Sanchez-Reillo R., Tilton C.J., Sanduijav E. (2015) Object-Oriented BioAPI Standard. In: Li S.Z., Jain A.K. (eds) *Encyclopedia of Biometrics*. Springer, Boston, MA

DOI: https://doi.org/10.1007/978-1-4899-7488-4_9197

Metadata of the chapter that will be visualized online

Chapter Title	Object-Oriented BioAPI Standard	
Copyright Year	2014	
Copyright Holder	Springer Science+Business Media New York	
Corresponding Author	Family Name	**Sanchez-Reillo**
	Particle	
	Given Name	**Raul**
	Suffix	
	Division	GUTI (University Group for Identification Technologies)
	Organization	University Carlos III of Madrid
	Address	Avda. Universidad, 30, 28911, Leganes, Madrid, Spain
	Email	raul.sanchezreillo@gmail.com
Author	Family Name	**Tilton**
	Particle	
	Given Name	**Catherine J.**
	Suffix	
	Division	VP, Standards & Technology
	Organization	Daon
	Address	11955 Freedom Drive Suite 16000, 20190, Reston, VA, USA
	Email	cathy.tilton@daon.com
Author	Family Name	**Sanduijav**
	Particle	
	Given Name	**Enkhbayar**
	Suffix	
	Organization	Hitachi Solutions, Ltd
	Address	40 Basinghall Ave, London, UK
	Email	enkhbayar.sanduijav.jj@hitachi-solutions.com

Encyclopedia of Biometrics
DOI 10.1007/978-3-642-27733-7_9197-2

Object-Oriented BioAPI Standard

Raul Sanchez-Reillo*[a], Catherine J. Tilton[b] and Enkhbayar Sanduijav[c]
[a]GUTI (University Group for Identification Technologies), University Carlos III of Madrid, Leganes, Madrid, Spain
[b]VP, Standards & Technology, Daon, Reston, VA, USA
[c]Hitachi Solutions, Ltd, London, UK

Q1

Synonyms

BioAPI Java; BioAPI C#; ISO/IEC 30106; OO BioAPI

Definition

Application Programming Interface (API) for programming biometric applications and Biometric Service Providers (BSP). It is based on BioAPI (i.e., ISO/IEC 19784-1), but it takes advantage of object-oriented programming. It covers a general architecture, plus its specification in different object-oriented programming languages, such as Java or C#.

Introduction

When developing biometric applications, particularly when integrating modules from third parties, a standardized API is needed. Such standardized API allows interoperability among vendors, speeding up the development of final applications while, at the same time, contributing to increase competitiveness among companies. It also helps in reducing cost for those companies offering Biometric Service Providers (BSP), since using a common API removes the need of continuously adapting the BSP to each final application.

As an international standard BioAPI was born at the end of the twentieth century. In 2006 the International Standard ISO/IEC 19784-1, Information technology – Biometric application programming interface – Part 1: BioAPI specification [1] was specified and developed in ANSI C language. Since 2006, it has been continuously evolving. From this evolution, it is important to highlight the addition of support for handling a graphical user interface (GUI) [2], the allowance of deploying the system using a framework-free approach [3], the support of security mechanisms [4], and the extension of the functionalities of the units that compose Biometric Service Providers (BSPs), with a comprehensive definition of Biometric Function Providers (BFP) [5–7].

But nowadays there is a need to develop the specification using object-oriented approaches, particularly in the case of the applications, although the development of BSPs with object-oriented languages is also required. From the initial specification given in ISO/IEC 19784-1, a new API is being defined in the ISO/IEC 30106 family of standards, which translates the ANSI C approach of 19784-1 to object-oriented programming languages, such as Java [8] and C# [9]. As the standard shall be opened to other object-oriented programming languages, part 1 of ISO/IEC 30106 provides the specification of a generic architecture for object-oriented BioAPI (OO BioAPI) [9].

*E-mail: rsreillo@ing.uc3m.es, raul.sanchezreillo@gmail.com

Encyclopedia of Biometrics
DOI 10.1007/978-3-642-27733-7_9197-2

OO BioAPI Description

In the specification of OO BioAPI, the first rule is not to lose any of the functionalities from ISO/IEC 19784-1, but easing the specification and the development. Within this goal, most of the functionality is kept intact, respecting the same process flow as in BioAPI. In a few words, this means that an application should follow these steps:

1. Initialize the framework (in case this is not done by the operating system during booting).
2. Request that the framework for the BSPs and BFPs be installed in the system.
3. Select one of the BSPs and load it. During the process of the loading, the BSP initializes itself and may ask the component registry for the BFPs installed, as to look for compatible ones and allocate in its list of available units those of the supported BFPs.
4. The application may ask the BSPs for its supported units, as to be able to choose from them in the next step.
5. Attach a session of that BSP, either indicating the units to be used or not saying anything and leaving the decision to the BSP.
6. Proceed with the calling of all those biometric methods that the application may need.
7. Whenever the application no longer needs the BSP, it will detach the session and unload the BSP.
8. Before exiting the application, terminate the entire BioAPI functionality.

In order to implement all this functionality, the hierarchical structure shown in Fig. 1 is defined. Going bottom-up, OO BioAPI defines interfaces for each of the 4 BioAPI_Units categories (i.e., Archive, Comparison, Processing, and Sensor). This level of interaction is only defined for object-oriented programming reasons, not corresponding to any of the different interface layers defined in ISO/IEC 19784-1. In other words, a developer or programmer shall never distribute unit

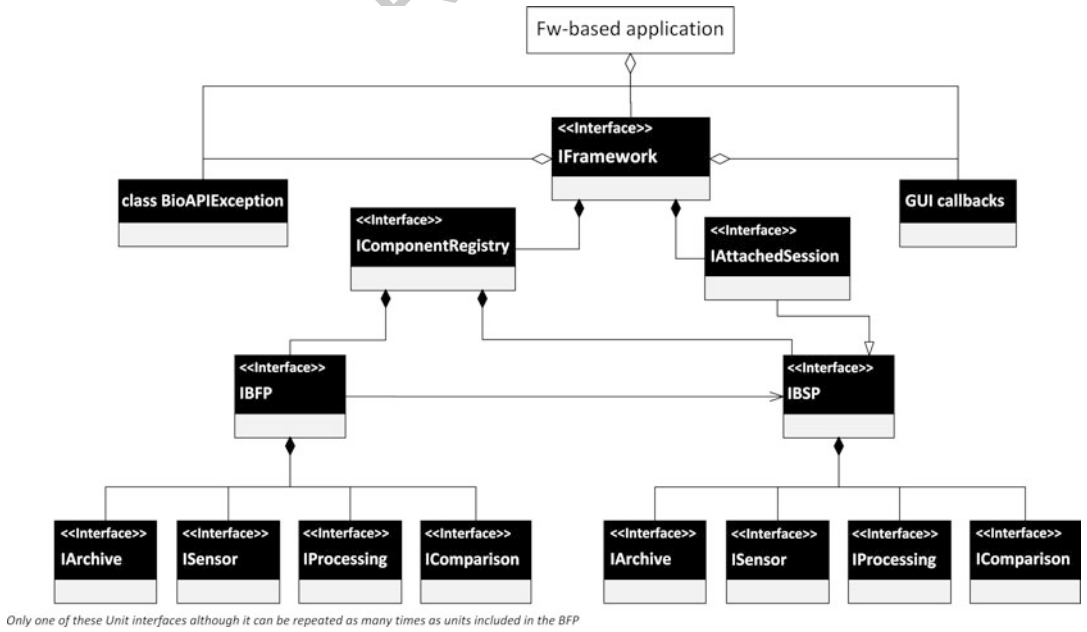

Fig. 1 Hierarchical model of OO BioAPI

Encyclopedia of Biometrics
DOI 10.1007/978-3-642-27733-7_9197-2

classes, but only either BSPs or BFPs. Each of these units provides atomic functionality, depending only on its own BioAPI_Unit, but not interacting with other units.

A BFP is defined as a collection of BioAPI_Units, all of them from the same category. Therefore, a BFP can be understood as library of unit objects that can be later on accessed by a BSP, whenever either the BSP itself or the application selects it for being used. Therefore, the BFP inherits all the functionality of each of the units, providing it to the BSP. The BFP interface (IBFP) also adds the procedures for registering itself into the Component Registry and allowing the connection of a BSP as to allow its access to one of the BioAPI_Units of the BFP. The communication between IBFP and BSP interfaces (IBSP) is the equivalent to the FPI interface in ISO/IEC 19784-1, which is called SFPI (for sensor BFPs), MFPI (for comparison BFPs), and PFPI (for processing BFPs) in parts 4 to 6 of ISO/IEC 19784.

A BSP may contain as many BioAPI_Units as desired, from any combination of the four categories. Each of these units is imported into the BSP, and it is up to the BSP to allow the external world to access the atomic functionalities of the BioAPI_Units or restrict the external access to the BSPs aggregated methods (i.e., those methods that combine calls to different atomic methods from any of the active BioAPI_Units, being the active units those that have been selected during the session attachment). In addition to importing the BioAPI_Units and including aggregated functionality, the BSP also includes methods to interact with the Component Registry and with the Framework. Therefore, IBSP is the interface of the BSP level, which corresponds to the SPI interface defined in ISO/IEC 19784.

With the BFP level defined, the mission of the Framework is to provide the link between the application and those BSPs installed. In order to do that, the Framework contains a list of attached sessions each of them inherits from a loaded BSP. Therefore, the functionality that the BSP developer has decided to export is provided to the application. In addition to this, the Framework level also provides interaction with the Component Registry, including the installation and uninstallation of BSPs and BFPs. Finally, it also allows the forwarding of callback functions in order to allow the BioAPI_Units to interact with the GUI from the application. All this functionality is what is defined in ISO/IEC 19784-1 as the API interface.

It is important to highlight the importance of the IAttachedSession interface. Whenever the application desires to use the functionality of a BSP, it shall attach a session, by using the method IFramework.BSPAttach(...), even indicating the units to be used during that attached session (only one unit per category for such session). Then the Framework creates an AttachedSession object which inherits all the properties and methods exported by the BSP selected. By accessing that object, the application can use the whole BSP functionality, until the application does not longer require using that BSP, and therefore the method IFramework.BSPDetach(...) is called and the AttachedSession object is destroyed.

Last, but not least, it is important to note two requirements in this specification. First, in OO BioAPI, error handling is done by the use of exceptions. Therefore, the class BioAPIException has been defined to provide that support all throughout OO BioAPI components and modules. Second, biometric data exchange is done by the use of CBEFF, with the full support either for Simple BIRs or for complex BIRs.

In the case of a framework-free implementation of OO BioAPI, the application talks directly with IBSP. In order to allow the interaction between BSPs and BFPs, the application shall implement a static Component Registry and the corresponding callback function that will allow the BSP to dynamically know the BFPs available in that particular application. The specification

Encyclopedia of Biometrics
DOI 10.1007/978-3-642-27733-7_9197-2

expects that the developer of BFPs and/or BSPs develops them without considering if they are going to be used with a Framework or in a framework-free environment.

Reference Implementations

In addition to the whole specification of OO BioAPI, and with the aim of helping its adoption and easing the understanding to developers, open-source reference implementations and examples are being provided for each of the supported languages. The reference implementation for Java was originated by the R&D group of Prof. Steve Elliott at Purdue University and is available in http://sourceforge.net/projects/bioapijava/. The C# reference implementation was started by Carlos III University of Madrid and is available in https://joinup.ec.europa.eu/software/bioapicsharp/home.

Summary

OO BioAPI is the specification of ISO/IEC 19784-1 (also known as BioAPI) for object-oriented programming languages. It is standardized in the ISO/IEC 30106 series of standards and includes a language-independent specification of its architecture in part 1, while the rest of the parts details the specification in different object-oriented programming languages, such as Java (in part 2) and C# (in part 3). For the language-specific definitions, also some open-source reference implementations have been defined.

Related Entries

- BioAPI, Standardization
- Biometric Technical Interface, Standardization
- CBEFF

References

1. ISO/IEC JTC1/SC37, ISO/IEC 19784-1:2006 information technology – biometric application programming interface – part 1: BioAPI specification (2006)
2. ISO/IEC JTC1/SC37, ISO/IEC 19784-1:2006/Amd 1:2007 BioGUI specification (2007)
3. ISO/IEC JTC1/SC37, ISO/IEC 19784-1:2006/Amd 2:2009 framework-free BioAPI (2009)
4. ISO/IEC JTC1/SC37, ISO/IEC 19784-1:2006/Amd 3:2010 support for interchange of certificates and security assertions, and other security aspects (2010)
5. ISO/IEC JTC1/SC37, ISO/IEC 19784-4:2011 information technology – biometric application programming interface – part 4: biometric sensor function provider interface (2011)
6. ISO/IEC JTC1/SC37, ISO/IEC CD 19784-5 information technology – biometric application programming interface – part 5: biometric processing algorithm function provider interface (under development)

Encyclopedia of Biometrics
DOI 10.1007/978-3-642-27733-7_9197-2

7. ISO/IEC JTC1/SC37, ISO/IEC CD 19784-6 information technology – biometric application programming interface – part 6: biometric matching algorithm function provider interface (under development)
8. ISO/IEC JTC1/SC37, ISO/IEC CD 30106-1 information technology – BioAPI for object oriented programming languages – part 1: architecture (under development)
9. ISO/IEC JTC1/SC37, ISO/IEC CD 30106-2 information technology – BioAPI for object oriented programming languages – part 2: Java implementation (under development)
10. ISO/IEC JTC1/SC37, ISO/IEC CD 30106-3 information technology – BioAPI for object oriented programming languages – part 3: C# implementation (under development)

Encyclopedia of Biometrics
DOI 10.1007/978-3-642-27733-7_9197-2

Author Queries

Query Refs.	Details Required
Q1	Please check if edit to entry title is okay.