An API is a defined way in which a software application talks to another, usually lower level, software package that performs a set of functions used by that application. The application requests services from and receives responses from a technology or service module. This is typical of any Software Developer Kit (SDK) provided by a technology manufacturer. SDK APIs are typically proprietary, however, and unique to that particular vendor.

A standard API defines a common way for an application to communicate to a related type of technology. By using a common language, or common set of function calls, parameters, and data structures, the application can communicate with various conforming implementations of the technology. An example of such a common API is the Crypto API, in which an application can use various Cryptologic Service Provider (CSP) modules with no change in the application code dependent on which encryption module(s) is “plugged in”.

A Biometric API standard defines a generic way of interfacing with a broad range of biometric technologies as well as defining a common method of interfacing with a particular biometric technology.

The BioAPI defines such a generic biometric application program interface for biometrics. That is, it allows any software application to integrate (communicate with) various different biometric technologies in a standard way. It also standardizes the format of biometric data records. This, in turn, allows for interchangeability and interoperability of biometric systems.

The purpose of any standard is to allow for interchangeability and/or interoperability, thus reducing risk to the integrator and end user. The creation of a standard generic biometric API allows:

- Easy substitution of biometric technologies
- Simple integration of multiple biometrics using the same interface
- Rapid extension of biometric technology across multiple applications

This avoids user “lock-in”, where it is difficult to change to a different biometric technology, thus “locking” the user into a particular biometric type and vendor.

Background

To better understand the evolution of the BioAPI, it might be worthwhile to review a little history. When biometrics were first used within automated computer systems, beginning in law enforcement and other government applications, most systems required custom tailoring. The integration of biometric technology was accomplished by inclusion of application source code, linking in of biometric routines, or communicating with single purpose computers dedicated to performing biometric search and/or match functions.

Eventually, when other uses of biometrics were beginning to be discovered, vendors began offering their own proprietary software development kits (SDKs) with their own unique APIs. This allowed integrators to more easily work with the biometric capture, processing, and matching algorithms and to better isolate that portion of their code.

The first work to standardize an API for a single biometric type began within the speaker verification
industry. In the 1996 timeframe, the Speaker Verification API (SVAPI) effort began, and by 1997 a usable specification and runtime code was available. This had the support of many of the vendors of this technology. The SVAPI specification is a relatively low-level interface specification that is specific to voice verification.

In late 1997, the biometrics research and engineering group at the National Security Agency (NSA) in Ft. Meade, Maryland, who had actively participated in the SVAPI work, expressed interest in sponsoring the development of a higher level generic biometric API which would allow for the interface to and interchangeability of a broad range of biometric technologies. This interest was primarily for the purpose of user authentication to improve information security within the government. By early 1998, NSA had contracted with The National Registry, Inc. (now SAFLINK) to jointly develop such an API and implement a proof-of-concept implementation. This work was completed in December of 1998 with the publication of the Human Authentication API (HA-API) specification, Version 1.03, and the demonstration of its use with fingerprint imaging, voice, and facial recognition technologies. In January of 1998, the HA-API Steering Group was formed, chaired by the U.S. Biometric Consortium, and the HA-API specification was publicly endorsed by 11 biometric companies. About the same time as HA-API was in progress, both IBM and I/O Software announced work on biometric APIs known as Advanced Identification System (AIS) and Biometric API (BAPI) respectively. Then, in April of 1998, an announcement was made by Compaq of the formation of the ‘BioAPI Consortium’, an alliance of six companies, including Microsoft, to create a multi-level industry standard biometric API. In August of 1998, Intel announced work on an extension to the Open Group Common Data Security Architecture (CDSA) standard called User Authentication Services (UAS) that incorporated biometrics and other technologies for user authentication and which was based on HA-API.

At this point, an industry with no API standards just a few months before now had too many to choose from. Later that year, both AIS and BAPI agreed to fold their efforts into the BioAPI effort. In February of 1999, the National Institute of Standards and Technology (NIST) hosted a unification meeting between the HA-API Working Group and the BioAPI Consortium, along with representatives from the CDSA/UAS Working Group. The merger was consummated in March of that year, with UAS agreeing to backfit to the BioAPI when available.

The BioAPI Consortium is a group of over 60 companies and organizations that have a common interest in promoting the growth of the biometrics market. It is a collaborative effort of the biometrics industry, government, and major information technology vendors. The members comprising the BioAPI Consortium cover a broad cross section of biometric technology manufacturers, integrators, OEMs, and end users. This organization is international in scope, with over one-third of its membership being non-U.S. based. Operations of the Consortium are managed by a seven-member steering committee elected by the Membership. Currently serving since October 1999 are SAFLINK Corporation (Chair), Unisys (Secretary), Iridian (Treasurer/Conformance Test Chair), Intel (Technical Editor), Mytec Technologies (Reference Implementation Chair), NIST (External Liaisons), and Compaq.

There are 5 working groups responsible for the work of the organization:

- **The Applications Working Group (AWG)** is chaired by John Wilson of Intel and is responsible for defining the top-level application interface of the BioAPI.
- **The Device Working Group (DWG)** is chaired by Aaron Watson of I/O Software, Inc. and is responsible for defining the lower, device level interface of the BioAPI.

---

**Who are the BioAPI Consortium?**

The BioAPI Consortium is a group of over 60 companies and organizations that have a common interest in promoting the growth of the biometrics market. It is a collaborative effort of the biometrics industry, government, and major information technology vendors. The members comprising the BioAPI Consortium cover a broad cross section of biometric technology manufacturers, integrators, OEMs, and end users. This organization is international in scope, with over one-third of its membership being non-U.S. based.

Operations of the Consortium are managed by a seven-member steering committee elected by the Membership. Currently serving since October 1999 are SAFLINK Corporation (Chair), Unisys (Secretary), Iridian (Treasurer/Conformance Test Chair), Intel (Technical Editor), Mytec Technologies (Reference Implementation Chair), NIST (External Liaisons), and Compaq.

There are 5 working groups responsible for the work of the organization:

- **The Applications Working Group (AWG)** is chaired by John Wilson of Intel and is responsible for defining the top-level application interface of the BioAPI.
- **The Device Working Group (DWG)** is chaired by Aaron Watson of I/O Software, Inc. and is responsible for defining the lower, device level interface of the BioAPI.
The External Working Group (XWG) is chaired by Fernando Podio of NIST and is responsible for transitioning the resulting BioAPI specification to a recognized standards body and for liaison with other related efforts.

The Reference Implementation Working Group (RWG) is chaired by Colin Soutar of Mytec and is responsible for implementing the reference implementation code in conformance with the BioAPI Specification.

The Conformance Test Working Group (RWG) is chaired by Jim Cambier of Iridian Technologies and is responsible for implementing a conformance test suite and any associated testing or branding programs.

Membership is open to all interested organizations, and new members are invited to join BioAPI and participate in its promotion of biometric standards. The Consortium has a single membership classification that applies to all current and future Members.

The BioAPI Consortium was founded to develop a multi-level biometric Application Programming Interface (API) that brings platform and device independence to application programmers and biometric service providers. BioAPI has developed a specification and reference implementation for a standardized API that will be compatible with a wide range of biometric applications programs and a broad spectrum of biometrics technologies.

A technical overview of the BioAPI specification

The BioAPI Specification is the culmination of several years of effort by a large cross-section of biometric vendors, integrators, and end-users, both in the United States and internationally. It represents an industry consensus that was not easily forged, and incorporates the best features from across the range of inputs to provide a standard that can accommodate a wide variety of biometric technologies (fingerprint, face, voice, iris, hand-geometry, signature, etc.) across a broad scale of applications (embedded systems to network security to national ID systems).

Most importantly, however, is that the BioAPI is an open system standard. This means that it has been designed for use in any environment. It is platform (operating system) independent. Therefore, it can be used in a heterogeneous environment and support cross-platform implementations.

Architecture

The BioAPI architecture is an ‘API/SPI’ model. The application communicates to the biometric technology (through the framework) in “API language”, whereas the biometric service provider (BSP) modules communicate (through the framework) to the application in “SPI language”. The translation is performed within the framework, which also handles module management. In the case of the BioAPI, the API and SPI definitions are nearly identical.
Functions

The specification is written as ‘C’ functions and data structures. The functions comprising the BioAPI fall into several categories as shown in the table below:

As an example, a typical sequence to perform a biometric verification using the BioAPI is shown in Figure 3. In step 1, the previously enrolled template is retrieved from the application’s user database. This template is passed into the BioAPI_Verify function in Step 2. The BSP executes the verify operation by performing a capture of live biometric data (Step 3), processing that data into a numeric identifier called a Biometric Identification Record (BIR) (Step 4), and performing a 1:1 match of the newly captured sample against the previously enrolled template (Step 5). In Step 6, the results of the operation, in terms of a boolean “true/false” as well as match score, are returned to the calling application. If enabled and supported, an optional return of the adapted template is also returned if a positive match was determined (verify result = true). Note that in Step 3, as a default the BSP is responsible for providing any user interface associated with the capture operation and for interfacing directly with the biometric sensor.

![Fig. 3: A typical sequence to perform a biometric verification using BioAPI](image)

---

### Management Operations

- **Framework functions**
  - BioAPI_Init: Initializes the BioAPI framework
  - BioAPI_Terminate: Releases the BioAPI framework
  - BioAPI_ModuleLoad: Loads a specific BSP
  - BioAPI_ModuleUnload: Unloads a previously loaded BSP
  - BioAPI_ModuleAttach: Attaches a specific BSP and device
  - BioAPI_ModuleDetach: Detaches a previously attached BSP

- **BSP functions**
  - Handle Operations:
    - BioAPI_FreeBIRHandle
    - BioAPI_GetBIRFromHandle
    - BioAPI_GetHeaderFromHandle
  - Callback & Event Operations:
    - BioAPI_EnableEvents
    - BioAPI_SetGUCallbacks
    - BioAPI_SetStreamCallbacks
    - BioAPI_StreamInputOutput
  - Set Power Mode
    - Sets device to requested power mode

### Biometric Operations

- **Basic functions**
  - BioAPI_Enroll: Capture biometric, create template, and store in user database
  - BioAPI_Verify: Capture live sample and match against one stored template
  - BioAPI_Identify: Capture live sample and match against set of stored templates

- **Primitive functions**
  - BioAPI_Capture: Capture live data for specified purpose
  - BioAPI_Process: Processes captured data for immediate verification or identification
  - BioAPI_CreateTemplate: Processes new or previously captured data to create an enrollment template
  - BioAPI_VerifyMatch: Performs a 1:1 match between 2 BIRs
  - BioAPI_ValidateMatch: Performs a 1:N match between a BIR and set of stored templates
  - BioAPI_Import: Non-real time import and processing of raw biometric data

### Database Operations

- **Database functions**
  - BioAPI_DbOpen: Opens specified database
  - BioAPI_DbClose: Closes an open database
  - BioAPI_DbCreate: Creates and opens a new database
  - BioAPI_DbDelete: Deletes a specified database
  - BioAPI_DbSetCursor: Sets cursor to point to specified record
  - BioAPI_DbStore: Stores a BIR record in specified DB
  - BioAPI_DbGetBIR: Retrieves a specified BIR record
  - BioAPI_DbGetNextBIR: Retrieves BIR from cursor position
  - BioAPI_DbQueryBIR: Returns pointer to BIR index
  - BioAPI_DbDeleteBIR: Deletes a specified BIR record

---

**GUI**

**Sensor**

**Verify** (h, template, result …)

**BioAPI**

**BSP**

**Match**

**Database**

Get template

“stored”

“live”

Compare “live” v “stored”

“stored” Responses Bitmap/stream

“live”

Capture

Process

Match

Template Application Database

Get template

“stored”

“live”

Capture

Process

Match

Fig. 3: A typical sequence to perform a biometric verification using BioAPI
**Data Format**

Within the BioAPI, biometric data is contained within a Biometric Identification Record (BIR), which refers to any biometric data returned to the application. This includes raw data (images, audio streams, etc.), intermediate data (suitable for processing), processed data (suitable for verification or identification matching), or enrollment data (reference template to be stored for later matching).

The BIR structure consists of a standard, fixed length header, an opaque biometric data block, and an optional digital signature, as shown in Figure 4.

The BIR header conforms to the Common Biometric Exchange File Format (CBEFF) published by NIST, due to the close coordination between the two groups to define a common header. The unique Format field in the header defines the format of the opaque biometric data, which may be a standard or proprietary format. Format Owner values are registered with the International Biometric Industry Association (IBIA). Format Type values are assigned by the Format Owner and may be optionally registered.

The BioAPI also allows a “payload” to be embedded within a biometric template and released only upon a positive verification match. The payload may consist of any application data (up to a BSP defined maximum size), such as a user's private key.

**Scoring and Thresholding**

BioAPI has further standardized the setting of thresholds and return of match scores by defining them in terms of False Accept Rate (FAR). This allows a common, or somewhat normalized, value across technologies and thus comparisons between them.

As an example, within the Verify function described above, the application sets the threshold by requesting a MaxFAR and optionally a MaxFRR as criteria for a successful match. If both values are provided, it also sets Precedence, to indicate which of these two values should take precedence should they conflict. The BSP, upon performing the match operation, will return as a score the FAR Achieved and optionally the FRR Achieved, indicating the closeness of the match. Support of FRR is not supported by all BSPs.

In order to foil any “hillclimbing” attack, the BSP may choose to return FAR Achieved as an incremental rather than continuous value, selecting the increment based on statistical data related to the resolution of data required by an attacker to use the returned scoring data to successfully mount such an attack. In this case, the BSP returns the nearest increment rather than the precise score achieved. The level of quantization required to neutralize this attack is dependent on the type of biometric and algorithm.

**Module Registry**

The BioAPI provides a platform independent module registry where the Framework and BSPs publish data about themselves. A schema for this registry is included in the specification. For example, upon installation the BSP posts general information about itself (UUID, version, etc.), default values, and information related to which options it supports.

Each BSP has a UUID based on a manufacturer generated GUID. This value is used to address the BSP for module management and to direct function calls (using a module handle returned when the BSP module is attached).

![Fig. 4: The BIR Structure](image)
Optional Capabilities

The BioAPI includes a number of useful optional capabilities, some of which have already been alluded to. These include the following:

- Return of raw/audit data
- Return of quality
- Application-controlled GUI
- GUI streaming callbacks
- Detection of source presence
- Payload carry
- Return of FRR
- BIR signing
- BIR encryption
- Model adaptation
- Binning
- Client/server communication
- BSP controlled databases
- Self-contained device

Client/server implementations within the BioAPI can be accomplished in two ways. One way, which is similar to the existing HA-API implementations, is to use the primitive functions, distributing the operations between the two machines, with communications handled by the application. For example, the Capture and Process functions might be performed on the client while the Verify_Match operation is performed on the server. The second method is to use a Client/Server BSP which utilizes the streaming callback functions to tunnel through the application for direct Client BSP to Server BSP communication. In this case, the Verify call can be made at either the client or the server, with the execution of the component operations occurring in a distributed manner.

Another important option is application control of the user interface. As a default, the BSP provides the user interface during operations requiring user interaction, such as during Capture and Enrollment. However, the BSP can optionally allow the application to control the user interface, to provide customization of the look and feel. In this case, events are predefined and GUI callbacks are implemented, including the option of streaming callbacks. The BSP then provides the data to be displayed to the application that determines the manner in which it is to be displayed.

Reference Implementation

The BioAPI reference implementation is the software implementing the BioAPI framework. It is the “middleware” between BioAPI compliant applications and service providers. It provides for module loading/attaching, module management, the module registry, and passthrough of calls (i.e., API-to-SPI translation). Framework components include:

- Framework
- Module Directory Services (MDS)
- Sample Password BSP
- Sample Application/Exerciser
- Installers
The first target operating system for the Framework is the Win-32 operating systems (Microsoft Windows NT, '95, '98, and 2000). The BioAPI Framework is designed for easy porting to other operating systems, such as Linux. The reference implementation is open source and available royalty free. The beta version became available in September and Ver 1.0 is due soon after year-end. A Linux port is scheduled in the following quarter.

**Relationship with Other Standards**

Through its External Liaisons Working Group, the BioAPI has developed strong relationships with other related biometric standards efforts. As a result, the BioAPI specification has been coordinated with and is compatible with a number of other specifications and standards.

Working with Intel, the BioAPI Specification was used as the basis for the Human Recognition Services (HRS) extension (formerly called UAS) to The Open Group’s Common Data Security Architecture (CDSA) standard. With the exception of function/parameter names and the addition of some CDSA specific capabilities, the HRS and BioAPI specification are identical.

BioAPI was a chief contributor to NIST’s Common Biometric Exchange File Format (CBEFF) specification, which describes a set of data elements necessary to support biometric technologies in a common way to promote interoperability of biometric based application programs and systems by allowing biometric data interchange. Thus the BioAPI BIR is CBEFF conformant and represents one of the CBEFF Patron formats.

The BioAPI Consortium also worked closely with the ANSI X9.84 working group defining the soon to be released X9.84 Biometric Management and Security standard. The X9.84 standard is also a CBEFF Patron format that uses common data elements, but formats them using ASN.1 encoding techniques.

Other established relationships include the IBIA, the BioTrust group of TeleTrust, the ECTF, Trusted Computing Platform Alliance (TCPA), and Eurosmart.

**Competing Standards**

In May of 2000, Microsoft announced its plans to incorporate the BAPI as a proprietary standard into some future version of the Windows operating system, although commitments had been made by BAPI author I/O Software to provide it as the lower device level interface of BioAPI. Although this news was disappointing, the BioAPI Consortium has expressed willingness to work with Microsoft to ensure compatibility of the two standards. This is especially important at the data format level. Since BioAPI runs on Microsoft platforms and supports cross-platform implementations and heterogeneous environments, it will provide versatility and freedom of choice for implementers.

**Status and Plans**

Version 1.1 of the BioAPI Specification is in progress. This version incorporates changes resulting from the reference implementation development as well as feedback from adopters. This is expected to be available early in the first quarter. Additionally, the final version of the reference implementation will also be released soon after year-end. A Linux version of the reference implementation is also expected in the first quarter of 2001. Work is also in progress on a conformance test suite for the BioAPI.

Vendors are busily working on BioAPI compliant products. Mytec Technologies demonstrated a BioAPI compliant demo application running with the beta version of the BioAPI reference implementation and two BPSs (their BioScrypt® device and the sample password BSP) at the Biometric Consortium 2000 conference this past September. Other BioAPI Consortium members have BioAPI BSPs in beta. It is expected that BioAPI compliant products will be commercially available in the first quarter of 2001.

**Conclusion**

Why is the BioAPI important and to whom?

The BioAPI is important to **vendors** because it allows them to compete based on features, performance, and quality; provides compatibility across a broad range and quantity of applications, opens new markets to the technology (that were hesitant to adopt due to the lock-in effect); creates a level playing ground for customer access; permits them to write their code once and port it to many environments; and allows them to capitalize on associated industry efforts (e.g., CDSA).

It is important to **integrators** because in addition to generally making biometric integration easier, it also provides a broader choice/selection of technologies to integrate, selectable levels of sophistication and control, configuration flexibility in their applications/systems, and support for test/evaluation. It also allows for them to matrix multiple applications with multiple technologies. The biggest benefit, however, is that it lowers their integration risks.
It is important to end-users and customers because it allows them the flexibility to configure their systems to best fit their user population, applications, and environment. They are no longer locked in to a point solution or proprietary technology (algorithm or hardware). It allows for ease of technology refreshment and system upgrades. The common data format facilitates interoperability. It supports biometric “plug-and-play”. And it will allow them to use biometrics in a broader context of platform environments and biometrically enabled applications – both horizontal packaged security products (such as network authentication) as well as more customized applications.

It is important to the enterprise because it addresses the needs of a biometric rollout across a large-scale organization. This is true for several reasons.

- First, it is because standards have the potential to influence costs/seat for biometric products downward as biometrics become more of a commodity item and competition is fostered.
- Second, it provides the flexibility to accommodate the diverse user population within a large enterprise. This will include those who are poor candidates for a particular biometric, those with physical disabilities that preclude the use of a particular biometric, and those with religious concerns. In each case, the availability of an alternate biometric can alleviate these situations.
- Third, it allows for accommodation of various environments within a geographically dispersed and multi-function enterprise. For example, conditions such as interference sources or environmental conditions can affect biometric selection as would other environmental constraints such as space limitations or areas where employees must be gloved or masked. In these cases, different biometrics can be installed in each location.
- Fourth, an enterprise must always look forward to future technology evolution and the cost to migrate with it. A standard such as BioAPI allows them to easily upgrade as new biometric technologies are discovered and become available, to expand the number of technologies they support, to substitute one biometric type/technology for another, to replace vendors, and to easily install revision updates within a given technology.
- Lastly, enterprises cannot tolerate lock-in to one technology or vendor with a point solution or proprietary implementation. This is due to cost and risk concerns.

Version 1.0 of the BioAPI Specification (released 30 March 2000) provides for simple biometric application interfaces; standard access methods to biometric functions, algorithms, and devices; secured and robust biometric data management and storage; standard methods of differentiating biometric data and device types; and support for biometric identification in distributed computing environments. BioAPI also standardizes biometric data formats (required for system interoperability) and normalizes scoring & thresholding (needed for comparisons). It provides configuration flexibility through basic and primitive operations. BioAPI is an industry consensus standard that has widespread, international support.

The BioAPI Specification is available in the public domain, royalty free, and is downloadable from the BioAPI website. The reference implementation is open source and is also royalty free and downloadable from the web.