2.2. BSP Calling Profiles 4
2.2.1. Calling Requirements for all BSPs 4
2.2.2. Calling Requirements for Capture BSPs 4
2.2.3. Calling Requirements for Verification Engine BSPs 5
2.2.4. Calling Requirements for Verification BSPs 6
2.3. Further Specifications for SFPI and BSFPs 7
2.3.1. Additional SFPI Communication 7
2.3.2. Additional BSFP Parameters 7
2.3.3. Serialisation of Additional Parameters Structure 9
List of Abbreviations 10
Bibliography 11

List of Figures

2.1. Software Architecture for (Biometric) Document Issuing Authorities 2
2.2. Software Architecture for Biometric Quality Assurance 3
2.3. QA Module Provider Interface for Face Biometrics 3
2.4. QA Module Provider Interface for Fingerprint Biometrics 3

List of Tables

2.1. Calling Requirements for BioSPI_ControlUnit 4
2.2. Calling Requirements for BioSPI_Capture 4
2.3. Calling Requirements for BioSPI_CreateTemplate 5
2.4. Calling Requirements for BioSPI_Process 5
2.5. Calling Requirements for BioSPI_VerifyMatch 5
2.6. Calling Requirements for BioSPI_Enroll 6
2.7. Calling Requirements for BioSPI_Verify 7
2.8. Calling Requirements for BioSPI_Verify 7
2.9. List of Additional Parameters 8

1. Introduction

This document specifies the Software Architecture within the scope of this guideline. The general software architecture and the REQUIRED interfaces are introduced and described.

This document describes a C-style interface for integration of biometric components which is currently not developed further by the International Standardization community. Therefore, this document is *deprecated* for new applications, but maintained for existing applications according to previous versions of this guideline. For new implementations refer to the volume TR-03121-2.2 which describes the High Level Biometric Services (HLBS) interface specification.

2. Software Architecture

2.1. General Architecture

The software architecture pursues the uniform strategy to integrate biometric processes in different enrolment-, verification- and identification scenarios within the scope of German public sector applications. The Software Architecture is based on open standards, in particular BioAPI 2.0 [ISO_19784-1]. Applications are using the BioAPI 2.0 Framework to access the particular functionality for the specific Application Profile, which is implemented in a BioAPI 2.0 Biometric Service Provider (BSP). Figure 2.1 gives an overview of the different enclosed layers, where the type of the applications and Biometric Service Providers (BSPs) should be seen as an example.

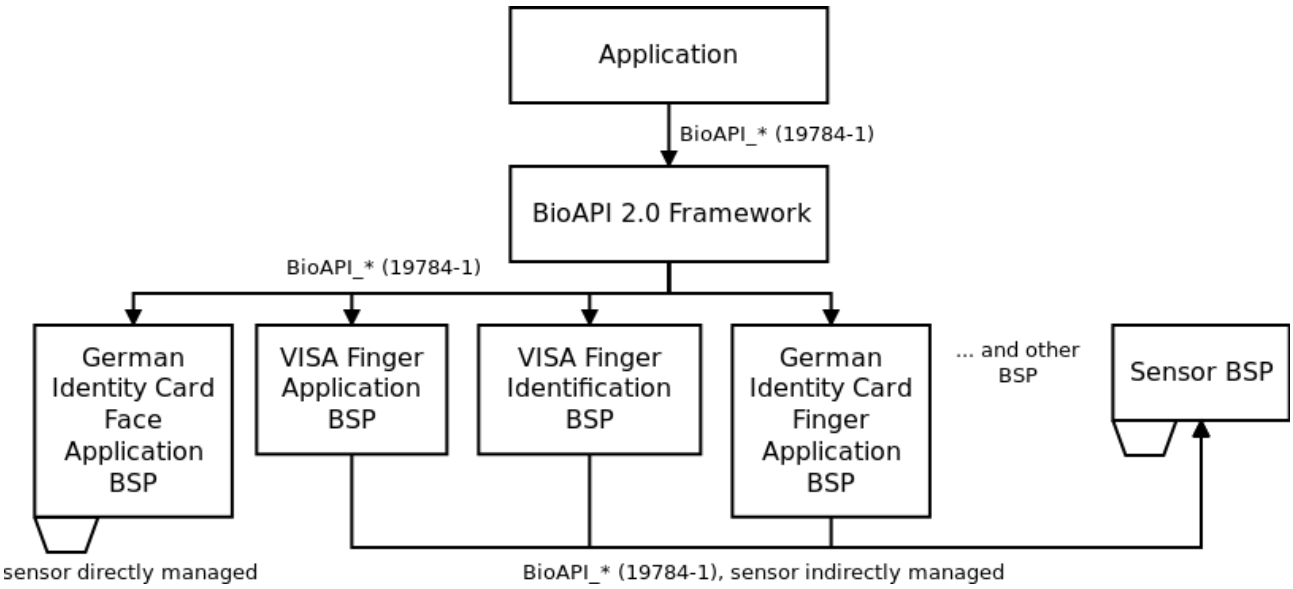

Figure 2.1. Software Architecture for (Biometric) Document Issuing Authorities

An Application Profile supports for every biometric feature one or more separate Biometric Service Providers (BSP) that can be accessed through a standardised BioAPI interface defined by [ISO_19784-1].

While the BSP typically implements a complex work flow, it uses a hardware component with a rather simple interface (e.g. acquisition of a single image). To address the sensor hardware, basically two approaches are possible:

- One possibility is that a BSP can manage a sensor directly, which means that all functionality for initialisation, loading, processing and termination of the device is included in the BSP in a vendor specific way, like integrating the sensor vendors Software Development Kit (SDK).
- Another possibility is the realisation of two disjoint components – a BSP on the application side and a Biometric Sensor Function Provider (BSFP) [ISO_19784-4] on the device side. The interface between BSP and BSFP is standardised through BioSFPI [ISO_19784-4]. Additional specifications for the use of BSFPs and the Sensor Function Provider Interface (SFPI) are made within this Technical Guideline. Section 2.3 specifies some additional parameters for fingerprint sensors and further communication requirements when using BSFPs.

The BioAPI service provider interface is REQUIRED for conformance testing. Interfaces and specifications for additional input data for conformance testing are described in part 3 of BSI-TR 03122.

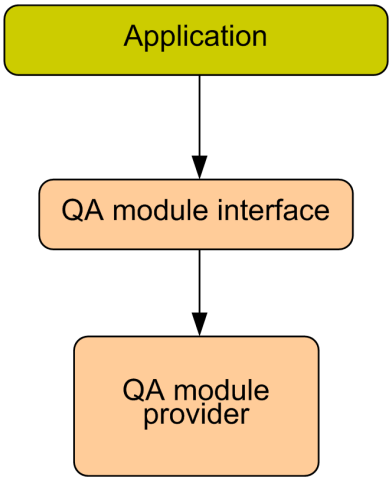

Figure 2.2. Software Architecture for Biometric Quality Assurance

For the purpose of quality assurance of biometric data no open standards or interfaces are available. Thus, a flexible provider-based architecture is REQUIRED for quality assurance. As shown in Figure 2.2, a common QA module interface SHALL be used which chooses a QA module provider for quality assurance. The actual quality check of biometric data is implemented within the QA module provider. This allows for use of multiple QA module simultaneously.

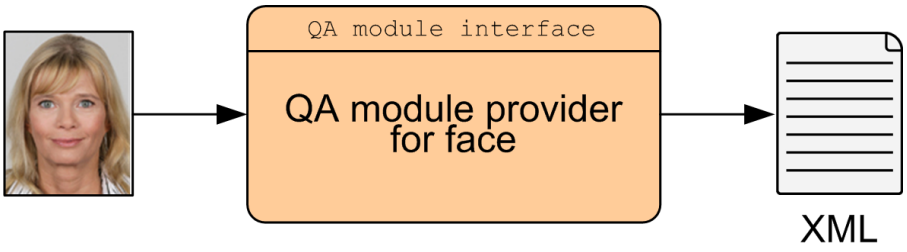

Figure 2.3. QA Module Provider Interface for Face Biometrics

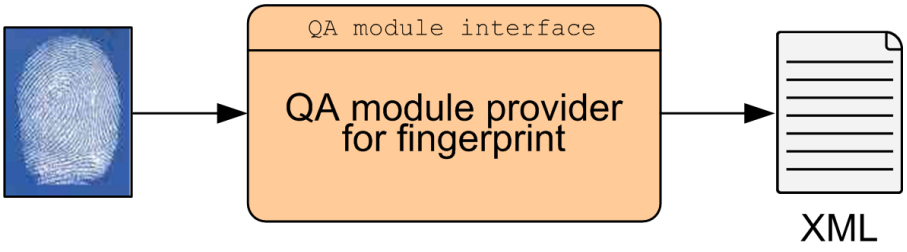

Figure 2.4. QA Module Provider Interface for Fingerprint Biometrics

It is up to the implementer to specify this interface. Providers SHALL return the quality assurance results in a way that they can easily be incorporated in the logging data of the specific application. One important requirement is to have an interface that supports various QA module providers which accept binary biometric data as input and return the quality assurance results in XML format according to the valid Function Module QA of the according profile (see Figure 2.3, Figure 2.4 for a schematic illustration).

2.1.1. Working Example

An enrolment example using BioAPI can be found in the file BioAPI_Demo_Application.cpp.

2.2. BSP Calling Profiles

The following calling conventions apply to BSPs used in the different Application Profiles.

2.2.1. Calling Requirements for all BSPs

Each BSP SHALL support the BioSPI_ControlUnit method call as defined in Table 2.1.

BioSPI_ControlUnit		
Parameter	**Type**	**Value/Description**
BioAPI_HANDLE BSPHandle	Input	Regular usage as defined by the BioAPI 2.0 standard.
BioAPI_UNIT_ID UnitID	Input	Regular usage as defined by the BioAPI 2.0 standard. Represents the ID of the BioAPI Unit.
uint32_t ControlCode	Input	This parameter SHALL denote the control code as described by the corresponding Function Module Coding.
const BioAPI_DATA *InputData	Input	Pointer to the input data structure related to the given ControlCode. This parameter SHALL denote the input data as described by the Function Module Coding.
BioAPI_DATA *OutputData	Output	Regular usage as defined by the BioAPI 2.0 standard.

Table 2.1. Calling Requirements for BioSPI_ControlUnit

2.2.2. Calling Requirements for Capture BSPs

Each Capture BSP SHALL support the BioSPI_Capture method call as defined in Table 2.2.

BioSPI_Capture		
Parameter	**Type**	**Value/Description**
BioAPI_HANDLE BSPHandle	Input	Regular usage as defined by the BioAPI 2.0 standard.
BIOAPI_BIR_PURPOSE Purpose	Input	• BioAPI_PURPOSE_ENROLL_FOR_VERIFICATION_ONLY (for pure verification enrolment) • BioAPI_PURPOSE_ENROLL (for multi-purpose enrolment) • BioAPI_PURPOSE_VERIFY (for later use with a Verification Engine BSP) • BioAPI_PURPOSE_IDENTIFY (for later use with a Identification Engine BSP)
BioAPI_BIR_SUBTYPE Subtype	Input	BioAPI_NO_SUBTYPE_AVAILABLE
const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat	Input	This parameter SHALL denote format owner and format type of the encoding as described by the Function Module Coding. These values are registered and published by the Federal Office for Information Security (BSI).
BioAPI_BIR_HANDLE *CapturedBIR	Output	Handle to the result data, encoded as a Biometric Information Record (BIR).
int32_t Timeout	Input	Regular usage as defined by the BioAPI 2.0 standard.

BioSPI_Capture		
Parameter	**Type**	**Value/Description**
BioAPI_BIR_HANDLE *AuditData	Output	This OPTIONAL parameter is not covered by this guideline, it is left to the implementation of the BSP to deliver audit data.

Table 2.2. Calling Requirements for BioSPI_Capture

2.2.3. Calling Requirements for Verification Engine BSPs

Verification Engine BSPs SHALL support the BioSPI_CreateTemplate (as defined in Table 2.3), BioSPI_Process (as defined in Table 2.4) and BioSPI_VerifyMatch (as defined in Table 2.5) method calls.

BioSPI_CreateTemplate		
Parameter	**Type**	**Value/Description**
BioAPI_HANDLE BSPHandle	Input	Regular usage as defined by the BioAPI 2.0 standard
const BioAPI_INPUT_BIR *CapturedBIR	Input	The previously captured data
const BioAPI_INPUT_BIR *ReferenceTemplate	Input	Unused
const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat	Input	This parameter SHALL denote format owner and format type of the encoding. The format is defined by the according vendor of the verification component.
BioAPI_BIR_HANDLE *NewTemplate	Output	The newly generated template
BioAPI_DATA *Payload	Input	Regular usage as defined by the BioAPI 2.0 standard
BioAPI_UUID *TemplateUUID	Output	Regular usage as defined by the BioAPI 2.0 standard

Table 2.3. Calling Requirements for BioSPI_CreateTemplate

BioSPI_Process		
Parameter	**Type**	**Value/Description**
BioAPI_Handle BSPHandle	Input	Regular usage as defined by the BioAPI 2.0 standard
const BioAPI_INPUT_BIR *CapturedBIR	Input	The previously captured data
const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat	Input	This parameter SHALL denote format owner and format type of the encoding. The format is defined by the according vendor of the verification component.
BioAPI_BIR_HANDLE *ProcessedBIR	Input	Regular usage as defined by the BioAPI 2.0 standard.

Table 2.4. Calling Requirements for BioSPI_Process

BioSPI_VerifyMatch		
Parameter	**Type**	**Value/Description**
BioAPI_Handle BSPHandle	Input	Regular usage as defined by the BioAPI 2.0 standard

BioSPI_VerifyMatch		
Parameter	**Type**	**Value/Description**
BioAPI_FMR MaxFMRRequested	Input	Regular usage as defined by the BioAPI 2.0 standard; value as defined by the corresponding Function Module Comparison.
const BioAPI_INPUT_BIR *ProcessedBIR	Input	The BIR to be verified
const BioAPI_INPUT_BIR *ReferenceTemplate	Input	The reference to be verified against
BioAPI_BIR_HANDLE *AdaptedBIR	Output	Unused
BioAPI_BOOL *Result	Output	The verification result
BioAPI_FMR *FMRAchieved	Output	The achieved FMR. Coarse scoring SHALL not be used.
BioAPI_DATA *Payload	Output	Regular usage as defined by the BioAPI 2.0 standard.

Table 2.5. Calling Requirements for BioSPI_VerifyMatch

2.2.4. Calling Requirements for Verification BSPs

Verification Engine BSPs SHALL support the BioSPI_Enroll (as defined in Table 2.6) and BioSPI_Verify (as defined in Table 2.7) method calls.

BioSPI_Enroll		
Parameter	**Type**	**Value/Description**
BioAPI_Handle BSPHandle	Input	Regular usage as defined by the BioAPI 2.0 standard
BioAPI_BIR_PURPOSE Purpose	Input	BioAPI_PURPOSE_ENROLL_FOR_VERIFICATION_ONLY
BioAPI_BIR_SUBTYPE Subtype	Input	BioAPI_NO_SUBTYPE_AVAILABLE
const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat	Input	This parameter SHALL denote format owner and format type of the encoding. The format is defined by the according vendor of the verification component.
const BioAPI_INPUT_BIR *ReferenceTemplate	Input	Unused
BioAPI_BIR_Handle *NewTemplate	Output	The newly generated template
const BioAPI_DATA *Payload	Input	Regular usage as defined by the BioAPI 2.0 standard.
int32_t Timeout	Input	Regular usage as defined by the BioAPI 2.0 standard.
BioAPI_BIR_HANDLE *AuditData	Output	This OPTIONAL parameter is not covered by this guideline, it is left to the implementation of the BSP to deliver audit data.
BioAPI_UUID *TemplateUUID	Output	Regular usage as defined by the BioAPI 2.0 standard.

Table 2.6. Calling Requirements for BioSPI_Enroll

BioSPI_Verify		
Parameter	**Type**	**Value/Description**
BioAPI_Handle BSPHandle	Input	Regular usage as defined by the BioAPI 2.0 standard.
BioAPI_FMR MaxFMRRequested	Input	Regular usage as defined by the BioAPI 2.0 standard; value as defined by the corresponding Function Module Comparison.
const BioAPI_INPUT_BIR *ReferenceTemplate	Input	The reference to be verified against
BioAPI_BIR_SUBTYPE Subtype	Input	BioAPI_NO_SUBTYPE_AVAILABLE
BioAPI_BIR_HANDLE *AdaptedBIR	Output	Unused
BioAPI_BOOL *Result	Output	The verification result
BioAPI_FMR *FMRAchieved	Output	The (best) achieved FMR. Coarse scoring SHALL not be used.
BioAPI_DATA *Payload	Output	Regular usage as defined by the BioAPI 2.0 standard.
int32_t Timeout	Input	Regular usage as defined by the BioAPI 2.0 standard.
BioAPI_BIR_HANDLE *AuditData	Output	In this parameter, the BSP SHALL return the XML verification information as defined by the corresponding Function Module Coding. Especially, the probes (live images) for all comparisons – as used by the internal comparator – SHALL be included.

Table 2.7. Calling Requirements for BioSPI_Verify

2.3. Further Specifications for SFPI and BSFPs

2.3.1. Additional SFPI Communication

Certain biometric sensors require information about the chosen subtype of the biometric modality to be acquired. As an example, especially in terms of auto-capture functionality, multi-fingerprint sensors need to know when to trigger auto-capture. This functionality is mainly based on the information of how many fingers should be placed on the sensor surface. Through the available Sensor Function Provider Interface (SFPI) standardised in [ISO_19784-4] such information exchange is not provided by given functions. Hence, a BioSFPI_ControlUnit call is REQUIRED to inform the selected BSFP which subtypes were chosen during the BSP function calls. Table 2.8 defines the according ControlCode; the necessary XML input data structure is defined as "bioapi-bsfp" by the XML schema file "bioapi.xsd".

Organisation Name	ControlCode	Input data
Federal Office for Information Security (BSI)	0x4000fe01	bioapi-bsfp

Table 2.8. Calling Requirements for BioSPI_Verify

An example can be found in the file "bioapi-bsfp.xml".

2.3.2. Additional BSFP Parameters

If biometric sensors are encapsulated within a Biometric Sensor Function Provider (BSFP) according to the biometric imaging sensor units specification in Annex A of [ISO_19784-4], further information about the biometric sensor MAY be needed in the Biometric Service Provider (BSP) or in the application using the BSP. Such

information of the BSFP MAY be needed by the BSP to determine which available and installed BSFPs might be appropriate for the REQUIRED application scenario. Thus, further information of biometric devices can be stored in the *AdditionalParameters* element of the *BioSFPI_BSFPImagePropertySchema* stored in the BioAPI component registry during installation of the BSFP.

In the following, additional parameters and requirements for fingerprint sensors encapsulated within BSFPs are specified.

2.3.2.1. Fingerprint Sensors

Fingerprint sensors encapsulated within BSFPs are REQUIRED to support hardware- or software-based auto-capture functionality. Furthermore, a BSFP SHALL support the function *BioSFPI_GetPackets* for fingerprint data exchange between the calling BSP and the selected BSFP. By sending "last packet" during the call of this function, triggered auto-capture of the BSFP SHALL be signalled to the BSP (see [ISO_19784-4]).

The function *BioSFPI_GetPackets* will only transmit the final captured image. For displaying live stream images in the BSP GUI callbacks SHALL be used via *BioSFPI_SetGUICallbacks*. GUI state callbacks can be used for transmitting further information. Additionally, the BSP SHALL respond to received GUI state callback messages. The response BioAPI_CAPTURE_SAMPLE SHALL be sent if the BSP manually triggers capturing. The BioAPI_CONTINUE response SHALL tell the BSFP that the capture process SHALL continue.

Certain fingerprint sensors return information about location and positioning errors. This information SHALL be provided to the BSP by using the Message parameter of the GUI state callback. Following error codes can be transmitted via the Message parameter to signal location errors:

- BioAPIERR_LOCATION_TOO_LEFT: The finger was located too far left on the device.
- BioAPIERR_LOCATION_TOO_RIGHT: The finger was located too far right on the device.
- BioAPIERR_LOCATION_TOO_FORWARD: The finger was located too far forward on the device.
- BioAPIERR_LOCATION_TOO_BACKWARD: The finger was located too far backward on the device.
- BioAPIERR_INVALID_LENGTHWISE_POSITION: The finger has an invalid lengthwise position on the device, i.e. the length of the fingerprint is too small.
- BioAPIERR_INVALID_CROSSWISE_POSITION: The finger has an invalid crosswise position on the device, i.e. the width of the fingerprint is too small (e.g. because the finger was not placed in the middle of the sensor surface).

This information can be used in the BSP to display guidance information to assist the user in correct placement of the fingers.

For storing additional information in the *AdditionalParameters* element of the *BioSFPI_BSFPImagePropertySchema* of the BFP schema of the installed BSFP, the structure TR03121_BSFP_IMAGE_PROPERTY_SCHEMA_ADDITIONAL_PARAMETERS_FINGERPRINT is specified (see definition of additional parameters structure for fingerprint sensors below). It contains following information:

Parameter	Description
AdditionalParametersID	This UUID describes the type and content of additional parameters being used within this AdditionalParameters element. For fingerprint sensors in the scope of this Technical Guideline it SHALL be the value ec7d9afb-45a0-4490-8e26-5d8bc3e71671
MaximumNumberOfSupportedFingers	This element describes the maximum number of supported fingers being captured at once by the fingerprint sensor.
SensorType	This element describes the sensor technology (optical with FTR, optical without FTR, capacitive, thermic, ultrasonic, radio-frequency, pressure sensitive, other)

Parameter	Description
SensorArchitecture	This element describes the sensor architecture (swipe, area sensors, other)
LifeFingerDetectionSupported	This element describes if the sensor is capable of detecting life fingers.
AcquisitionMethod	This element describes which acquisition method the sensor is capable of (flat fingerprints, rolled fingerprints, other)
SensorDPI	This element describes the maximum scanning resolution in dots per inch (dpi)
SensorAreaWidth	This element describes the width of the sensor area surface (in millimetres)
SensorAreaHeight	This element describes the height of the sensor area surface (in millimetres)
AutoCaptureSupported	This element describes if the sensor supports either hardware- or software-based auto-capture functionality
SensorCertification	This element contains information about certifications of the fingerprint sensor (e.g. FBI Appendix F, PIV, BSI certified, other)

Table 2.9. List of Additional Parameters

The C header definition is given in the file BioSFPI_BSFPImagePropertySchema_AdditionalParameters.h

2.3.3. Serialisation of Additional Parameters Structure

Serialisation is necessary for writing the above mentioned data structure into the *AdditionalParameters* element of the *BioSFPI_BSFPImagePropertySchema*. Furthermore, de-serialisation is necessary for the other way around. Below, appropriate functions are defined for this purpose. Following Annex D.2 of the BioAPI specification [ISO_19784-1], functions needed for serialisation and de-serialisation of Biometric Information Records are used, accordingly.

A normative reference implementation of the serialisation and de-serialisation process based on the conversion function given in [ISO_19784-1], Annex D.2 is given in the files BioSFPI_BSFPImagePropertySchema_AdditionalParameters_ConversionFunctions.{h,cpp}.

List of Abbreviations

Abbreviation	Description
BSP	Biometric Service Provider
HLBS	High Level Biometric Services

Bibliography

[ISO_19784-1] *ISO/IEC 19784-1:2006 "Information technology – Biometric application programming interface – Part 1: BioAPI specification".*

[ISO_19784-4] *ISO/IEC 19784-4:2011: "Information technology – Biometric application programming interface – Part 4: Biometric sensor function provider interface".*