Sánchez-Reíllo, R., Niesing, M. (2015) BioAPI, Standardization. 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_9202

Metadata of the chapter that will be visualized online

Chapter Title	BioAPI, Standardization	
Copyright Year	2014	
Copyright Holder	Springer-Verlag US	
Author	Family Name	**Sanchez-Reillo**
	Particle	
	Given Name	**Raul**
	Suffix	
	Email	rsreillo@ing.uc3m.es
	Email	raul.sanchezreillo@gmail.com
Author	Family Name	**Niesing**
	Particle	
	Given Name	**Matthias**
	Suffix	

Encyclopedia of Biometrics
DOI 10.1007/978-3-642-27733-7_9202-2

BioAPI, Standardization

Raul Sanchez-Reillo[a]* and Matthias Niesing[b]
[a]GUTI (University Group for Identification Technologies), Carlos III University of Madrid, Leganes, Madrid, Spain
[b]Secunet Security Networks AG, Essen, Germany

Synonyms

ISO/IEC 19784; ISO/IEC 19784-1

Definition

BioAPI is the abbreviation for Biometric Application Programming Interface (API). It is an API to develop biometric-related applications and was created by the BioAPI Consortium. In 2005, BioAPI was published as an international standard under ISO/IEC 19784-1:2005 [1]. From that publication it has evolved by adding new functionalities, being now under a revision to create the new BioAPI 3.0.

Origins and Basic Specification

The use of standardized Application Programming Interface (API) is demanded for allowing interoperability among developers and removing the cost for adapting the biometric solution to the evaluation protocol. In Biometrics, the reference API for developing biometric-related applications is BioAPI. BioAPI was originated by the BioAPI Consortium [2]. The BioAPI Consortium first announced its formation and intent to develop a biometric API standard in April of 1998. By the end of the year, this group had developed a multi-level API architecture and begun defining the associated components.

In March of 1999, the Information Technology Laboratory of the National Institute of Standards and Technology (NIST) and the US Biometric Consortium sponsored a unification meeting in which the Human Authentication API (HA-API) working group (which had published a high level biometric API in 1997) agreed to merge their activities with the BioAPI Consortium. As part of this agreement, the BioAPI Consortium agreed to restructure their organization.

The reconstituted BioAPI Consortium completed its efforts to define the biometric API architecture and to solidify its organizational structure and operations by mid-1999. Version 1.0 of the Specification was released in March, 2000, and the Reference Implementation was released in September 2000. Version 1.1 of both the Specification and Reference Implementation were released in March 2001.

In February of 2002, BioAPI Version 1.1 was approved as an American National Standard through INCITS (ANSI/INCITS 358-2002). When ISO/IEC JTC1/SC37 was constituted, the US delegation offered that standard for adoption as an international standard through ISO/IEC,

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

Encyclopedia of Biometrics
DOI 10.1007/978-3-642-27733-7_9202-2

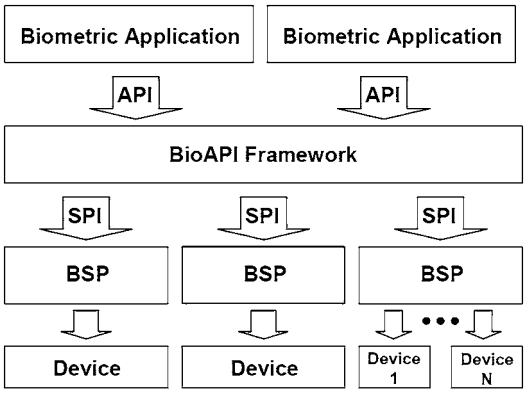

Fig. 1 BioAPI's API/SPI model [1]

although it was decided that instead of adopting the standard directly, a new project would be started to improve the specification of BioAPI. This project led to the publication, in 2005, of the ISO/IEC 19784-1 standard [1], which is also known as BioAPI 2.0.

The basic idea of BioAPI is given in Fig. 1. It is based on a framework that serves as an interface between the application and those biometric units available. A biometric unit can be understood as any element that can provide biometric-related services, such as capture devices, algorithms, or storage units. To communicate with any external application, the framework offers an API (set of functions) that works independently of the devices being developed. On the other hand, to communicate with the units, the framework offers another API (here called SPI), with all of the functions needed to access such units. Independent of the type of unit, they provide a component with the supported services to allow them to be accessed by the framework. Such a driver is called a Biometric Service Provider (BSP).

BSPs can have nearly any form. They can be related to just one physical device, more than one, or none (e.g., being an algorithm). Due to the functions provided by the SPI, BSPs can be dynamically loaded and unloaded from the framework. Any external application can use any of the BSPs loaded through the framework to allow any system complexity. Moreover, BSPs can communicate with the framework to access other loaded BSPs.

Having given this short introduction, a general idea on how BioAPI works could be perceived, but with the further improvements that came in the years following its publication, its implementation is far from being understood. Particularly it is difficult to understand not only how this idea can be implemented but also how could this be implemented without the need of a framework (i.e., it is called framework-free BioAPI). This entry presents an overview that tries to clarify the whole architecture, how this interacts with a Graphical User Interface (GUI), and the steps taken to implement a framework-free and a full-framework version of BioAPI. Future works in this area are outline.

BioAPI Architecture

BioAPI is defined in a way that allows both structured development of applications and services, as well as interoperability between application and Biometric Service Providers (BSP) and among BSPs. In a few words, this API has to allow the development of applications in a system based

Encyclopedia of Biometrics
DOI 10.1007/978-3-642-27733-7_9202-2

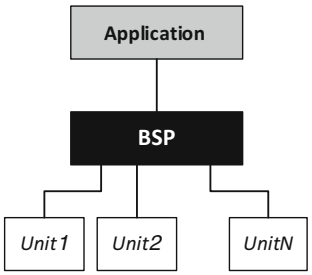

Fig. 2 Framework-free application with a single BSP

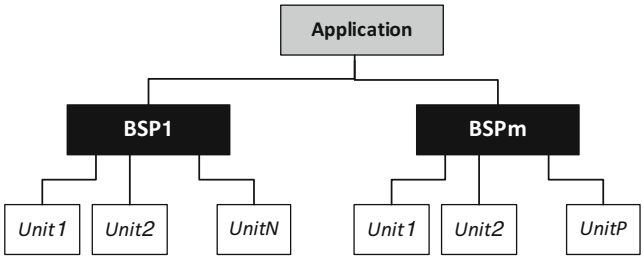

Fig. 3 Framework-free application with multiple BSPs

on a BioAPI Framework, as well as applications in a system where no framework is used (framework-free). Starting with the later, an application in a framework-free based system is developed using a BioAPI that allows the instantiation of a BSP, which is based on the instantiation of one or several BioAPI_Units. The BSP can host more than one BioAPI_Unit of each category, but when a session is attached, only up to a maximum of 1 BioAPI_Unit per category can be selected and, therefore, used. Q1

It is not allowed that the application access BioAPI_Units directly. Therefore this standard does not define a BioAPI_Unit interface, but only a hierarchical interface/class model that may ease the implementation of the BSP. The BSP shall inherit all public methods and data structures of the BioAPI_Units, and it is up to the implementation of the BSP developer to decide which of them are offered to the external world and which of them are only returning the lack of support of such method. This is represented in Fig. 2.

But a framework-free application may also be able to use several BSPs. Figure 3 represents how this can be implemented. As it can be seen, each BSP is used as a library to be used by the application. In such a case, if, for example, an application would like to use two different sensors and one processing and comparison BioAPI_Unit, then three BSPs can be used, one for each sensor and another for the Comparison and Processing BioAPI_Units.

Although the previous solution is fully valid, it may raise certain concerns in practical situations. The first concern is dealing with industry providing elements to be used in applications. It may happen that a provider is only providing sensors and would like to include support to all of its family of sensors as a single entity (e.g., a library). That provider may consider implementing that as a single BSP, but it may not be interested in providing monolithic methods (i.e., aggregated functionality such as Enrol, which performs in one single call the capture, the processing of the sample, the creation of the biometric reference, and even the archiving).

Therefore it will need to request these kinds of functions to be done at the application level, exchanging biometric information between the BSP and the application. Then, the application may forward that biometric data to another BSP to complete the action. So a possible solution for

Encyclopedia of Biometrics
DOI 10.1007/978-3-642-27733-7_9202-2

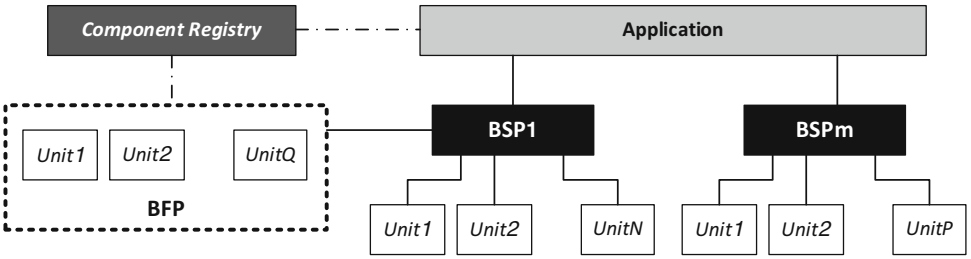

Fig. 4 Example of a framework-free application using BFPs

this inconvenience is to allow those sensors to directly interact with other BSPs, instead of using the application to do that. To achieve this, the biometric product provider may create an entity (e.g., a library) containing several BioAPI_Units of the same kind. This is called a Biometric Function Provider (BFP), which has mainly the following characteristics:

- The BFP shall only host BioAPI_Units of the same category.
- The BFP allows a BSP to be linked to one of its BioAPI_Units, in order to complete or adapt the functionality of the BSP.
- The BFP shall not provide functionality to the application, but only the link to the BSP. It is the BSP that provides functionality to the application.

The abovementioned situation also solves a problem from a developers' point of view, which deals with simplicity in developing applications. If an application requires the use of BioAPI_Units from different providers (e.g., a sensor from one provider and processing, comparison, and archive BioAPI_Units from other provider), then it will have to load two different BSPs, calling each of the methods independently. As mentioned earlier, this has the inconvenience that it won't be possible to call a monolithic method, such as Verify(), which performs the data capture, the processing, extraction of the biometric reference from the database, the comparison, and taking the decision, all within the same single call. Therefore the application programmer will have to know which individual methods have to be called from each of the BSPs, in order to get the same functionality. By using a BSP that supports monolithic methods, and requesting the BSP to be linked to those BioAPI_Units for the BFPs of other product providers, once that link is established, the application can call those monolithic methods without taking into account that the functionality is provided by different vendors.

Last, but not least, there is a concern about security in certain operations. As biometric data is sensitive personal data, some clients may require that the biometric application won't directly access the user biometric data (i.e., the BIRs), avoiding the possibility of malware to tamper with such data. By using BFPs, all sensible data will be handled at the BSP level, and no Biometric Information Record (BIR) may be accessible to the application, not only simplifying the application development (by the use of monolithic methods) but also the security level achieved.

Figure 4 represents how a framework-free application is structured using BFPs.

The BFP is not accessed directly by the application. BioAPI calls are created to allow the application to know the BioAPI_Units that are contained in the BFPs so that the application may later request one BSP to attach one of those BioAPI_Units of the BFP.

All the above ideas are implemented in systems where the components to be used by the application (BSPs and BFPs) are known a priori, and only used by a single application (i.e., a static

Encyclopedia of Biometrics
DOI 10.1007/978-3-642-27733-7_9202-2

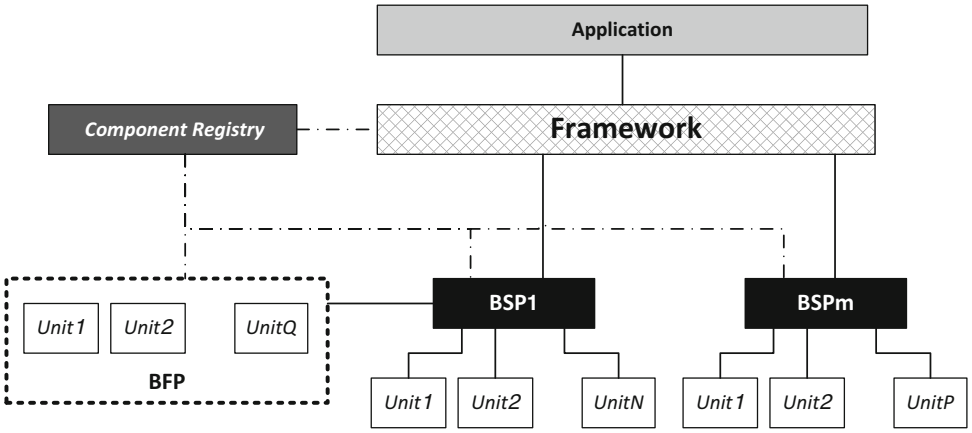

Fig. 5 Generic structure of a framework-based application

implementation of a biometric application, with the components chosen at time of compilation). But the same approach can be followed to allow the implementation of mechanisms for the dynamic selection of components to be used by the application, but with no a priori knowledge from the application developer. This is achieved by the inclusion of a common framework, which can be installed in the system where the application is expected to be running. In such a case, the application would request the framework the list of the BSPs and BFPs installed and select the BSPs (with the requested attachment of BioAPI_Units from BFPs) to be instantiated dynamically by the framework. Then their methods and data structures would be accessed through the framework. The application would never be allowed to access the BSPs directly. This is depicted in Fig. 5.

In a framework-based system, BSPs may be accessed by several applications at the same time, and it may also happen that BioAPI_Units in the BFPs are also accessed by several BSPs at the same time. It is up to the implementation of the framework-based system as to how this is implemented (e.g., this may be done by queuing requests from the different sources or by responding to occurring events).

BioAPI Evolution and Adoption

From the basic specification given in ISO/IEC 19784-1:2005, additional specifications have been developed. This evolution includes different parts of the ISO/IEC 19784 family of standards as specified in Biometric Technical Interface, Standardization. It includes further specifications on how to use the application Graphical User Interface (GUI), as well as providing further definition of the BFPs for each of the four categories of BioAPI_Units.

From the publication of version 1.1, BioAPI has been adopted by several companies for a variety of products. This can be seen in http://www.bioapi.org/products.asp. Since the publication of ISO/IEC 19784-1:2005, the adoption of BioAPI 2.0 has increased progressively, either by the development of brand-new BSPs and/or BFPs by major industrial players or by the adaptation of previous BioAPI 1.1 developments to a BioAPI 2.0 compliant version. The development of BioAPI versions in object oriented languages [3] is expected to help in the global adoption of this technology.

Encyclopedia of Biometrics
DOI 10.1007/978-3-642-27733-7_9202-2

Summary

BioAPI is a comprehensive definition of an API for biometric-related applications, which can be adopted by any kind of application and under any kind of platform. Although defined in ANSI C language, there are also specifications in object-oriented languages, such as the specified family of standards ISO/IEC 30106 [3]. The basic specification is currently being revised as to build up a new 3.0 version that integrates all the evolutions that BioAPI has accomplished in the past years.

Related Entries

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

References

1. ISO/IEC JTC1/SC37, ISO/IEC 19784-1, Information technology – biometric application programming interface – Part 1: BioAPI specification, 2005. Available at http://www.iso.org/iso/home/store. There is a revision in process (more information in http://www.iso.org/iso/home/search.htm?qt=19784-1&sort=rel&type=simple&published=on)
2. BioAPI Consortium webpage: http://www.bioapi.org
3. ISO/IEC JTC1/SC37, ISO/IEC 30106, Information technology – object oriented BioAPI. (Under development, more information in http://www.iso.org/iso/home/search.htm?qt=30106&published=on&active_tab=standards&sort_by=rel)

Encyclopedia of Biometrics
DOI 10.1007/978-3-642-27733-7_9202-2

Author Queries

Query Refs.	Details Required
Q1	Please check if edit to sentence starting “Starting with the…” is okay.
Q2	Title “CBEFF” is not provided in TOC. Please check.
Q3	Title “Object Oriented BioAPI, Standardization” is mismatched with TOC. Please check.