INTERNATIONAL STANDARD

ISO/IEC 30106-2

Second edition
2020-11

Information technology — Object oriented BioAPI —

Part 2: Java implementation

Technologies de l'information — Objet orienté BioAPI —

Partie 2: Mise en oeuvre Java

Reference number
ISO/IEC 30106-2:2020(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
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*.

This second edition cancels and replaces the first edition (ISO/IEC 30106-2:2016), which has been technically revised.

The main changes compared to the previous edition are as follows:

— correction of typing errors;

— addition of AnalyseQuality method.

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

This document specifies an application programming interface expressed in Java language. Java is intended to be a simple, general-purpose, object-oriented programming language that is aimed at enabling programmers to quickly build a wide range of applications for multiple platforms.

This Java implementation allows an easy use of Java BSPs, Java-based application servers or Java applets. It is therefore the best way to write desktop and web applications/services. This document provides an advanced and well-designed remote framework.

Although the best practices of Java programming state that variables should be written in lowercase letters, in the case of symbols, such as BSP or BFPs, they have been retained in uppercase letters.

Information technology — Object oriented BioAPI —

Part 2:
Java implementation

1 Scope

This document specifies an interface of a BioAPI Java framework and BioAPI Java BSP, which will mirror the corresponding components, specified in ISO/IEC 30106-1. The semantic equivalent of ISO/IEC 30106-1 is maintained in this document.

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 10646, *Information technology — Universal Coded Character Set (UCS)*

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

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 Java package structure

4.1 Overall structure

The BioAPI Java interface is divided into several packages. The following is the package structure.

— Package org.bioapi: Contains functionality to manage units, BSPs, BFPs, the Framework and Applications.

— Package org.bioapi.data: Contains all the data structures.

4.2 Package org.bioapi

4.2.1 Package description

This package contains all the components responsible for managing and executing the functionality of BioAPI. Component Registry interface is also defined in this package.

4.2.2 Structure

The description of this package is given explaining a bottom-up structure. In Clause 6, the interfaces that need to be implemented for each of the Unit types are explained. It is important to note that such interfaces do not refer to an implemented class by itself, as the accessible class will be either the Biometric Service Provider (BSP) or the Biometric Function Provider (BFP), but the specifications in this clause are common to the methods and properties to be added to the implemented BSP and/or BFP classes.

This will be followed by the specification of the implementation of the BFP (Clause 7) and BSP (Clause 8) interfaces. These two interfaces provide the lower layer interoperability level, equivalent to the SPI and BFPI interfaces in ISO/IEC 19784-1.

The higher layer of interoperability level is provided by the specification of the Framework (Clause 9, with the Framework Interface and the Component Registry) and the Application interaction (Clause 10, with the specification of the Exceptions and Callback functions). This provides the equivalence to the API interface in ISO/IEC 19784-1.

4.3 Package org.bioapi.data

4.3.1 Package description

This package contains all data structures required for the implementation of OO BioAPI.

4.3.2 Structure

Several data structures are provided to comply with the requirements specified by this document. The whole org.bioapi.data package is specified in Clause 5, where all required classes and enumerations are defined. This has to be complemented to the constants defined in ISO/IEC 30106-1.

5 Data types and constants

5.1 Class ACBioParameters

5.1.1 Description

Provides the information which is used to generate ACBio instances.

5.1.2 Method summary

5.1.2.1 int[] getChallenge()

Description:	Return the challenge from the validator of a biometric verification when ACBio is used. This value shall be sent to the field controlValue of type ACBioContentInformation in ACBio instances.
Return Value:	The challenge from the validator of a biometric verification when ACBio is used.

5.1.2.2 int[] getInitialBPUIOIndexOutput()

Description:	Return the initial value of BPU IO index which is to be assigned to the output from the BioAPI Unit, BFP, or BSP when the ACBio instances are generated. The range between InitialBPUIOIndexOutput and SupremumBPUIOIndexOutput shall be divided into the number of BSP Units and BFPs which are inside the BSP, and assigned to the BSP Units and BSPs.
Return Value:	The initial value of BPU IO index.

5.1.2.3 int[] getSupremumBPUIOIndexOutput()

Description:	Return the supremum of BPU IO indexes which are to be assigned to the output from the BioAPI Unit, BFP, or BSP when the ACBio instances are generated.
Return Value:	The supremum of BPU IO index.

5.2 Class BFPListElement

5.2.1 Description

Identifies a BFP by category and UUID. A list is returned by a BSP when queried for the installed BFPs that it supports.

5.2.2 Method summary

5.2.2.1 UUID getBFPID()

Description:	Return the UUID assigned to the BFP.
Return Value:	The UUID assigned to the BFP.

5.2.2.2 UnitCategoryType getUnitCategory()

Description:	Return the category of the units.
Return Value:	The category of the units.

5.2.2.3 void setBFPID(UUID bfpID)

Description:	Set the UUID assigned to the BFP.
Parameters:	— *bfpID:* The UUID assigned to the BFP.

5.2.2.4 void setUnitCategory(UnitCategoryType unitCategory)

Description:	Set the category of the units.
Parameters:	— *unitCategory:* The category of the units.

5.3 Class BFPSchema

5.3.1 Description

Represents the record in the component registry that defines the properties of the BFP installed in the system.

5.3.2 Method summary

5.3.2.1 String getBFPDescription()

Description:	Return a NULL-terminated string containing a text description of the BFP.
Return Value:	A NULL-terminated string containing a text description of the BFP.

5.3.2.2 Vector<RegistryID> getBFPSupportedFormats()

Description:	Return a list the data formats that are supported by the BFP.
Return Value:	A list the data formats that are supported by the BFP.

5.3.2.3 UUID getBFPUUID()

Description:	Return the BFP UUID.
Return Value:	The BFP UUID.

5.3.2.4 Vector<BiometricType> getFactorsMask()

Description:	Return a list of the biometric types supported by the BFP.
Return Value:	A list of the biometric types supported by the BFP.

5.3.2.5 byte[] getFWProperty()

Description:	Return the address and length of a memory buffer containing the BFP property. The format and content of the BFP property can either be specified by a vendor or can be specified in a related standard.
Return Value:	The address and length of a memory buffer containing the BFP property.

5.3.2.6 UUID getFWPropertyID()

Description:	Return the UUID of the format of the following BFP property.
Return Value:	UUID of the format of the following BFP property.

5.3.2.7 String getPath()

Description:	Return a pointer to a NULL-terminated string containing the path of the file containing the BFP executable code, including the filename. The path may be a URL. This string shall consist of ISO/IEC 10646 characters encoded in UTF-8 (see ISO/IEC 10646:2017, Annex D). When BFPSchema is used within a function call, the component that receives the call allocates the memory for the Path schema element and the calling component frees the memory.
Return Value:	A pointer to a NULL-terminated string containing the path of the file containing the BFP executable code, including the filename.

5.3.2.8 String getProductVersion()

Description:	Return the version string of the BFP software.
Return Value:	The version string of the BFP software.

5.3.2.9 String getSpecVersion()

Description:	Return the major/minor version number of the BioAPI specification to which the BFP was implemented.
Return Value:	The major/minor version number of the BioAPI specification to which the BFP was implemented.

5.3.2.10 UnitCategoryType getUnitCategory()

Description:	Return the category of the BFP identified by the BFP UUID.
Return Value:	The category of the BFP identified by the BFP UUID.

5.3.2.11 String getVendor()

Description:	Return a NULL-terminated string containing the name of the BFP vendor.
Return Value:	A NULL-terminated string containing the name of the BFP vendor.

5.4 Class BIR

5.4.1 Description

Represents BIRs (Biometric Information Records). It supports ISO/IEC 19785-1 definitions, both for Simple-BIRs or for Complex-BIRs. The specification of the patron format that shall be used is given in ISO/IEC 30106-1.

5.4.2 Method summary

5.4.2.1 void birFromByteArray(byte[] record)

Description:	Fills in the BIR data from a byte array coded as CBEFF record, as indicated in the relevant clauses of ISO/IEC 19785-3 and ISO/IEC 19785-4.
Parameters:	— *record*: The byte array containing the CBEFF record.
Exception:	If the input parameters are invalid, the format is not supported or operation fails due to error, BioAPIException (see 10.1).

5.4.2.2 byte[] birToByteArray()

Description:	Serializes a BIR record so as to provide it as a byte array representing the CBEFF information.
Return Value:	The byte array containing the CBEFF information.
Exception:	If the input parameters are invalid, the format is not supported or operation fails due to error, BioAPIException (see 10.1).

5.4.2.3 void destroy()

Description:	Removes all the information in the current BIR, leaving it empty for a next use.
Exception:	None.

5.4.2.4 BiometricSubtype getBDBBiometricSubtype()

Description:	Return the BDB biometric subtype.
Return Value:	The BDB biometric subtype.

5.4.2.5 BiometricType getBDBBiometricType()

Description:	Return the BDB biometric type.
Return Value:	The BDB biometric type.

5.4.2.6 byte[] getBDBChallengeResponse()

Description:	Return the BDB challenge response.
Return Value:	The BDB challenge response.

5.4.2.7 Date getBDBCreationDate()

Description:	Return the BDB creation date.
Return Value:	The BDB creation date.

5.4.2.8 byte[] getBDBData()

Description:	Return the BDB data array.
Return Value:	The BDB data array.

5.4.2.9 RegistryID getBDBFormat()

Description:	Return the format of the BDB data.
Return Value:	The format of the BDB data.

5.4.2.10 byte[] getBDBIndex()

Description:	Return the BDB index.
Return Value:	The BDB index.

5.4.2.11 ProcessedLevel getBDBProcessedLevel()

Description:	Return the BDB processed level.
Return Value:	The BDB processed level.

5.4.2.12 Purpose getBDBPurpose()

Description:	Return the BDB purpose.
Return Value:	The BDB purpose.

5.4.2.13 byte getBDBQuality()

Description:	Return the BDB quality.
Return Value:	The BDB quality.

5.4.2.14 Vector<Date> getBDBValidityPeriod()

Description:	Return the BDB validity period.
Return Value:	The BDB validity period.

5.4.2.15 Date getBIRCreationDate()

Description:	Return the BIR creation date.
Return Value:	The BIR creation date.

5.4.2.16 byte[] getBIRCreator()

Description:	Return the BIR creator array.
Return Value:	The BIR creator array.

5.4.2.17 byte[] getBIRIndex()

Description:	Return the BIR index.
Return Value:	The BIR index.

5.4.2.18 byte[] getBIRAdditionalData()

Description:	Return the BIR additionalData.
Return Value:	The BIR additionalData.

5.4.2.19 Vector<Date> getBIRValidityPeriod()

Description:	Return the BIR validity period.
Return Value:	The BIR validity period.

5.4.2.20 byte getCBEFFVersion()

Description:	Return the version of the CBEFF component.
Return Value:	The version of the CBEFF component.

5.4.2.21 RegistryID getPatronFormat()

Description:	Return the patron format.
Return Value:	The patron format.

5.4.2.22 byte getPatronHeaderVersion()

Description:	Return the header version.
Return Value:	The header version.

5.4.2.23 byte[] getSBData()

Description:	Return the Security Block data array.
Return Value:	The Security Block data array.