INTERNATIONAL STANDARD

**ISO/IEC 30106-4**

First edition
2019-09

# Information technology — Object oriented BioAPI —

## Part 4: C++ implementation

*Technologies de l'information — Objet orienté BioAPI —*

*Partie 4: Implémentation C++*



Reference number
ISO/IEC 30106-4:2019(E)



ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: 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.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

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. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see http://patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html.



This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 37, *Biometrics*.



A list of all parts in the ISO/IEC 30106 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

# Introduction

In this document an application programming interface expressed in C++ language is specified. C++ is a basic, general-purpose, object-oriented programming language that is used in most computer-based platforms and operating systems.

One of the advantages of using C++ is that, as it is supported by most computer platforms, the development, in source code, may be fully (or at least to a great part) supported from one platform to another.

C++ is a programming language standardized by ISO/IEC 14882, and most development platforms allow its use in conjunction to other programming languages and frameworks.

This document is drafted to make a clear and usable interpretation of ISO/IEC 30106-1, when using C++ programming language.

# Information technology — Object oriented BioAPI —

# Part 4:
# C++ implementation

## 1 Scope

This document specifies an interface of a BioAPI C++ framework and BioAPI C++ BSP which will mirror the corresponding components specified in ISO/IEC 30106-1. The semantic equivalence of this document will be maintained with ISO/IEC 30106-2 (Java implementation) and ISO/IEC 30106-3 (C# implementation). In spite of the differences in actual parameters passed between functions, the names and interface structure are the same.

## 2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.



ISO/IEC 30106-1, *Information technology — Object oriented BioAPI — Architecture*

ISO/IEC 30106-2, *Information technology — Object oriented BioAPI — Part 2: Java implementation*

ISO/IEC 30106-3, *Information technology — Object oriented BioAPI — Part 3: C# implementation*



ISO/IEC 14882, *Programming languages — C++*

## 3 Terms and definitions

No terms and definitions are listed in this document.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

## 4 BioAPI C++ general requirements

All classes defined in the scope of this document shall have a constructor with all its properties as parameters, as well as a destructor that will free all allocated memory. In addition, the definition of each of the classes may contain further requirements for the constructor.

The destructor shall fullfill at least, the same requirements that the corresponding Dispose() method defined in ISO/IEC 30106-2 and ISO/IEC 30106-3.

In addition, most implementations shall declare the classes as exportable. Therefore, in this document a constant called BIOAPI is defined, allowing exportation of the classes.

EXAMPLE In Windows the definition of BIOAPI is: `#define BIOAPI __declspec(dllexport)`

To illustrate the use of the specification given in this document, refer to Annex A.

# 5 Data types and constants

## 5.1 Basic data types

### 5.1.1 Enumerations

| 5.1.1.1 BiometricSubtype | |
|---|---|
| Description: | Subtype of the biometric data used (e.g. which finger used in finger modalities). When transferring this information into/from a binary format, the Biometric Subtype constants defined in ISO/IEC 30106-1 shall be used. |
| Enum Constant Summary: | — *NoValueAvailable*<br>— *Left*<br>— *Right*<br>— *LeftThumb*<br>— *LeftIndexFinger*<br>— *LeftMiddleFinger*<br>— *LeftRingFinger*<br>— *LeftLittleFinger*<br>— *RightThumb*<br>— *RightIndexFinger*<br>— *RightMiddleFinger*<br>— *RightRingFinger*<br>— *RightLittleFinger*<br>— *LeftPalm*<br>— *LeftBackOfHand*<br>— *LeftWrist*<br>— *RightPalm*<br>— *RightBackOfHand*<br>— *RightWrist* |

| 5.1.1.2 BiometricType | |
|---|---|
| Description: | Type of the biometric data used (e.g. modality). When transferring this information into/from a binary format, the Biometric Type constants defined in ISO/IEC 30106-1 shall be used. |
| Enum Constant Summary: | — *NoValueAvailable*<br>— *MultipleBiometricTypes*<br>— *Face*<br>— *Voice*<br>— *Finger*<br>— *Iris*<br>— *Retina*<br>— *HandGeometry*<br>— *SignatureOrSign*<br>— *Keystroke*<br>— *LipMovement*<br>— *Gait*<br>— *Vein*<br>— *DNA*<br>— *Ear*<br>— *Foot*<br>— *Scent* |



| 5.1.1.3 BIRDatabaseAccess | |
|---|---|
| Description: | Defines the access mode to the database |
| Enum Constant Summary: | — *Read* – access mode which allows only retrieval of records.<br>— *ReadWrite* – access mode which allows addition, deletion and retrieval of records.<br>— *Write* – access mode which allows addition and deletion of records, but retrieval is not allowed |

| 5.1.1.4 BSPSchemaOperations | |
|---|---|
| Description: | Enumerates the different operations that a BSP can offer to the biometric application (see 5.6) |
| Enum Constant Summary: | — *CalibrateSensor (0x00020000)*<br>— *Capture (0x00000004)*<br>— *CheckQuality (0x00080000)*<br>— *ControlUnit (0x00400000)*<br>— *CreateTemplate (0x00000008)*<br>— *CreateTemplateWithAuxBIR (0x00000020)*<br>— *EnableEvents (0x00000001)*<br>— *Enrol (0x00000100)*<br>— *GetIndicatorStatus (0x00010000)*<br>— *Identify (0x00000080)*<br>— *IdentifyAggregate (0x00000400)*<br>— *PresetIdentifyPopulation (0x00001000)*<br>— *Process (0x00000010)*<br>— *ProcessWithAuxBIR (0x01000000)*<br>— *QueryBFPs (0x00200000)*<br>— *QueryUnits (0x00100000)*<br>— *Security (0x10000000)*<br>— *SetIndicatorStatus (0x00008000)*<br>— *SetPowerMode (0x00004000)*<br>— *Verify (0x00000040)*<br>— *VerifyAggregated (0x00000200)*<br>— *VerifyWithAuxBIR (0x02000000)* |

| 5.1.1.5 BSPSchemaOptions | |
|---|---|
| Description: | Enumerates the different options that can a handle a BSP (see 5.6) |
| Enum Constant Summary: | — *Adaptation (0x00000800)*<br>— *AppGUI (0x00000010)*<br>— *AchiveBFP (0x00020000)*<br>— *Binning (0x00001000)*<br>— *BirEncrypt (0x00000200)*<br>— *BirSign (0x00000100)*<br>— *CaptureMultiple (0x00400000)*<br>— *CoarseScores (0x00100000)*<br>— *ComparisonBFP (0x00040000)*<br>— *GUIProgressEvents (0x00000020)*<br>— *IdentifyIndicator (0x00200000)*<br>— *OCC (0x00004000) (on-card comparison, formerly known as MOC)*<br>— *AdditionalData (0x00000080)*<br>— *ProcessingBFP (0x00080000)*<br>— *ProcessMultiple (0x00800000)*<br>— *QualityIntermediate (0x00000004)*<br>— *QualityProcessed (0x00000008)*<br>— *QualityRaw (0x00000002)*<br>— *Raw (0x00000001)*<br>— *SelfContainedDevice (0x00002000)*<br>— *SensorBFP (0x00010000)*<br>— *SourcePresent (0x00000040)*<br>— *SubtypeToCapture (0x00008000)*<br>— *TemplateUpdate (0x00000400)*<br>— *Disabilities*<br>— *PADFeature* |

| 5.1.1.6 EventKind | |
|---|---|
| Description: | Defines the kind of sources that can originate an event |
| Enum Constant Summary: | — *Insert (0x00000001)*<br>— *Remove (0x00000002)*<br>— *Fault (0x00000004)*<br>— *SourcePresent (0x00000008)*<br>— *SourceRemoved (0x00000010)* |

| 5.1.1.7 Facility | |
|---|---|
| Description: | Defines originator of the error in an exception (see 10.1) |
| Enum Constant Summary: | — *Framework – The error was reported by the framework component.*<br>— *BSP – The error was reported by the biometric service provider.*<br>— *Unit – The error was reported by the biometric unit* |



| 5.1.1.8 GUIEnrolType | |
|---|---|
| Description: | Indicates the enrol type of a BSP (see 10.2) |
| Enum Constant Summary: | — *TestVerify*<br>— *MultipleCapture* |



| 5.1.1.9 GUIMoment | |
|---|---|
| Description: | Determines the moment when the processing of an operation is at the time of calling a GUI callback function (see 10.2) |
| Enum Constant Summary: | — *BeforeStart*<br>— *AfterEnd* |

| 5.1.1.10 GUIOperation | |
|---|---|
| Description: | Determines the operation being performed when using GUI callback functions (see 10.2) |
| Enum Constant Summary: | — *Capture*<br>— *Enrol*<br>— *Identify*<br>— *Verify* |

| 5.1.1.11 GUIResponse | |
|---|---|
| Description: | Enumeration of the possible actions to be performed bya BSP after a GUI event notification callback made by the BSP has returned control to the BSP (see 10.2) |
| Enum Constant Summary: | — *CycleStart*<br>— *CycleRestart*<br>— *Default*<br>— *OpComplete*<br>— *OpCancel*<br>— *ProgressContinue*<br>— *ProgressAbort*<br>— *Recapture*<br>— *SubOpStart*<br>— *SubOpNext* |



| 5.1.1.12 GUISuboperation | |
|---|---|
| Description: | An enumeration of the possible types of suboperations performed by a BSP during an operation, to be reported to the application in GUI event notifications (see 10.2) |
| Enum Constant Summary: | — *Capture*<br>— *CreateTemplate*<br>— *Identify*<br>— *Process*<br>— *Verify* |

| 5.1.1.13 ProcessedLevel | |
|---|---|
| Description: | Determines the level of processing of the BIR |
| Enum Constant Summary: | — *Intermediate*<br>— *Processed*<br>— *Raw* |

| 5.1.1.14 Purpose | |
|---|---|
| Description: | Defines the purpose for which the BIR or process is intended |
| Enum Constant Summary: | — *Verify*<br>— *Identify*<br>— *Enrol*<br>— *EnrolForVerificationOnly*<br>— *EnrolForIdentificacionOnly*<br>— *Audit*<br>— *Decide*<br>— *NoPurposeAvailable* |



| 5.1.1.15 ResultOptions | |
|---|---|
| Description: | Defines the request to some BioAPI methods, to provide additional results to the originally defined (e.g. see 6.3.2). |
| Enum Constant Summary: | — *RequestAdaptedBIR – Request that a BIR be constructed by adapting the reference template using the processed BIR that is the input to the biometric verification.*<br>— *RequestAdditionalData – Request that the additionalData should be returned upon successful verification*<br>— *RequestAdditionalData – Request additional data to be used, for example, in an auditing process.* |

| 5.1.1.16 SecurityOptionsType | |
|---|---|
| Description: | Defines which security options are supported by the BioAPI_Unit |
| Enum Constant Summary: | — *Encryption (0x00000001) – Indicates that the BioAPI Unit supports encryption.*<br>— *MAC (0x00000002) – Indicates that the BioAPI Unit supports MAC generation.*<br>— *DigitalSignature (0x00000004) – Indicates that the BioAPI Unit supports digital signature.*<br>— *ACBioGenerationWithMAC (0x00000010) – Indicates that the BioAPI Unit supports ACBio generation using MAC.*<br>— *ACBioGenerationWithDigitalSignature (0x00000020) – Indicates that the BioAPI Unit supports ACBio generation using digital signature.* |

| 5.1.1.17 UnitCategoryType | |
|---|---|
| Description: | List the different categories for a BioAPI_Unit |
| Enum Constant Summary: | — *Archive* – the unit manages BSP's BIR database. (0x00000001)<br>— *Comparison* – the unit is the collection of comparison algorithms. (0x00000002)<br>— *Processing* – the unit is the collection of processing algorithms. (0x00000004)<br>— *Sensor* – the unit manages hardware sensor (0x00000008)<br>— *QualityAssessment* – the unit is the collection of the different quality assessment processes (0x00000010) |



| 5.1.1.18 UnitIndicatorStatus | |
|---|---|
| Description: | Defines possible values for the indicator status |
| Enum Constant Summary: | — *Accept*<br>— *Busy*<br>— *Failure*<br>— *Ready*<br>— *Reject* |