INTERNATIONAL STANDARD

ISO/IEC 30106-1

First edition
2016-03-15

AMENDMENT 1
2019-05

Information technology — Object oriented BioAPI —

Part 1: **Architecture**

AMENDMENT 1: Additional specifications and conformance statements

Technologies de l'information — Objet orienté BioAPI —

Partie 1: Architecture

AMENDEMENT 1: Spécifications et déclarations de conformité complémentaires

Reference number
ISO/IEC 30106-1:2016/Amd.1:2019(E)

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.

Information technology — Object oriented BioAPI —

Part 1: **Architecture**

AMENDMENT 1: Additional specifications and conformance statements

Page 1, Clause 2 Normative references

Update the listing for ISO/IEC 19785-3 to the following:

ISO/IEC 19785, *Information technology — Common Biometric Exchange Formats Framework (CBEFF) — Part 3: Patron format specifications*

Page 2

Add new Clause 5 Conformance with the following text and renumber subsequent clauses.

Clause 5 Conformance

Those products that claim conformance with any of the parts in the ISO/IEC 30106 series, shall comply with the requirements stated in Annex A.

Page 8, Clause 6

Replace Clause 6 with the following:

Clause 6 Object Oriented BioAPI CBEFF Patron Formats

Object Oriented BioAPI is able to use biometric data coded as Self-Identifying BIRs, either Simple BIRs or Complex BIRs, following the structure and definition of CBEFF (i.e. ISO/IEC 19785). In particular, Object Oriented BioAPI shall use the following CBEFF Patron Formats:

— For Simple BIRs the patron format to be used is the one called "Self-identifying Tag-oriented Simple BIR", registered as Patron Format Owner 257 and Patron Format Type 12, and described in ISO/IEC 19785-3.

— For Complex BIRs the patron format to be used is the one called "Self-identifying Tag-oriented Complex BIR", registered as Patron Format Owner 257 and Patron Format Type 13, and described in ISO/IEC 19785-3.

The particular tagged format (e.g. TLV, XML or JSON) to be used will be determined by the application and/or platform used.

Page 29

Insert the following clause after 8.7, to become Clause 9.

Clause 9 Additional specifications

9.1 Standardized control codes for additional functionality

9.1.1 General

When a BSP or BFP requires the addition of further functionality to the one already defined in previous clauses, the method to be used is:

byte[] ControlUnit (int unitID, int controlCode, byte[] inputData)

The method can be used by different vendors, providing different functionalities. In order to minimize interoperability issues, the control codes shall be defined to allow both standardized behaviour, and proprietary functionality.

Therefore, control codes shall have the same format:

Code (in hex)	Description
00 01 XX XX	Standardized functionality for Archive units
00 02 XX XX	Standardized functionality for Processing units
00 03 XX XX	Standardized functionality for Comparison units
00 04 XX XX	Standardized functionality for Capture units
00 0X XX XX	RFU for SC37 standardized functionality
(01-FF) XX XX XX	Proprietary functionality (out of the scope of SC37 description)

9.1.2 Control codes for Archive Units

The following codes are standardized for Archive units:

Code	Input data	Output data	Description
00 01 00 01			

9.1.3 Control codes for Processing Units

The following codes are standardized for Archive units:

Code	Input data	Output data	Description
00 02 00 01	Byte threshold	00 – OK 01 – Not accepted by unit 02 – Not available in unit 03 – Other error	Change the quality threshold to accept a BIR as in input for Processing. The algorithm may have its own threshold and also rules to accept new thresholds (e.g. not accepting thresholds below a determined minimum value).

9.1.4 Control codes for Comparison Units

The following codes are standardized for Archive units:

Code	Input data	Output data	Description
00 03 00 01	int minimumFMR	00 – OK 01 – Not accepted by unit 02 – Not available in unit 03 – Other error	Change the comparison threshold to determine the minimum FMR to allow a match. The unit may have its own threshold and also rules to accept new thresholds (e.g. not accepting thresholds below a determined minimum value).

9.1.5 Control codes for Capture Units

The following codes are standardized for Archive units:

Code	Input data	Output data	Description
00 04 00 01	Message (string)	00 – OK 01 – Not accepted by unit 02 – Not available in unit 03 – Other error	Message to be shown by the acquisition unit to request a trait to be presented
00 04 00 02	Message (string)	00 – OK 01 – Not accepted by unit 02 – Not available in unit 03 – Other error	Message to be shown by the acquisition unit to ask the user to wait while presenting the trait
00 04 00 03	Message (string)	00 – OK 01 – Not accepted by unit 02 – Not available in unit 03 – Other error	Message to be shown by the acquisition unit to ask the user to remove the trait
00 04 00 04	Message (string)	00 – OK 01 – Not accepted by unit 02 – Not available in unit 03 – Other error	Message to be shown by the acquisition unit to ask the user to retry the presentation of the trait
00 04 01 (01 – 04)	Byte string with the acoustic information of the message	00 – OK 01 – Not accepted by unit 02 – Not available in unit 03 – Other error	Same as codes 00 04 00 (01 – 04) but instead of a text message, an acoustic signal.

Code	Input data	Output data	Description
00 04 00 03	Message (string)	00 – OK 01 – Not accepted by unit 02 – Not available in unit 03 – Other error	Message to be shown by the acquisition unit to ask the user to remove the trait
00 04 00 04	Message (string)	00 – OK 01 – Not accepted by unit 02 – Not available in unit 03 – Other error	Message to be shown by the acquisition unit to ask the user to retry the presentation of the trait
00 04 01 (01 – 04)	Byte string with the acoustic information of the message	00 – OK 01 – Not accepted by unit 02 – Not available in unit 03 – Other error	Same as codes 00 04 00 (01 – 04) but instead of a text message, an acoustic signal.

Add new Annex A.

Annex A
(normative)

Conformance statements

A.1 General

Conformance to this document falls into the following three classes:

— OO BioAPI conformant biometric application

— OO BioAPI conformant BioAPI Framework

— OO BioAPI conformant BSP, comprising one of the following sub-classes:

  — OO BioAPI conformant Verification BSP

  — OO BioAPI conformant Identification BSP

  — OO BioAPI conformant Capture BSP

  — OO BioAPI conformant Verification Engine

  — OO BioAPI conformant Identification Engine

Conformance requirements for biometric applications, OO BioAPI Frameworks, and for BSPs are defined in A.2, A.3, and A.4, respectively.

NOTE Conformance of BFPs is not addressed in this document.

A.2 OO BioAPI Conformant Biometric Application

To claim compliance to the OO BioAPI specification, a biometric application shall, for each OO BioAPI function call utilized, invoke that operation consistently with this document. That is, all input parameters shall be present and valid. The application shall accept all valid output parameters and return values.

The biometric application shall conform to the call dependencies identified for the functions.

A.3 OO BioAPI Conformant Framework

The OO BioAPI Framework component serves the following general purposes:

a) BSP loading.

b) BSP and BFP management.

c) Component registry maintenance and management.

d) Handling of event notifications from BSPs, and sending those event notifications to (possibly multiple) event handlers in applications that have loaded that BSP.

e) Supporting API calls related to the installation or de-installation of OO BioAPI components, with appropriate update of the component registry.

f) Supporting queries from a BSP about installed BFPs.

To claim conformance to the OO BioAPI specification, an OO BioAPI Framework shall:

a) Provide component management functions as specified in **bioapi package** definition of following parts.

b) Provide component registry services in accordance with **ComponentRegistry interface definition**.

c) Conform to the data structures as defined in **data package defined in followings parts** and the error codes as defined in **BioAPIException class** when implementing a) through c), above.

d) Handle event notifications as defined in **EventHandler (bioapi package) and Event (data package)** and interfaces as defined in **GUI interface**.

A conformant OO BioAPI Framework is required to support ALL options identified in this document, since it will provide services to applications and BSPs that may implement any of those options.

A.4 OO BioAPI Conformant BSPs

A.4.1 General

To claim conformance to the OO BioAPI specification, BSPs shall implement mandatory functions for their conformance sub-class, as defined below. BSPs claim conformance to one of the conformance sub-classes specified in A.1.

BSPs shall accept all valid input parameters and return valid outputs. Optional capabilities and returns are not required to claim conformance; but any optional functions or parameters that are implemented shall be implemented in accordance with the specification requirements. Additional parameters shall not be required.

The BSP installation process shall perform the population of all required component registry entries.

BSPs shall possess a valid and unique UUID that is associated with a specific BSP product and version.

The UUID may be self-generated (see ISO/IEC 9834-8) and should (but need not) be the same on multiple systems where the same BSP product/version is installed.

BIRs generated by the BSP shall conform to the data structures **BIR interface of data package** (they shall be BioAPI BIRs). BSPs shall only return BIR object containing a registered FormatOwner with an associated valid FormatType (see **relevant interfaces in following parts**).

BSPs shall perform error handling as defined in **BioAPIException class.**

All BSPs shall support basic Component Management **bioapi package**), Utility (**BSPSchema and FramewoirkSchema interfaces**) and Event (**EventHandler interface**) operations. Callback (**GUI interfaces**), BioAPI Unit (**Unit Interface**) and Database (**BIRDatabase interface**) operations are optional.

The following table is a summary of BSP conformance requirements by subclass of BSP. Details are provided in the following sub-clauses. A.4.6 addresses conformance with respect to optional capabilities.

Table A.1 — BSP/BFP conformance sub-classes

Function	Verification BSP/BFP	Identification BSP/BFP	Capture BSP/BFP	Verification engine	Identification engine	Framework
Component Management Functions						
org.bioapi.Framework.loadBSP	X	X	X	X	X	
org.bioapi.ComponentRegistry.refresh	X	X	X	X	X	
org.bioapi.ComponentRegistry.install	X	X	X	X	X	

Table A.1 *(continued)*

Function	Verification BSP/BFP	Identification BSP/BFP	Capture BSP/BFP	Verification engine	Identification engine	Framework
org.bioapi.ComponentRegistry.uninstall	X	X	X	X	X	
org.bioapi.BSP.getUnits(Query<UnitSchema> query)	X	X	X			
org.bioapi.BSP.getBFPs(Query<BFPSchema> query)						
Callback and Event Functions						
org.bioapi.AttackSession.enableEvents	X	X	X	X	X	
org.bioapi.AttackSession.setGuiObservers						
Biometric Functions						
org.bioapi.Sensor.capture			X			
org.bioapi.Processing.createTemplate				X	X	
org.bioapi.Processing.process(BIR capturedBIR, BIR.Format ouputFormat)				X	X	
org.bioapi.Processing.process(BIR captureBIR, BIR auxiluaryBIR, BIR.Format outputFormat)						
org.bioapi.Matching.verify				X	X	
org.bioapi.Matching.identify					X	
org.bioapi.AttachSession.enroll	X	X				
org.bioapi.AttachSession.verify	X	X				
org.bioapi.AttachSession.identify		X				
Org.bioapi.AttachSession.importBIR						
Org.bioapi.Matching.presetIdentifyPopulation						
Database Functions						
org.bioapi.Archive.openDatabase						
org.bioapi.BIRDatabase.close				X	X	
org.bioapi.Archive.createDatabase				X	X	
org.bioapi.Archive.deleteDatabase						
org.bioapi.BIRDatabase.Market.terminate				X	X	
org.bioapi.BIRDatabase.storeBIR						
org.bioapi.BIRDatabase.getSingleBIR						
Org.bioapi.BIRDatabase.getBIRs						
Org.bioapi.BIRDatabase.deleteBIR						
BioAPI Unit Functions						
org.bioapi.Unit.getIndicatorStatus						
org.bioapi.Unit.setIndicatorStatus						
org.bioapi.Unit.setPowerMode						
org.bioapi.Sensor.calibrate						
Utility Functions						
org.bioapi.AttachSession.cancel	X	X	X	X	X	
org.bioapi.AttachSession.terminate, org.bioapi.BSP.terminate, org.bioapi.Framework.terminate	X	X	X	X	X	

A.4.2 OO BioAPI Conformant Verification BSPs

A.4.2.1 General

Verification BSPs are those which are capable of performing 1:1 matching (or authentication), but not 1:N identification matching.

Few matching may be supported as a series of 1:1 calls.

A OO BioAPI conformant Verification BSP shall support the following biometric functions:

— org.bioapi.Comparison.verify;

— org.bioapi.BSP.enroll.

A.4.2.2 org.bioapi.Comparison.verify

Only the nearest, better supported MaxFMRRequested is required to be supported; however, the BSP shall return that supported value (FMRAchieved). (The BSP shall indicate in the component registry whether it implements coarse scoring.).

Acceptance and use of Subtype is optional.

Return of Payload is required only if one is associated with the input ReferenceTemplate and if the score sufficiently exceeds the FMRAchieved (the BSP shall post minimum FMR required to return payload in the component registry).

Return of AdaptedBIR is optional.

Return of raw data (AuditData) is optional.

All BSPs shall provide any necessary user interface (as a default user interface) associated with the capture portion of the Verify operation. Support for application control of a GUI is optional.

A.4.2.3 org.bioapi.BSP.enroll

The Purpose value ENROLL_FOR_VERIFICATION_ONLY shall be accepted. If a different purpose value is requested but not supported, an error condition shall be set.

Acceptance of Payload is optional (i.e. if a payload is provided, it doesn't have to be accepted if the BSP does not support payload carry).

Acceptance and use of Subtype is optional.

If a BSP supports more than one output BIR data format (as indicated in the BSP schema in the component registry) it shall accept the input OutputFormat and return the output in that format.

Use of the input ReferenceTemplate, when provided, to create the output is optional. Return of raw data (AuditData) is optional. All BSPs shall provide any necessary user interface (as a default user interface) associated with the capture portion of the **Enroll** operation. Support for application control of a GUI is optional.

A.4.3 OO BioAPI Conformant Identification BSPs

A.4.3.1 General

Identification BSPs are those which are capable of performing both 1: N identification matching as well as 1:1 matching (or verification). An OO BioAPI conformant Identification BSP shall support the following biometric functions:

— org.bioapi.Comparison.verify;

— org.bioapi.Comparison.identify.

— org.bioapi.BSP.enroll

A.4.3.2 org.bioapi.verify

Only the nearest, better supported MaxFMRRequested shall be supported; however, the BSP shall return that supported value (FMRAchieved). (The BSP shall indicate in component registry whether it implements coarse scoring.).

Acceptance and use of Subtype is optional.

Return of Payload is required only if one is associated with the input ReferenceTemplate and if the score sufficiently exceeds the FMRAchieved (the BSP shall post minimum FMR required to return payload in the component registry).

Return of AdaptedBIR is optional. Return of raw data (AuditData) is optional.

As a default, all BSPs shall provide any GUI associated with the capture portion of the **Verify** operation. However, support for application control of the GUI is optional.

A.4.3.3 org.bioapi.identify

Only the nearest, better supported MaxFMRRequested shall be supported.

Return of matching Candidates is required; however, the BSP may return values for the FMRAchieved field as next nearest step/increment. (The BSP shall indicate in component registry whether it implements coarse scoring.).

Acceptance and use of Subtype is optional. Support of binning is optional. Return of raw data (AuditData) is optional.

As a default, all BSPs shall provide any GUI associated with the capture portion of the **Identify** operation. However, support for application control of the GUI is optional.

A.4.3.4 org.bioapi.BSP.enroll

Only the Purpose values ENROLL, ENROLL_FOR_VERIFICATION_ONLY, and ENROLL_FOR_IDENTIFICATION ONLY shall be accepted. If another purpose is set but not supported, an error condition shall be set. Acceptance of Payload is optional.

Acceptance and use of Subtype is optional.

If a BSP supports more than one output BIR data format (as indicated in the BSP schema in the component registry) it shall accept the input OutputFormat and return the output in that format.

Use of the input ReferenceTemplate, when provided, to create the output is optional.

Return of raw data (AuditData) is optional.

As a default, all BSPs shall provide any GUI associated with the capture portion of the **Enroll** operation. However, support for application control of the GUI is optional.

A.4.4 OO BioAPI Conformant Capture BSPs

A.4.4.1 General

OO BioAPI Capture BSPs are BSPs which provide an interface to one or more biometric sensors and return intermediate BIRs that can then be used by other BSPs (such as an OO BioAPI Biometric Engine, see A.4.4 and A.4.5). This type of BSP does not provide the capability to process or match the data it captures. An OO BioAPI conformant Capture BSP shall support the following biometric functions:

— **org.bioapi.Sensor.capture.**

A.4.4.2 org.bioapi.Sensor.capture

Return of raw data (AuditData) is optional.

Acceptance and use of Subtype is optional.

If a BSP supports more than one output BIR data format (as indicated in the BSP schema in the component registry) it shall accept the input OutputFormat and return the CapturedBIR in that format.

As a default, all BSPs shall provide any GUI associated with the **capture** operation. However, support for application control of the GUI is optional.

A.4.5 BioAPI Conformant Verification Engines

A.4.5.1 General

OO BioAPI Conformant Verification Engines are BSPs which contain the biometric processing and 1:1 matching algorithms, but do not perform the biometric capture. These are typically used in conjunction with another BSP which handle the capture operation (either a Capture or full BSP). An OO BioAPI compliant biometric engine shall support the following biometric functions:

— **org.bioapi.Processing.createTemplate;**

— **org.bioapi.Processing.process;**

— **org.bioapi.Comparison.verify.**

A.4.5.2 org.bioapi.Processing.createTemplate

Only the Purpose value ENROLL_FOR_VERIFICATION_ONLY shall be accepted. If another purpose is set but not supported, an error condition shall be set.

If a BSP supports more than one output BIR data format (as indicated in the BSP schema in the component registry) it shall accept the input OutputFormat and return the output in that format.

Acceptance of Payload is optional.

Template update (through ReferenceTemplate input) is optional.

A.4.5.3 org.bioapi.Processing.process

Only input CapturedBIRs with the Purpose value of VERIFY shall be accepted. If another purpose value is specified, an error condition shall be set.

If a BSP supports more than one output BIR data format (as indicated in the BSP schema in the component registry) it shall accept the input OutputFormat and return the ProcessedBIR in that format.

A.4.5.4 org.bioapi.Comparison.verify

Only input BIRs (ProcessedBIR) with a Purpose value of VERIFY, and (ReferenceTemplate) with a Purpose value of either ENROLL or ENROLL_FOR_VERIFICATION_ONLY shall be accepted. If another purpose is set, an error condition shall be set.

Only the nearest, better supported MaxFMRRequested shall be supported; however, the BSP shall return that supported value (FMRAchieved). (The BSP shall indicate in component registry whether it implements coarse scoring.).

Return of Payload is required only if one is associated with the input ReferenceTemplate and if the score is better than the FMRAchieved (the BSP shall post minimum FMR required to return payload in the component registry).

Return of AdaptedBIR is optional.

A.4.6 OO BioAPI Conformant Identification Engines

A.4.6.1 General

The requirements for a OO BioAPI conformant Identification Engine are identical to those of a Verification Engine, with the exception that **org.bioapi.Processing.createTemplate** shall also accept a CapturedBIR with a Purpose value of ENROLL_FOR_IDENTIFICATION_ONLY and **org.bioapi.Processing.process(BIR capturedBIR, Bir.Format outputFormat)** shall also accept a CapturedBIR with a Purpose value of IDENTIFY as well as the addition of the following required function:

— **org.bioapi.Comparison.identify.**

A.4.6.2 org.bioapi.Matching.identify

Only the nearest, better supported MaxFMRRequested shall be supported.

Return of matching Candidates is required; however, the BSP may return values for the FMRAchieved field as next nearest step/increment. (The BSP shall indicate in component registry whether it implements coarse scoring.).

Support of binning is optional.

A.4.7 Optional capabilities

A.4.7.1 General

The following capabilities are considered optional in terms of BSP support and compliance. Note that:

— If implemented, optional capabilities shall conform to specification definitions.

— BSPs are required to post (at installation) to the component registry (via the Options element of the BSPSchema within the ComponentRegistry function, ComponentRegistry definition interface in followings parts) whether or not each option is supported.

— BSP documentation shall include a table that identifies which options are supported and which options are not supported.

A.4.7.2 Optional functions

A.4.7.2.1 Primitive functions

Support of primitive functions is optional.

A.4.7.2.1.1 org.bioapi.Sensor.capture

If this function is supported, Verification BSPs need only accept the Purpose values VERIFY or ENROLL_FOR_VERIFICATION_ONLY. If another purpose value is specified and not supported, an error condition shall be set. This function shall return CapturedBIR with a purpose of VERIFY, ENROLL_FOR_VERIFICATION_ONLY, or ENROLL accordingly.

If this function is supported, Identification BSPs shall accept all purpose values (except AUDIT), even if there is no difference in the content or format of the returned data.

Acceptance and use of Subtype is optional.

If a BSP supports more than one output BIR data format (as indicated in the BSP schema in the component registry) it shall accept the input OutputFormat and return the CapturedBIR in that format.

Return of raw data (AuditData) is optional.

As a default, BSPs shall provide any GUI associated with the **org.bioapi.Sensor.capture** operation. However, support for application control of the GUI is optional.

A.4.7.2.1.2 org.bioapi.Processing.createTemplate

If this function is supported, Verification BSPs need only accept input CapturedBIRs with the Purpose value of ENROLL_FOR_VERIFICATION_ONLY. If another purpose value is specified and not supported, an error condition shall be set.

If this function is supported, Identification BSPs shall accept input CapturedBIRs with all Enroll purpose values.

Acceptance of Payload is optional.

Template update (through ReferenceTemplate input) is optional.

If a BSP supports more than one output BIR data format (as indicated in the BSP schema in the component registry) it shall accept the input OutputFormat and return the output in that format.

A.4.7.2.1.3 org.bioapi.Processing.process(BIR capturedBIR, Bir.Format outputFormat)

If this function is supported, Verification BSPs need only accept input CapturedBIRs with the Purpose value of VERIFY. If another purpose value is specified and not supported, an error condition shall be set.

If this function is supported, Identification BSPs shall accept input CapturedBIRs with the Purpose value of either VERIFY or IDENTIFY, even if there is no difference in the content or format of the returned data.

If a BSP supports more than one output BIR data format (as indicated in the BSP schema in the component registry) it shall accept the input OutputFormat and return the ProcessedBIR in that format.

A.4.7.2.1.4 org.bioapi.Processing.process(BIR capturedBIR, BIR auxiluaryBIR, Bir.Format outputFormat)

If this function is supported, Verification BSPs need only accept input CapturedBIRs with the Purpose value of VERIFY. If another purpose value is specified and not supported, an error condition shall be set.

If this function is supported, Identification BSPs shall accept input CapturedBIRs with the Purpose value of either VERIFY or IDENTIFY, even if there is no difference in the content or format of the returned data.

BSPs supporting this function will include in their documentation the format and content (or alternatively the source) of the AuxiliaryData which it accepts as input.

If a BSP supports more than one output BIR data format (as indicated in the BSP schema in the component registry) it shall accept the input OutputFormat and return the ProcessedBIR in that format.

A.4.7.2.1.5 org.bioapi.Comparison.verify

If this function is supported, only input BIRs (ProcessedBIR) with a Purpose value of VERIFY and ReferenceTemplates with a purpose value of either ENROLL or ENROLL_FOR_VERIFICATION_ONLY shall be accepted. If another purpose is set and not supported, an error condition shall be set.

Only the nearest, better supported MaxFMRRequested shall be supported; however, the BSP shall return that supported value (FMRAchieved). (The BSP shall indicate in component registry whether it implements coarse scoring.).

Return of Payload is required only if one is associated with the input ReferenceTemplate and if the score is better than the FMRAchieved (the BSP shall post minimum FMR required to return payload in the component registry).

Return of AdaptedBIR is optional.

A.4.7.2.1.6 org.bioapi.Comparison.identify

If this function is supported, only input BIRs (ProcessedBIR) with a Purpose value of IDENTIFY shall be accepted. If another purpose value is specified and not supported, an error condition shall be set.

Only the nearest, better supported MaxFMRRequested shall be supported. Return of matching Candidates is required; however, the BSP may return values for the FMRAchieved field as next nearest step/increment. (The BSP shall indicate in component registry whether it implements coarse scoring.).

Support of binning is optional.

A.4.7.2.2 Database operations

BSPs are not required to provide an internal (or BSP-controlled) BIR database. If one is provided, however, ALL database functions shall be provided in order to access and maintain it. All provided database functions shall conform to the definitions.

A.4.7.2.3 org.bioapi.importBIR

This function, as a whole, is optional. If it is provided, the component registry shall so reflect, and it shall be implemented as defined.

Verification BSPs are only required to process imported data if the purpose value is ENROLL or ENROLL_FOR_VERIFICATION_ONLY. Identification BSPs are only required to process imported data if the purpose value with any Enroll purpose value.

If a BSP supports more than one output BIR data format (as indicated in the BSP schema in the component registry) it shall accept the input OutputFormat and return the output in that format.

A.4.7.2.4 org.bioapi.Comparison.presetIdentifyPopulation

This function, as a whole, is optional. If it is provided, the component registry shall so reflect, and it shall be implemented as defined.

A.4.7.2.5 OO BioAPI Unit operations

All OO BioAPI Unit operations, as a whole, are optional. If any are provided, the component registry shall so reflect, and each function shall be implemented as defined.

A.4.7.3 Optional subfunctions

A.4.7.3.1 General

The following table identifies optional capabilities, each of which the BSP shall declare as being supported or not supported.

Table 1 — Optional subfunctions [TBR]

Capability	Supported	Not supported
Return of raw/audit data		
Return of quality		
Application-controlled GUI		
GUI streaming callbacks		
Detection of source presence		
Payload		