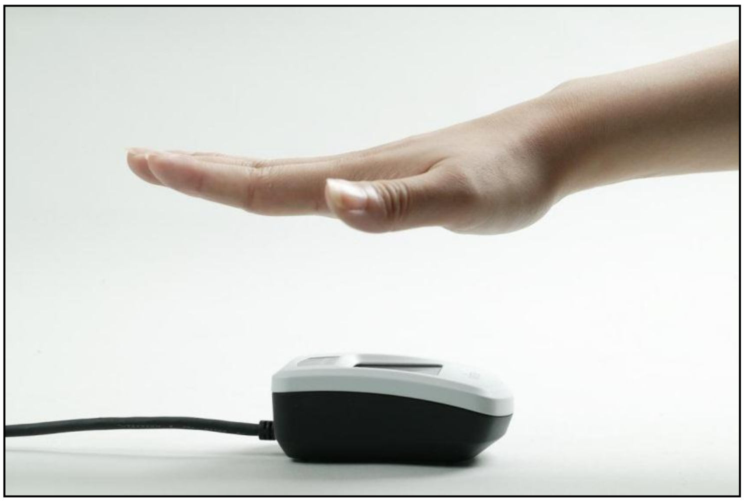
U4PS-LCA2-01ENZ3

PalmSecure™ SDK V02

Sample Interface Module for Java

V03 Manual

(Windows (x86) Version / Windows (x64) Version / Linux (x86)Version)



FUJITSU

◆ Revision History

Revision	Issued Date	Revised Page	Modification Details
1st Rev.	Oct 2013	Entire document	Newly created

◆ Introduction

Thank you for purchasing PalmSecure™ SDK V02 (hereinafter called "this product").

This document explains how to use the Sample interface module for Java. It is intended for readers who have a basic knowledge of the following.

- Windows or Linux operations
- Java Virtual Machine
- Java language

This sample Interface module is provided in order to help customer's application development, and it is not guaranteed to work properly in customer's actual environment. Please note that it is customer's responsibility to do the quality assurance.

October 2013

October 2013 : First Edition

Regarding to High Safety Required Usage

This Product is designed, developed and manufactured as contemplated for general use, including without limitation, general office use, personal use, household use, and ordinary industrial use, but is not designed, developed and manufactured as contemplated for use accompanying fatal risks or dangers that, unless extremely high safety is secured, could lead directly to death, personal injury, severe physical damage or other loss (hereinafter “High Safety Required Use”), including without limitation, nuclear reaction control in nuclear facility, aircraft flight control, air traffic control, mass transport control, medical life support system, missile launch control in weapon system. You shall not use this Product without securing the sufficient safety required for the High Safety Required Use. If you wish to use this Product for High Safety Required Use, please consult with the sales representatives in charge before such use.

Cautions for Exporting This Product

When exporting or providing this product and this document to a foreign country, check relevant laws such as "Foreign Exchange and Foreign Trade Control Law", and regulations such as U.S. export control law, and follow the necessary procedures.

Warnings

- It is forbidden to copy all or part of this document without permission.
- Items described in this document are subject to change without previous notice.

PalmSecure is a trademark of Fujitsu Ltd.

Microsoft, Windows, Visual C++, and Visual Studio are registered trademarks of Microsoft Corporation in the United States and other countries.

Java and all trademarks and logos based on Java are trademarks or registered trademarks of Oracle Corporation in the U.S. and other countries.

Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

BioAPI is a trademark of BioAPI Consortium.

Other company names and product names described in this document are trademarks or registered trademarks of each company.

◆ Abbreviations and Common Terms

Abbreviations and common terms used in this document are as follows:

Abbreviations/ Common Term	Description
This product	Abbreviation for “PalmSecure™ SDK V02”.
Interface module	Abbreviation for “Sample interface module for Java V03”.
Old Interface library	Abbreviation for “Interface library sample for Java V01”.
Authentication library	Abbreviation for “Authentication library V32 Professional Edition”.
"Authentication Library Reference Guide"	Abbreviation for “Authentication Library V32 Reference Guide”.
PalmSecure Sensor	Abbreviation for “PalmSecure™ Sensor”.
PalmSecure Sensor V2	Abbreviation for “PalmSecure™ Sensor V2”.
Sensor	Common term for “PalmSecure Sensor” and “PalmSecure Sensor V2”.
Windows 7	Abbreviation for “Microsoft® Windows® 7”.
Windows 8	Abbreviation for “Microsoft® Windows® 8”.
Windows 8.1	Abbreviation for “Microsoft® Windows® 8.1”.
Windows	Common term for “Windows 7” , “Windows 8” and “Windows 8.1”.
Visual C++ 2010	Abbreviation for “Microsoft® Visual C++® 2010”.
BIR class	Abbreviation for “JAVA_BioAPI_BIR class”.
BIR handle	Indicates “handle to identify JAVA_BioAPI_BIR class”.
JNI	Abbreviation for “Java Native Interface”.
JRE	Abbreviation for “Java Runtime Environment”.
JDK	Abbreviation for “Java Development Kit”.

◆ Notations

The following symbols are used in this document.

Symbol	Description
!Caution	Describes things that you have to look out for. You must read it.
★Tip	Provides reference information. Read it if necessary.
>See>	Indicates an item to be referred.
✋Operation	Describes operation procedures.
[ ] button	Indicates a button shown on the screen.

◆ Table of Contents

Chapter1 Introduction of the Interface module

1.1 Overview

1.2 List of Contents

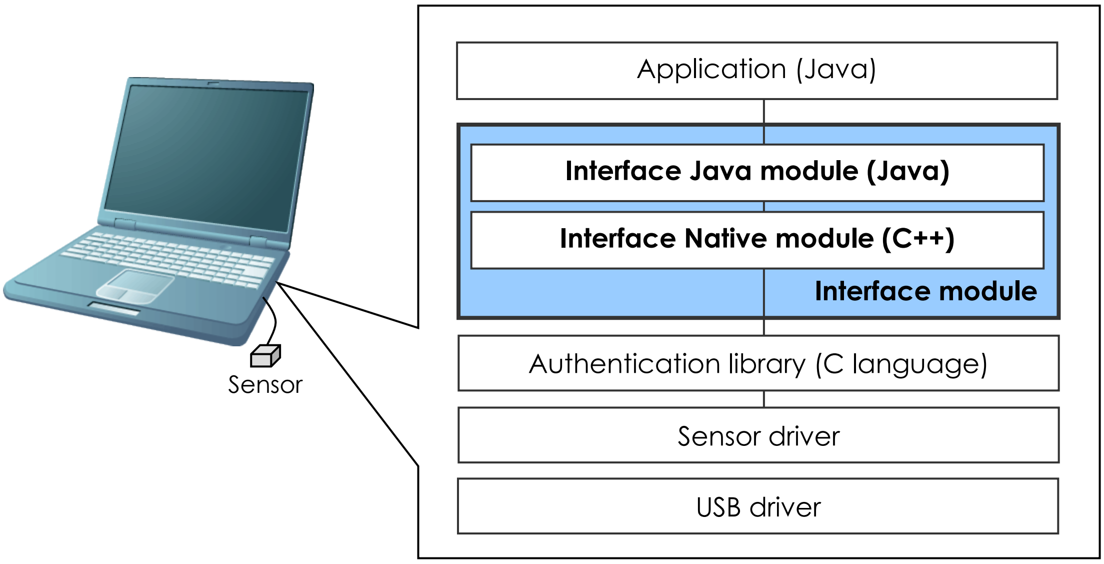
1.1 Overview

This Interface module is provided to call functions supported by Authentication library from Java application, which could be useful for application development using Java.

Interface module uses JNI in order to call Authentication library from Java application.

Sample application for Java based on the Interface module is also provided.

The software structure including the Interface module is as follows.


Sensor
Application (Java)
Interface Java module (Java)
Interface Native module (C++)
Interface module
Authentication library (C language)
Sensor driver
USB driver

>See> As for the Sample application for Java, refer to the " Sample Application for Java Manual ".

1.2 List of Contents

Interface module contains the following files.

◆ List of Provided Files

Folder				Stored file / folder	Description
1st Hierarchy	2nd Hierarchy	3rd Hierarchy	4th Hierarchy		
PalmSecure Interface ForJava	java	release		f3bc4jav.jar	Class file to call Authentication library from Java application (Note)
		source	f3bc4jav	JavaBuild.bat	Batch file for Windows to create class file from source program
				com	Source program (For detail, refer to ★Tip "Source program of Interface Java module") (Note)
	native	release	linux	libf3bc4jni.so	Library to handle native code
			windows		
			x64	F3BC4JNI.DLL	
			x86	F3BC4JNI.DLL	
		source	PalmSecure Interface ForJNI	Include	Folder to store the Lib file of Authentication library
				Lib	Folder to store the header file of Authentication library
				PalmSecure Interface ForJNI	Project (For detail, refer to ★Tip "Project of Interface Native module") (Note)

Note) Common for Windows and Linux.

★Tip Source program of Interface Java module

- The following table shows the details of the source program of Interface Java module.

Package					Stored file
1st Hierarchy	2nd Hierarchy	3rd Hierarchy	4th Hierarchy	5th Hierarchy	
com	Fujitsu	frontech	palm secure	util	ConvertEndian.java
					PalmSecureConstant.java
					PalmSecureException.java
					PalmSecureHelper.java
				-	JAVA_BioAPI_BIR.java
					JAVA_BioAPI_BIR_ARRAY_POPULATION.java
					JAVA_BioAPI_BIR_BIOMETRIC_DATA_FORMAT.java
					JAVA_BioAPI_BIR_HEADER.java
					JAVA_BioAPI_CALLOC_IF.java
					JAVA_BioAPI_CANDIDATE.java
					JAVA_BioAPI_DATA.java
					JAVA_BioAPI_DBBIR_ID.java
					JAVA_BioAPI_FREE_IF.java
					JAVA_BioAPI_FUNC_NAME_ADDR.java
					JAVA_BioAPI_GUI_BITMAP.java
					JAVA_BioAPI_GUI_STATE_CALLBACK_IF.java
					JAVA_BioAPI_GUI_STREAMING_CALLBACK_IF.java
					JAVA_BioAPI_IDENTIFY_POPULATION.java
					JAVA_BioAPI_INPUT_BIR.java
					JAVA_BioAPI_MALLOC_IF.java
					JAVA_BioAPI_MEMORY_FUNCS.java
					JAVA_BioAPI_ModuleEventHandler_IF.java
					JAVA_BioAPI_PROC_ADDR_IF.java
					JAVA_BioAPI_REALLOC_IF.java
					JAVA_BioAPI_VERSION.java
					JAVA_PvAPI_ErrorInfo.java
					JAVA_PvAPI_LBINFO.java
					JAVA_PvAPI_SensorInfo.java
					JAVA_PvAPI_SensorInfoEx.java
					JAVA_sint32.java
					JAVA_sint8.java
					JAVA_uint32.java
					JAVA_uint8.java

Package					Stored file
1st Hierarchy	2nd Hierarchy	3rd Hierarchy	4th Hierarchy	5th Hierarchy	
com	fujitsu	frontech	palm secure	-	LOCAL_GUI_STATE_CB.java
					LOCAL_GUI_STREAMING_CB .java
					LOCAL_INT.java
					LOCAL_LONG.java
					LOCAL_SHORT.java
					PalmSecureIf.java

★Tip **Project of Interface JNI library**

- The following table shows the details of the project of Interface Native module.

Folder			Stored file / folder	Description
1st Hierarchy	2nd Hierarchy	3rd Hierarchy		
PalmSecure InterfaceFor JNI	-	-	app.rc	-
			Makefile	Makefile to create a library for Linux
			PalmSecureIf.cpp	-
			PalmSecureIf.h	-
			PalmSecureInterfaceForJNI.sln	Solution file
			PalmSecureInterfaceForJNI.vcxproj	Project file
			PalmSecureInterfaceForJNI.vcxproj .filters	
			PalmSecureUtil.cpp	-
			PalmSecureUtil.h	-
			resource.h	-

Chapter2 Before Using the Interface Module

2.1 Hardware and Software Requirements

In order to use Interface module, the following hardware and software are necessary.

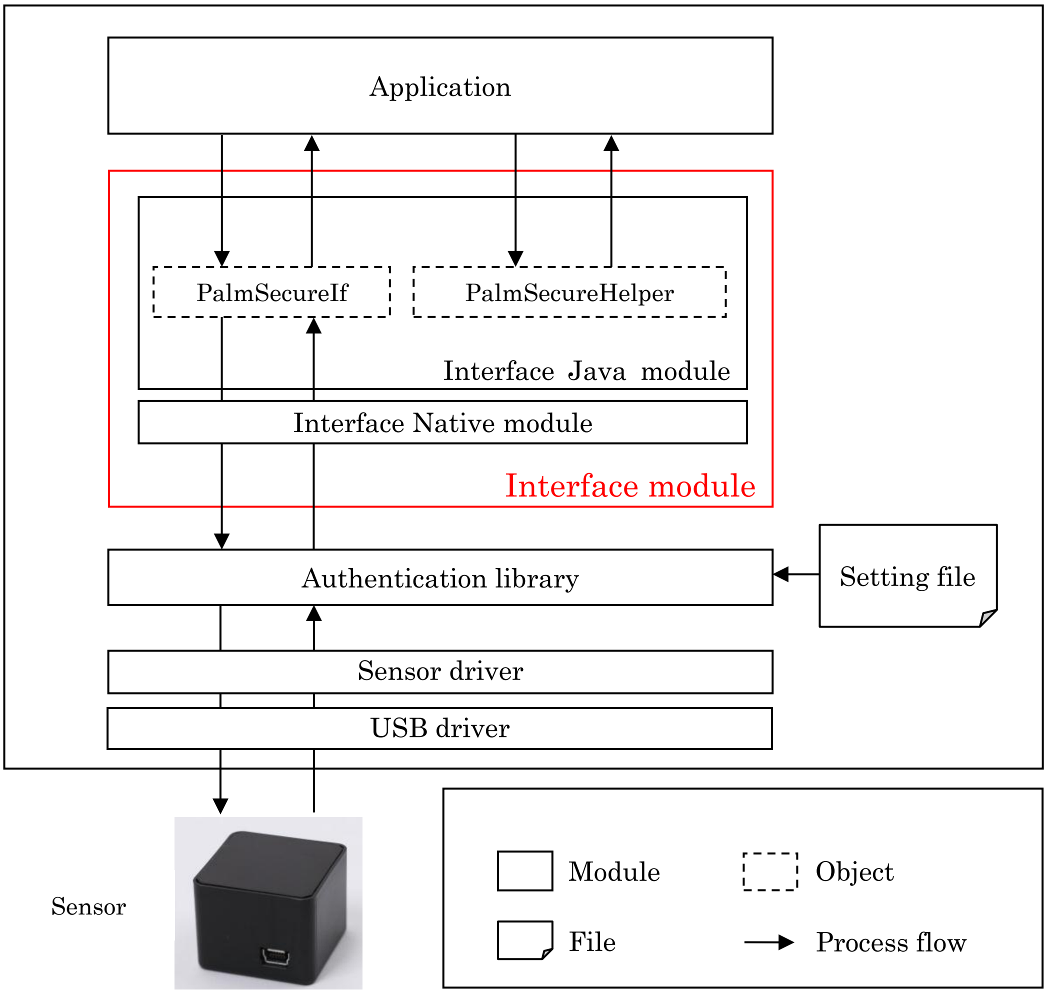
Hardware and Software Requirements		Description
Hardware Requirements		Please refer to the “Authentication Library Reference Guide”.
Software Requirements	Tested OS (Note1)	• Windows 7 Professional SP1 (x86 / x64) • Windows 8 Pro (x86 / x64) • Windows 8.1 Pro (x86 / x64) • CentOS 6.4 (x86) (kernel 2.6.32-358.el6.i686)
	Sensor driver	Please refer to the “Authentication Library Reference Guide”.
	Authentication library	• Windows(x86) version Professional Edition V32L10-B01 • Windows(x64) version Professional Edition V32L60-B01 • Linux(x86) version Professional Edition V32L27-B01
	Interface module (Note 2)	Please download and use the latest version from SDK Support Website.
	Tested JRE	1.7.0_17
	Application	Java application which runs on the above JRE

Note 1) Usage on virtual environment is not tested (such as XP mode in Windows 7). Please test it by yourself when you use this product on virtual environment.

Note 2) Since the interface Native module for Windows is developed based on Microsoft Visual C++2010, some environments require installation of “Visual C++ 2010 Redistributable Package”. Please download and install the proper package from the website of Microsoft Co. if necessary.

2.2 Software Structure

The following diagram shows software structure including Interface module.



No.	Module		Description
1	Application		Java application
2	Interface Java module		JAR file which contains classes to call functions of Authentication library from application.
	(1)	PalmSecureIf	PalmSecureIf is the name of class to call functions of Authentication library from application.
	(2)	PalmSecureHelper	PalmSecureHelper class is the name of class which provides help method to convert output data from PalmSecureIf class.
3	Interface Native module		Interface library to call Authentication library for C language from PalmSecureIf class.

2.3 Installation of Interface Module

Please install Interface module as follows.

Operation

Step1 Confirm the following settings on target hardware environment.

1. Installation of PalmSecure Sensor driver
2. Installation of Authentication library
3. Setting of "PvAPI.INI" file of Authentication library
4. Confirmation of the firmware version of Sensor unit and update of the firmware if necessary

>See> For installation of PalmSecure Sensor driver, refer to the "Sensor Driver Installation Guide".

>See> For installation of Authentication library, refer to the "Authentication Library Reference Guide".

>See> For confirmation of firmware version, refer to the "Sensor Maintenance Tool Operation Guide".

>See> For update of firmware to the latest version, refer to the "System Development Guide".

Step2 Confirm whether Java platform is enabled by the following steps.

➢ **Windows Version:**

1. Start "Command Prompt".
2. Enter the following command.

   java -version

3. Confirm that Java version is shown.

➢ **Linux Version:**

1. Start "Terminal."
2. Enter the following command.

   java -version

3. Confirm that Java version is shown.

Step3 As for Windows usage, download and install "Microsoft Visual C++ 2010 Redistributable Package" from the website of Microsoft Co. in order to execute an application on target hardware which has not been installed Visual C++ 2010.

!Caution **In case Visual C++ 2010 or "Microsoft Visual C++ 2010 Redistributable Package" is not installed on target hardware**

- Since Interface module cannot load necessary library, it is not possible to run application.

Step4 Copy Interface Native module ("F3BC4JNI.DLL" or "libf3bc4jni.so") to the same folder as Authentication library.

The Interface Native module is stored in the following folder.

- **Windows (x86) Version:**

  "\PalmSecureInterfaceForJava\release\ native\windows\x86"
- **Windows (x64) Version:**

  "\PalmSecureInterfaceForJava\release\ native\windows\x64"
- **Linux (x86) Version:**

  "\PalmSecureInterfaceForJava\release\ native\linux"

Step5 Set Interface Java module ("f3bc4jav.jar") to classpath and start an application.

Chapter3 Interface

3.1 Package and Class

The Following table shows package and class contained in Interface module.

Package	com.fujitsu.frontech.palmsecure	
No.	**Class name**	**Description**
1	JAVA_BioAPI_BIR	Class corresponds to structures and types of Authentication library.
2	JAVA_BioAPI_BIR_ARRAY_POPULATION	
3	JAVA_BioAPI_BIR_BIOMETRIC_DATA_FORMAT	
4	JAVA_BioAPI_BIR_HEADER	
5	JAVA_BioAPI_CALLOC_IF	
6	JAVA_BioAPI_CANDIDATE	
7	JAVA_BioAPI_DATA	
8	JAVA_BioAPI_DBBIR_ID	
9	JAVA_BioAPI_FREE_IF	
10	JAVA_BioAPI_FUNC_NAME_ADDR	
11	JAVA_BioAPI_GUI_BITMAP	
12	JAVA_BioAPI_GUI_STATE_CALLBACK_IF	
13	JAVA_BioAPI_GUI_STREAMING_CALLBACK_IF	
14	JAVA_BioAPI_IDENTIFY_POPULATION	
15	JAVA_BioAPI_INPUT_BIR	
16	JAVA_BioAPI_MALLOC_IF	
17	JAVA_BioAPI_MEMORY_FUNCS	
18	JAVA_BioAPI_ModuleEventHandler_IF	
19	JAVA_BioAPI_PROC_ADDR_IF	
20	JAVA_BioAPI_REALLOC_IF	
21	JAVA_BioAPI_VERSION	
22	JAVA_PvAPI_ErrorInfo	
23	JAVA_PvAPI_LBINFO	
24	JAVA_PvAPI_SensorInfo	
25	JAVA_PvAPI_SensorInfoEx	
26	JAVA_sint32	
27	JAVA_sint8	
28	JAVA_uint32	
29	JAVA_uint8	
30	PalmSecureIf	Class provides main method. In order to call Authentication library, create one instance of this class and always call its method.

Package	com.fujitsu.frontech.palmsecure.util	
No.	Class name	Description
1	PalmSecureHelper	Class provides help method to exchange data format between instance of BIR class and byte array. (1) Convert instance of BIR class to byte array: In order to output enrollment data and/or authentication data to file, database and so on. (2) Convert byte array to instance of BIR class: In order to acquire instance of BIR class from enrollment data and/or authentication data stored in file, database and so on.
2	PalmSecureConstant	Class defines constants used in Interface module.
3	PalmSecureException	Class thrown in case an error occurs in Interface module.

3.2 PalmSecureIf Class

3.2.1 List of Methods

“PalmSecureIf” class provides the following methods.

No.	Method	Description	Reference
1	PalmSecureIf	Constructor	-
2	JAVA_BioAPI_ModuleLoad	Method corresponds to each function of Authentication library. Method corresponds to each function of Authentication library.	3.2.2.1
3	JAVA_BioAPI_ModuleUnload		3.2.2.2
4	JAVA_BioAPI_ModuleAttach		3.2.2.3
5	JAVA_BioAPI_ModuleDetach		3.2.2.4
6	JAVA_BioAPI_FreeBIRHandle		3.2.2.5
7	JAVA_BioAPI_GetBIRFromHandle		3.2.2.6
8	JAVA_BioAPI_SetGUICallbacks		3.2.2.7
9	JAVA_BioAPI_Capture		3.2.2.8
10	JAVA_BioAPI_VerifyMatch		3.2.2.9
11	JAVA_BioAPI_IdentifyMatch		3.2.2.10
12	JAVA_BioAPI_Enroll		3.2.2.11
13	JAVA_BioAPI_Verify		3.2.2.12
14	JAVA_BioAPI_Identify		3.2.2.13
15	JAVA_PvAPI_ApAuthenticate		3.2.2.14
16	JAVA_PvAPI_SetProfile (setup for JAVA_uint32)		3.2.2.15
17	JAVA_PvAPI_SetProfile (setup for String)		3.2.2.16
18	JAVA_PvAPI_GetErrorInfo		3.2.2.17
19	JAVA_PvAPI_Sense		3.2.2.18
20	JAVA_PvAPI_Cancel		3.2.2.19
21	JAVA_PvAPI_PreSetProfile		3.2.2.20
22	JAVA_PvAPI_PresetIdentifyPopulation		3.2.2.21
23	JAVA_PvAPI_GetConnectSensorInfoEx		3.2.2.22
24	JAVA_PvAPI_GetLibraryInfo		3.2.2.23
25	JAVA_BioAPI_GUI_STATE_CALLBACK		3.2.2.24
26	JAVA_BioAPI_GUI_STREAMING_CALLBACK		3.2.2.25

>See> As for the function of Authentication library corresponding to the method of “PalmSecureIf” class, refer to the “Authentication Library Reference Guide”.

3.2.2 Reference for Method

Methods of “PalmSecureIf” class converts parameters between Java code and native code, and call each function of Authentication library.

For “Return value”, it returns value acquired from Authentication library as it is.

For “Unused” parameter, set the following value.

- For class or instance: specify “null”
- Other than the above: specify “0”.

!Caution **In order to call Authentication library**

Create only one instance of “PalmSecureIf” class and always call methods of the instance.

>See> As for error information, refer to the “ Chapter4 Error Information ”.

3.2.2.1 JAVA_BioAPI_ModuleLoad

[Function]

This method registers Authentication library modules.

[Coding syntax]

```
public long JAVA_BioAPI_ModuleLoad
 (
   byte[]                             ModuleGuid,
   JAVA_uint32                        Reserved,
   JAVA_BioAPI_ModuleEventHandler_IF  AppNotifyCallback,
   Object                             AppNotifyCallbackCtx
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleGuid	input	Using byte array, specify the UUID of Authentication library; “0xe1, 0x9a, 0x69, 0x01, 0xb8, 0xc2, 0x49, 0x80, 0x87, 0x7e, 0x11, 0xd4, 0xd8, 0xf1, 0xbe, 0x79”.
Reserved	input	Reserved Specify null.
AppNotifyCallback	input/ optional	Unused
AppNotifyCallbackCtx	input/ optional	Unused

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “BioAPI_ModuleLoad” in Authentication library.

3.2.2.2 JAVA_BioAPI_ModuleUnload

[Function]

This method releases Authentication library modules.

[Coding syntax]

```
public long JAVA_BioAPI_ModuleUnload
 (
   byte[]                              ModuleGuid,
   JAVA_BioAPI_ModuleEventHandler_IF   AppNotifyCallback,
   Object                              AppNotifyCallbackCtx
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleGuid	input	Specify the same UUID specified in module registration.
AppNotifyCallback	input/ optional	Unused
AppNotifyCallbackCtx	input/ optional	Unused

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “BioAPI_ModuleUnload” in Authentication library.

[Caution]

If “JAVA_BioAPI_ModuleLoad” has already been called, be sure to call this method before termination of application.

3.2.2.3　JAVA_BioAPI_ModuleAttach

[Function]

This method attaches Authentication library modules.

[Coding syntax]

```
public long JAVA_BioAPI_ModuleAttach
 (
   byte[]                          ModuleGuid,
   JAVA_BioAPI_VERSION             Version,
   JAVA_BioAPI_MEMORY_FUNCS        MemoryFuncs,
   JAVA_uint32                     DeviceID,
   JAVA_uint32                     Reserved1,
   JAVA_uint32                     Reserved2,
   JAVA_uint32                     Reserved3,
   JAVA_BioAPI_FUNC_NAME_ADDR      FunctionTable,
   JAVA_uint32                     NumFunctionTable,
   Object                          Reserved4,
   JAVA_uint32                     NewModuleHandle
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleGuid	input	Specify the same UUID specified in module registration.
Version	input	Unused
MemoryFuncs	input	Unused
DeviceID	input	Unused
Reserved1	input	Unused
Reserved2	input	Unused
Reserved3	input	Unused
FunctionTable	input/ output/ optional	Unused
NumFunctionTable	input	Unused
Reserved4	input	Unused
NewModuleHandle	input/ output	Create and specify the instance of “JAVA_uint32” class. Handle of attached module is set in the instance.

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “BioAPI_ModuleAttach” in Authentication library.

[Remark]

In this method, a process of firmware update for Sensor unit is performed as necessary before starting sensor.

>See> In order to update firmware, refer to the “Authentication Library Reference Guide”.

3.2.2.4　JAVA_BioAPI_ModuleDetach

[Function]

This method detaches Authentication library modules.

[Coding syntax]

```
public long JAVA_BioAPI_ModuleDetach
 (
   JAVA_uint32                    ModuleHandle
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of attached module.

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “BioAPI_ModuleDetach” in Authentication library.

[Caution]

If “JAVA_BioAPI_ModuleAttach” has already been called, be sure to call this method and “JAVA_BioAPI_ModuleUnload” before termination of application.

3.2.2.5 JAVA_BioAPI_FreeBIRHandle

[Function]

This method releases BIR handle.

[Coding syntax]

```
public long JAVA_BioAPI_FreeBIRHandle
 (
   JAVA_uint32                    ModuleHandle,
   JAVA_sint32                    BIRHandle
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of attached module.
BIRHandle	input	Specify the BIR handle to be released.

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “BioAPI_FreeBIRHandle” in Authentication library.

[Caution]

Release all BIR handles before calling “JAVA_BioAPI_ModuleDetach”.

If BIR handles are not released before calling "JAVA_BioAPI_ModuleDetach", it is not possible to release the BIR handles.

3.2.2.6　JAVA_BioAPI_GetBIRFromHandle

[Function]

This method acquires BIR class associated with BIR handle.

[Coding syntax]

```
public long JAVA_BioAPI_GetBIRFromHandle
 (
   JAVA_uint32                       ModuleHandle,
   JAVA_sint32                       BIRHandle,
   JAVA_BioAPI_BIR                   BIR
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of attached module.
BIRHandle	input	Specify the BIR handle acquired by the following methods. ・JAVA_BioAPI_Capture ・JAVA_BioAPI_Enroll
BIR	Input/ output	Create and specify the instance of “JAVA_BioAPI_BIR” class. Palm vein data is stored in the instance.

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “BioAPI_GetBIRFromHandle” in Authentication library.

3.2.2.7 JAVA_BioAPI_SetGUICallbacks

[Function]

This method sets event listener for status notification and for guidance image notification in Interface module.

[Coding syntax]

```
public long JAVA_BioAPI_SetGUICallbacks
 (
   JAVA_uint32                        ModuleHandle,
   JAVA_BioAPI_GUI_STREAMING_CALLBACK_IF GuiStreamingCallback,
   Object                             GuiStreamingCallbackCtx,
   JAVA_BioAPI_GUI_STATE_CALLBACK_IF    GuiStateCallback,
   Object                             GuiStateCallbackCtx
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of attached module.
GuiStreamingCallback	input	Specify the instance of the class implements the "JAVA_BioAPI_GUI_STREAMING_CALLBACK_IF" interface (event listener for guidance image notification), to notify guidance image for guiding the palm to correct position. Or specify null if not necessary to show the guidance image.
GuiStreamingCallback Ctx	input	Specify the object notified to application as a parameter of the “JAVA_BioAPI_GUI_STREAMING_CALLBACK” method (event handler for guidance image notification). Or specify null if not necessary to show the guidance image.
GuiStateCallback	input	Specify the instance of the class implements the “JAVA_BioAPI_GUI_STATE_CALLBACK_IF” interface (event listener for status notification).
GuiStateCallbackCtx	input	Specify the object notified to application as a parameter of “JAVA_BioAPI_GUI_STATE_ CALLBACK” method (event handler for status notification).

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “BioAPI_SetGUICallbacks” in Authentication library.

[Remark 1]

Set the following, and it is possible to notify the information to application for guiding the palm to correct position while capturing palm vein.

Three type of information can be notified.

1. Guidance for guiding palm to correct position
2. Silhouette image of captured palm
3. Guidance image for guiding palm to correct position

In order to notify information 1 and 2 of the above, set as follows in advance.

- Create an instance of the class implements the "JAVA_BioAPI_GUI_STATE_CALLBACK_IF" interface, and set it in Interface module using this method.

>See> For information on the “JAVA_BioAPI_GUI_STATE_CALLBACK”, refer to the ”3.2.2.24 JAVA_BioAPI_GUI_STATE_CALLBACK”.

In order to notify information 3 of the above, set as follows in advance.

- Set “detailed information notification function for guidance image display” of setting file of Authentication library as ”CBGUIMessageDetail” with “1” (Use) .
- Create an instance of the class implements the “JAVA_BioAPI_GUI_STREAMING_CALLBACK_IF” interface and set it in Interface module using this method.

>See> As for the “PvAPI.INI” file (setting file of Authentication library), refer to the “Authentication Library Reference Guide”.

>See> For information on the “JAVA_BioAPI_GUI_STREAMING_CALLBACK”, refer to the “3.2.2.25 JAVA_BioAPI_GUI_STREAMING_CALLBACK”.

[Remark 2]

The object set using this method can be used freely in event handler by application.

3.2.2.8 JAVA_BioAPI_Capture

[Function]

This method captures palm vein of one hand and creates authentication data.

>See> As for the structure of authentication data, refer to the "Authentication Library Reference Guide".

[Coding syntax]

```
public long JAVA_BioAPI_Capture
 (
   JAVA_uint32                        ModuleHandle,
   JAVA_uint8                         Purpose,
   JAVA_sint32                        CapturedBIR,
   JAVA_sint32                        Timeout,
   JAVA_sint32                        AuditData
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of attached module.
Purpose	input	Create an instance of "JAVA_uint8" class and set the setting value for verification (Fixed to "JAVA_BioAPI_PURPOSE_VERIFY") to field "value" of the instance. Then specify the instance to this parameter.
CapturedBIR	input/ output	Create and specify the instance of "JAVA_sint32" class. BIR handle of authentication data is set to the instance.
Timeout	input	Create an instance of "JAVA_sint32" class and set timeout duration in millisecond to field "value" of the instance. However, specify "0" (no timeout) at present. Then specify the instance to this parameter.
AuditData	output/ optional	Unused

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls "BioAPI_Capture" in Authentication library.

[Caution 1]

Release the BIR handle acquired using this method by calling “JAVA_BioAPI_FreeBIRHandle”.

[Caution 2]

Processing time for capturing is slightly longer if “image compression function” is used.

>See>　For information on the “image compression function”, refer to the “System Development Guide”.

[Remark 1]

Calling this method, “JAVA_BioAPI_GUI_STATE_CALLBACK” event handler is called by Authentication library. Authentication library sets the guidance for guiding the palm to correct position and so on to the parameter, and calls the event handler.

In case “JAVA_BioAPI_GUI_STREAMING_CALLBACK_IF” event listener is set to Interface module, “JAVA_BioAPI_GUI_STREAMING_CALLBACK” event handler is called by Authentication library. Authentication library sets the guidance image for guiding the palm to correct position to the parameter, and calls the event handler.

>See>　For information on the “JAVA_BioAPI_GUI_STATE_CALLBACK”, refer to the ”3.2.2.24　JAVA_BioAPI_GUI_STATE_CALLBACK”.

>See>　For information on the “JAVA_BioAPI_SetGUICallbacks”, refer to the ”3.2.2.7　JAVA_BioAPI_SetGUICallbacks”.

>See>　For information on the “JAVA_BioAPI_GUI_STREAMING_CALLBACK”, refer to the “3.2.2.25　JAVA_BioAPI_GUI_STREAMING_CALLBACK”.

[Remark 2]

In case of specifying “compressed format” for enrollment format of palm vein data, only enrollment data is compressed. Authentication data is not compressed.

>See>　For information on the enrollment format of palm vein data, refer to the “System Development Guide” and “3.2.2.15　JAVA_PvAPI_SetProfile [JAVA_uint32 Type setting value]”.

3.2.2.9 JAVA_BioAPI_VerifyMatch

[Function]

This method performs verification between captured authentication data and enrollment data of user identification information such as ID, and returns authentication result.

[Coding syntax]

```
public long JAVA_BioAPI_VerifyMatch
 (
   JAVA_uint32                  ModuleHandle,
   JAVA_sint32                  MaxFARRequested,
   JAVA_sint32                  MaxFRRRequested,
   JAVA_uint32                  FARPrecedence,
   JAVA_BioAPI_INPUT_BIR        ProcessedBIR,
   JAVA_BioAPI_INPUT_BIR        StoredTemplate,
   JAVA_sint32                  AdaptedBIR,
   JAVA_uint32                  Result,
   JAVA_sint32                  FARAchieved,
   JAVA_sint32                  FRRAchieved,
   JAVA_BioAPI_DATA             Payload
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of attached module.
MaxFARRequested	input	Unused
MaxFRRRequested	input/ optional	Create an instance of “JAVA_sint32” class and set matching level to the field “value” of the instance. It is possible to specify the following values. ・JAVA_PvAPI_MATCHING_LEVEL_HIGHEST (Highest) ・JAVA_PvAPI_MATCHING_LEVEL_HIGH (High) ・JAVA_PvAPI_MATCHING_LEVEL_NORMAL (Normal) ・JAVA_PvAPI_MATCHING_LEVEL_LOW (Low) ・JAVA_PvAPI_MATCHING_LEVEL_LOWEST (Lowest) Then specify the instance to this parameter.
FARPrecedence	input	Create an instance of “JAVA_uint32” class and set “JAVA_BioAPI_FALSE” to the field “value” of the instance. Then specify the instance to this parameter.

Name	I/O	Description
ProcessedBIR	input	Create an instance of “JAVA_BioAPI_INPUT_BIR” class and set authentication data to field “BIR” of the instance. Then, specify the instance to this parameter. Set authentication data as instance of BIR class. In order to specify the instance, and set values to other fields, refer to the following. >See> For information on the “JAVA_BioAPI_INPUT_BIR” class, refer to the “Appendix A.1 JAVA_BioAPI_INPUT_BIR Class”.
StoredTemplate	input	Create an instance of “JAVA_BioAPI_INPUT_BIR” class and set enrollment data to field “BIR” of the instance. Then specify the instance to this parameter. Set enrollment data as instance of BIR class. In order to specify the instance, and set values to other fields, refer to the following. >See> For information on the “JAVA_BioAPI_INPUT_BIR” class, refer to the “Appendix A.1 JAVA_BioAPI_INPUT_BIR Class”.
AdaptedBIR	output/ optional	Unused
Result	input/ output	Create and specify the instance of “JAVA_uint32” class. Authentication result is set to the field “value” of the instance. ・JAVA_BioAPI_TRUE　（Authentication OK） ・JAVA_BioAPI_FALSE　（Authentication NG）

Name	I/O	Description
FARAchieved	input/ output	Create and specify the instance of "JAVA_sint32" class. The following value is set according to the setting of "authentication result score notification function". ● Use "authentication result score notification function" <When authentication is OK> Score value against enrollment data is set to field "value" of the instance in a range from 1,000 to 10,000, by 1,000 units. The larger the value, the more the similarity. <When authentication is NG> "0" is set to field "value" of the instance. ● Not use "authentication result score notification function" Please do not use the value set in field "value" of the instance. >See> For information on "authentication result score notification function", refer to the "Authentication Library Reference Guide" or "3.2.2.15 JAVA_PvAPI_SetProfile [JAVA_uint32 Type setting value]".
FRRAchieved	output/ optional	Unused
Payload	output/ optional	Unused

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls "BioAPI_VerifyMatch" in Authentication library.

[Remark 1]

Calling this method, "JAVA_BioAPI_GUI_STATE_CALLBACK" event handler is called by Authentication library. The Authentication library sets guidance for starting the verification and so on to the parameter and calls event handler.

>See> For information on the "JAVA_BioAPI_GUI_STATE_CALLBACK", refer to the "3.2.2.24 JAVA_BioAPI_GUI_STATE_CALLBACK".

[Remark 2]

Call this method twice if it is necessary to authenticate using both hands of one user for enrollment data.

3.2.2.10 JAVA_BioAPI_IdentifyMatch

[Function]

This method performs identification against entire enrollment data to search similar data with captured authentication data, and returns similar data items as candidates.

!Caution **Authentication with identification**

Identification has higher risk of false acceptance compared to verification.
Therefore, it is necessary to consider measures against false acceptance when designing application.

>See> As for the measures against false acceptance of authentication with identification mode, refer to the “System Development Guide”.

[Coding syntax]

```
public long JAVA_BioAPI_IdentifyMatch
 (
   JAVA_uint32                      ModuleHandle,
   JAVA_sint32                      MaxFARRequested,
   JAVA_sint32                      MaxFRRRequested,
   JAVA_uint32                      FARPrecedence,
   JAVA_BioAPI_INPUT_BIR            ProcessedBIR,
   JAVA_BioAPI_IDENTIFY_POPULATION  Population,
   JAVA_uint32                      Binning,
   JAVA_uint32                      MaxNumberOfResults,
   JAVA_uint32                      NumberOfResults,
   JAVA_BioAPI_CANDIDATE[]          Candidates,
   JAVA_sint32                      Timeout
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of attached module.
MaxFARRequested	input	Unused

Name	I/O	Description
MaxFRRRequested	input/ optional	Create an instance of “JAVA_sint32” class and set matching level to field “value” of the instance. It is possible to specify the following values. ・JAVA_PvAPI_MATCHING_LEVEL_HIGHEST (Highest) ・JAVA_PvAPI_MATCHING_LEVEL_HIGH (High) ・JAVA_PvAPI_MATCHING_LEVEL_NORMAL (Normal) ・JAVA_PvAPI_MATCHING_LEVEL_LOW (Low) ・JAVA_PvAPI_MATCHING_LEVEL_LOWEST (Lowest) Then specify the instance to this parameter.
FARPrecedence	input	Create an instance of “JAVA_uint32” class and set “JAVA_BioAPI_FALSE” to field “value” of the instance. Then specify the instance to this parameter.
ProcessedBIR	input	Create an instance of “JAVA_BioAPI_INPUT_BIR” class and set authentication data to field “BIR” of the instance. Then specify the instance to this parameter. Set authentication data as the instance of BIR class. In order to specify the instance, and set values to other fields, refer to the following. See For information on the “JAVA_BioAPI_INPUT_BIR” class, refer to the “Appendix A.1 JAVA_BioAPI_INPUT_BIR Class”.
Population	input	Create and specify an instance of “JAVA_BioAPI_IDENTIFY_POPULATION” class. In order to specify the instance, refer to the following. See For information on the “JAVA_BioAPI_IDENTIFY_POPULATION” class, refer to the “Appendix A.2 JAVA_BioAPI_IDENTIFY_POPULATION Class”.
Binning	input	Create an instance of “JAVA_uint32” class and specify the “JAVA_BioAPI_FALSE” to field “BIR” of the instance. Then specify the instance to this parameter.
MaxNumberOfResults	input	Create an instance of “JAVA_uint32” class and specify the maximum number of enrollment data item (candidates) set in “NumberOfResults” and “Candidates” to field “value” of the instance. It can be specified in range from 1 to 30. Then specify the instance to this parameter.
NumberOfResults	input/ output	Create and specify the instance of “JAVA_uint32” class. The number of enrollment data items similar to authentication data (candidates) is set to field “value” of the instance.

Name	I/O	Description
Candidates	input/ output	Create and specify the array of “JAVA_BioAPI_CANDIDATE” class type. Set the size of array to be greater than specified value of “MaxNumberOfResults”. Interface module creates the instance of “JAVA_BioAPI_CANDIDATE” class, and sets information of similar enrollment data (candidates) to the instance. Also, the Interface module sets instances in order of decreasing similarity level from index 0 of array. As for detailed information set in “JAVA_BioAPI_ CANDIDATE”, refer to the following. >See> For information on the “JAVA_BioAPI_ CANDIDATE” class, refer to the “Appendix A.3 JAVA_BioAPI_CANDIDATE Class”. >See> As for the “authentication result score notification function”, refer to the “Authentication Library Reference Guide” and “3.2.2.15　JAVA_PvAPI_SetProfile [JAVA_uint32 Type setting value]”.
Timeout	input	Create an instance of “JAVA_sint32” class and set timeout duration in millisecond to field “value” of the instance. However, specify “0” (no timeout) at present. Then specify the instance to this parameter.

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “BioAPI_IdentifyMatch” in Authentication library.

[Caution 1]

This method can be called only when palm vein data is “Non-compressed format”.

>See> For information on enrollment format of palm vein data, refer to the “ System Development Guide” and “3.2.2.15　JAVA_PvAPI_SetProfile [JAVA_uint32 Type setting value]”.

[Caution 2]

Processing time for identification is slightly longer if “Image compression function” is used.

>See> For information on “Image compression function”, refer to the “System Development Guide”.

[Remark 1]

Calling this method, “JAVA_BioAPI_GUI_STATE_CALLBACK” event handler is called by Authentication library. The Authentication library sets “identification starting guidance” and so on to the parameter and calls event handler.

>See> For information on the “JAVA_BioAPI_GUI_STATE_CALLBACK”, refer to the “3.2.2.24 JAVA_BioAPI_GUI_STATE_CALLBACK”.

[Remark 2]

Use “JAVA_PvAPI_PresetIdentifyPopulation” when it is necessary to set a group of identification data to Authentication library in advance.

>See> For information on the “JAVA_PvAPI_PresetIdentifyPopulation”, refer to the “3.2.2.21 JAVA_PvAPI_PresetIdentifyPopulation”.

3.2.2.11 JAVA_BioAPI_Enroll

[Function]

This method captures palm vein on one hand and creates enrollment data.

>See> For information on enrollment data, refer to the “Authentication Library Reference Guide”.

[Coding syntax]

```
public long JAVA_BioAPI_Enroll
 (
   JAVA_uint32                      ModuleHandle,
   JAVA_uint8                       Purpose,
   JAVA_BioAPI_INPUT_BIR            StoredTemplate,
   JAVA_sint32                      NewTemplate,
   JAVA_BioAPI_DATA                 Payload,
   JAVA_sint32                      Timeout,
   JAVA_sint32                      AuditData
) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of attached module.
Purpose	input	Create an instance of “JAVA_uint8” class and specify the setting value for verification （Fixed to JAVA_BioAPI_PURPOSE_VERIFY） to field “value” of the instance. Then specify the instance to this parameter.
StoredTemplate	input/ optional	Unused
NewTemplate	output/ optional	Create and specify the instance of “JAVA_sint32” class. BIR handle of enrollment data is set to the instance.
Payload	input/ optional	Unused
Timeout	input	Create an instance of “JAVA_sint32” class and set timeout duration in millisecond to field “value” of the instance. However, specify “0” (no timeout) at present. Then specify the instance to this parameter.
AuditData	output/ optional	Unused

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “BioAPI_Enroll” in Authentication library.

[Caution]

Release the BIR handle acquired using this method by calling “JAVA_BioAPI_FreeBIRHandle”.

[Remark]

Calling this method, “JAVA_BioAPI_GUI_STATE_CALLBACK” event handler is called by Authentication library. The Authentication library sets guidance for “guiding the palm to correct position” and so on to the parameter, and calls event handler.

In case “JAVA_BioAPI_GUI_STREAMING_CALLBACK_IF” event listener is set to Interface module, “JAVA_BioAPI_GUI_STREAMING_CALLBACK” event handler is called by the Authentication library. Authentication library sets the “guidance image for guiding palm to correct position” to the parameter and calls event handler.

In order to use the “enrollment data score notification function”, Authentication library sets quality of enrollment data as a score value to the parameter, and calls event handler.

>See> For information on the “JAVA_BioAPI_SetGUICallbacks”, refer to the ”3.2.2.7 JAVA_BioAPI_SetGUICallbacks”.

>See> For information on the “JAVA_BioAPI_GUI_STATE_CALLBACK”, refer to the ”3.2.2.24 JAVA_BioAPI_GUI_STATE_CALLBACK”.

>See> For information on the “JAVA_BioAPI_GUI_STREAMING_CALLBACK”, refer to the “3.2.2.25 JAVA_BioAPI_GUI_STREAMING_CALLBACK”.

>See> As for the “enrolled data score notification function”, refer to the “Authentication Library Reference Guide”.

3.2.2.12 JAVA_BioAPI_Verify

[Function]

This method captures palm vein and performs verification between captured data and enrollment data of user identification information such as ID, then returns authentication result.

[Coding syntax]

```
public long JAVA_BioAPI_Verify
 (
   JAVA_uint32                      ModuleHandle,
   JAVA_sint32                      MaxFARRequested,
   JAVA_sint32                      MaxFRRRequested,
   JAVA_uint32                      FARPrecedence,
   JAVA_BioAPI_INPUT_BIR            StoredTemplate,
   JAVA_sint32                      AdaptedBIR,
   JAVA_uint32                      Result,
   JAVA_sint32                      FARAchieved,
   JAVA_sint32                      FRRAchieved,
   JAVA_BioAPI_DATA                 Payload,
   JAVA_sint32                      Timeout,
   JAVA_sint32                      AuditData
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of attached module.
MaxFARRequested	input	Unused
MaxFRRRequested	input/ optional	Create an instance of "JAVA_sint32" class and specify matching level to field "value" of the instance. It is possible to specify the following values. ・JAVA_PvAPI_MATCHING_LEVEL_HIGHEST (Highest) ・JAVA_PvAPI_MATCHING_LEVEL_HIGH (High) ・JAVA_PvAPI_MATCHING_LEVEL_NORMAL (Normal) ・JAVA_PvAPI_MATCHING_LEVEL_LOW (Low) ・JAVA_PvAPI_MATCHING_LEVEL_LOWEST (Lowest) Then specify the instance to this parameter.

Name	I/O	Description
FARPrecedence	input	Create an instance of “JAVA_uint32” class and set “JAVA_BioAPI_FALSE” to field “value” of the instance. Then specify the instance to this parameter.
StoredTemplate	input	Create an instance of “JAVA_BioAPI_INPUT_BIR” class and set enrollment data to field “BIR” of the instance. Then specify the instance to this parameter. Set the enrollment data as instance of BIR class. In order to specify the instance, and set values to other fields, refer to the following. >See> For information on the “JAVA_BioAPI_INPUT_BIR” class, refer to the “Appendix A.1 JAVA_BioAPI_INPUT_BIR Class”.
AdaptedBIR	output/ optional	Unused
Result	input/ output	Create and specify the instance of “JAVA_uint32” class. Authentication result is set to the field “value” of the instance. ・JAVA_BioAPI_TRUE (Authentication OK) ・JAVA_BioAPI_FALSE (Authentication NG)
FARAchieved	input/ output	Create and specify the instance of “JAVA_sint32” class. The following value is set according to the setting of “authentication result score notification function”. ● Use “authentication result score notification function” <When authentication is OK> Score value against enrollment data is set to field “value” of the instance in a range from 1,000 to 10,000, by 1,000 units. The larger the value, the more the similarity. <When authentication is NG> “0” is set to field “value” of the instance. ● Not use “authentication result score notification function” Please do not use the value set in field “value” of the instance. >See> For information on the “authentication result score notification function”, refer to the “Authentication Library Reference Guide” or “3.2.2.15 JAVA_PvAPI_SetProfile [JAVA_uint32 Type setting value]”.
FRRAchieved	output/ optional	Unused
Payload	output/ optional	Unused

Name	I/O	Description
Timeout	input	Create an instance of “JAVA_sint32” class and set timeout duration in millisecond to field “value” of the instance. However, specify “0” (no timeout) at present. Then specify the instance to this parameter.
AuditData	output/ optional	Unused

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “BioAPI_Verify” in Authentication library.

[Caution]

Processing time for verification is slightly longer if “Image compression function” is used.

>See> As for the “Image compression function”, refer to the “System Development Guide”.

[Remark 1]

Calling this method, “JAVA_BioAPI_GUI_STATE_CALLBACK” event handler is called by Authentication library. The Authentication library sets guidance for guiding the palm to correct position and so on to the parameter, and calls the event handler.

In case “JAVA_BioAPI_GUI_STREAMING_CALLBACK_IF” event listener is set to Interface module, “JAVA_BioAPI_GUI_STREAMING_CALLBACK” event handler is called by the Authentication library. The Authentication library sets the “guidance image for guiding palm to correct position” to the parameter and calls the event handler.

>See> For information on the “JAVA_BioAPI_SetGUICallbacks”, refer to the ”3.2.2.7　JAVA_BioAPI_SetGUICallbacks”.

>See> For information on the “JAVA_BioAPI_GUI_STATE_CALLBACK”, refer to the ”3.2.2.24　JAVA_BioAPI_GUI_STATE_CALLBACK”.

>See> For information on the “JAVA_BioAPI_GUI_STREAMING_CALLBACK”, refer to the “3.2.2.25　JAVA_BioAPI_GUI_STREAMING_CALLBACK”.

[Remark 2]

In order to authenticate one user using both hands of enrollment data, capture authentication data using “JAVA_BioAPI_Capture” and then call “JAVA_BioAPI_VerifyMatch” twice.

>See> For information on the “JAVA_BioAPI_Capture”, refer to the ”3.2.2.8 JAVA_BioAPI_Capture”.

>See> For information on the “JAVA_BioAPI_VerifyMatch, refer to the ”3.2.2.9 JAVA_BioAPI_VerifyMatch”.

3.2.2.13 JAVA_BioAPI_Identify

[Function]

This method captures palm vein and performs identification against entire enrollment data to search similar data with captured data, and returns similar data items as candidates.

!Caution **Authentication with identification**

Identification has higher risk of false acceptance compared to verification.
Therefore, it is necessary to consider measures against false acceptance when designing application.

>See> For information on measures against false acceptance of authentication with identification mode, refer to the “System Development Guide”.

[Coding syntax]

```
public long JAVA_BioAPI_Identify
 (
   JAVA_uint32                        ModuleHandle,
   JAVA_sint32                        MaxFARRequested,
   JAVA_sint32                        MaxFRRRequested,
   JAVA_uint32                        FARPrecedence,
   JAVA_BioAPI_IDENTIFY_POPULATION    Population,
   JAVA_uint32                        Binning,
   JAVA_uint32                        MaxNumberOfResults,
   JAVA_uint32                        NumberOfResults,
   JAVA_BioAPI_CANDIDATE[]            Candidates,
   JAVA_sint32                        Timeout,
   JAVA_sint32                        AuditData
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of attached module.
MaxFARRequested	input	Unused

Name	I/O	Description
MaxFRRRequested	input/ optional	Create an instance of “JAVA_sint32” class and specify matching level to field “value” of the instance. It is possible to specify the following values. ・JAVA_PvAPI_MATCHING_LEVEL_HIGHEST (Highest) ・JAVA_PvAPI_MATCHING_LEVEL_HIGH (High) ・JAVA_PvAPI_MATCHING_LEVEL_NORMAL (Normal) ・JAVA_PvAPI_MATCHING_LEVEL_LOW (Low) ・JAVA_PvAPI_MATCHING_LEVEL_LOWEST (Lowest) Then specify the instance to this parameter.
FARPrecedence	input	Create an instance of “JAVA_uint32” class and set “JAVA_BioAPI_FALSE” to field “value” of the instance. Then specify the instance to this parameter.
Population	input	Create and specify the instance of “JAVA_BioAPI_IDENTIFY_POPULATION” class. In order to specify the instance, refer to the following. >See> For information on the “JAVA_BioAPI_ IDENTIFY_POPULATION” class, refer to the “Appendix A.2　JAVA_BioAPI_IDENTIFY_ POPULATION Class”.
Binning	input	Create an instance of “JAVA_uint32” class and specify the “JAVA_BioAPI_FALSE” to field “BIR” of the instance. Then specify the instance to this parameter.
MaxNumberOfResults	input	Create an instance of “JAVA_uint32” class and specify the maximum number of enrollment data item (candidates) set in “NumberOfResults” and “Candidates” to field “value” of the instance. It can be specified in range from 1 to 30. Then specify the instance to this parameter.
NumberOfResults	input/ output	Create and specify the instance of “JAVA_uint32” class. The number of enrollment data items similar to authentication data item is set to field “value” of the instance.

Name	I/O	Description
Candidates	input/ output	Create and specify the “JAVA_BioAPI_CANDIDATE” class type array. Set the size of array to be greater than specified value of “MaxNumberOfResults”. Interface module creates instance of “JAVA_BioAPI_CANDIDATE” class, and sets information of similar enrollment data (candidates) to the instance. Also, Interface module sets instances in order of decreasing similarity level from index 0 of array. As for detailed information set in “JAVA_BioAPI_ CANDIDATE”, refer to the following. >See> For information on the “JAVA_BioAPI_ CANDIDATE” class, refer to the “Appendix A.3 JAVA_BioAPI_CANDIDATE Class”. >See> As for the authentication result score notification function, refer to the “Authentication Library Reference Guide” and “3.2.2.15 JAVA_PvAPI_SetProfile [JAVA_uint32 Type setting value]”.
Timeout	input	Create an instance of “JAVA_sint32” class and set timeout duration in millisecond to field “value” of the instance. However, specify “0” (no timeout) at present. Then specify the instance to this parameter.
AuditData	output/ optional	Unused

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “BioAPI_Identify” in Authentication library.

[Caution 1]

This method can be called when palm vein data is “Non-compressed format”.

>See> For information on enrollment format of palm vein data, refer to the “System Development Guide” and “3.2.2.15 JAVA_PvAPI_SetProfile [JAVA_uint32 Type setting value]”.

[Caution 2]

Processing time for identification is slightly longer if the “Image compression function” is used.

>See>　For information on “Image compression function”, refer to the “System Development Guide”.

[Remark 1]

Calling this method, “JAVA_BioAPI_GUI_STATE_CALLBACK” event handler is called by Authentication library. Authentication library sets guidance for guiding the palm to correct position and so on to the parameter, and calls the event handler.

In case “JAVA_BioAPI_GUI_STREAMING_CALLBACK_IF” event listener is set to Interface module, “JAVA_BioAPI_GUI_STREAMING_CALLBACK” event handler is called by the Authentication library. The Authentication library sets the “guidance image for guiding palm to correct position” to the parameter and calls the event handler.

>See>　For information on the “JAVA_BioAPI_SetGUICallbacks”, refer to the “3.2.2.7　JAVA_BioAPI_SetGUICallbacks”.

>See>　For information on the “JAVA_BioAPI_GUI_STATE_CALLBACK”, refer to the “3.2.2.24　JAVA_BioAPI_GUI_STATE_CALLBACK”.

>See>　For information on the “JAVA_BioAPI_GUI_STREAMING_CALLBACK”, refer to the “3.2.2.25　JAVA_BioAPI_GUI_STREAMING_CALLBACK”.

[Remark 2]

Use “JAVA_PvAPI_PresetIdentifyPopulation” when it is necessary to deliver identification data group to Authentication library in advance.

>See>　For information on the “JAVA_PvAPI_PresetIdentifyPopulation”, refer to the “3.2.2.21　JAVA_PvAPI_PresetIdentifyPopulation”.

3.2.2.14 JAVA_PvAPI_ApAuthenticate

[Function]

This method authenticates application using specified key.

[Coding syntax]

```
public long JAVA_PvAPI_ApAuthenticate
 (
   String                      Key
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
Key	input	Specify application key (Note) described in “License agreement”. Note) Please specify the same application key used to acquire license file.

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “PvAPI_ApAuthenticate” in Authentication library.

[Caution]

Do not call following methods until process of application authentication terminates normally.

[Remark]

In addition to application, this method also authenticates license.

3.2.2.15 JAVA_PvAPI_SetProfile [JAVA_uint32 Type setting value]

[Function]

This method sets operation mode of Authentication library.

[Coding syntax]

```
public long JAVA_PvAPI_SetProfile
 (
   JAVA_uint32                       ModuleHandle,
   JAVA_uint32                       dwFlag,
   JAVA_uint32                       dwParam1,
   JAVA_uint32                       dwParam2,
   JAVA_uint32                       dwReserve
) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of attached module.
dwFlag	input	Create an instance of "JAVA_uint32" class and set flag for operation mode to field "value" of the instance. Then specify the instance to this parameter. >See> As for the flag for operation mode, refer to the "Flag" of [Definition].
dwParam1	input	Create an instance of "JAVA_uint32" class and set value for operation mode to field "value" of the instance. Then specify the instance to this parameter. >See> As for the value for operation mode, refer to the "Setting value" of [Definition].
dwParam2	input	Unused
dwReserve	input	Unused

[Definition]

[Flag and setting value]

Setting Item	Flag	Setting value
Enrollment format of palm vein data (Professional Edition only)	JAVA_PvAPI _PROFILE _REGIST_DATA_TYPE	Specify enrollment format of palm vein data. It is possible to specify the following values. JAVA_PvAPI_PROFILE_REGIST_DATA_TYPE_NO_COMPRESS_2 ：Non-compressed format (default) JAVA_PvAPI_PROFILE_REGIST_DATA_TYPE_COMPRESS_2 ：Compressed format (Note1) (Note 2) Note 1) The following functions for identification are not available when “Compressed format” is specified. ・JAVA_BioAPI_IdentifyMatch ・JAVA_BioAPI_Identify ・JAVA_PvAPI_PresetIdentifyPopulation Neither the following function is available. ・“Authentication result score notification function” Note 2) Only enrollment data can be compressed when compressed format is specified for enrollment format. Authentication data is not compressed. >See> As for the enrollment format of palm vein data, refer to the “System Development Guide”.
Encryption method of palm vein data	JAVA_PvAPI _PROFILE _CR_INFO	Specify encryption method of enrollment/authentication data. It is possible to specify the following values. JAVA_PvAPI_PROFILE_CR_KIND_2 ：AES128 (Default) JAVA_PvAPI_PROFILE_CR_KIND_3 ：AES256 >See> As for the encryption method of palm vein data, refer to the “System Development Guide”.

Setting Item	Flag	Setting value
Guide mode (Professional Edition only)	JAVA_PvAPI _PROFILE _GUIDE_MODE	Specify guide mode for capturing palm vein data. It is possible to specify the following values. JAVA_PvAPI_PROFILE_GUIDE_MODE_GUIDE ：With guide mode JAVA_PvAPI_PROFILE_GUIDE_MODE_NO_GUIDE : Without guide mode (Default) (Note) Note) Be sure to specify “without guide mode” for the following case. ・ Connect mouse type Sensor (sold separately) >See> As for the guide mode, refer to the “System Development Guide”. There are several issues should be considered for the usage of the guide mode. >See> For consideration of guide mode usage, refer to the“Authentication Library Reference Guide”.
Capturing angle (Professional Edition only)	JAVA_PvAPI _PROFILE _SENSOR_DIRECTION	Specify capturing angle of palm when placing a hand on sensor to capture palm vein. It is possible to specify the following values. JAVA_PvAPI_PROFILE_SENSOR_DIRECTION_0 : 0° (Default) (Note) JAVA_PvAPI_PROFILE_SENSOR_DIRECTION_90 : 90° JAVA_PvAPI_PROFILE_SENSOR_DIRECTION_180 : 180° JAVA_PvAPI_PROFILE_SENSOR_DIRECTION_270 : 270° Note) Be sure to set “0°” for the following case. ・ Specify “ without guide mode” to guide mode >See> For capturing angle, refer to the “System Development Guide”.

Setting Item	Flag	Setting value
Authentication result score notification function	JAVA_PvAPI _PROFILE _SCORE _NOTIFICATIONS	Specify whether to use "authentication result score notification function". It is possible to specify the following values. JAVA_PvAPI_PROFILE_SCORE_NOTIFICATIONS _OFF　: Do not use (Default) JAVA_PvAPI_PROFILE_SCORE_NOTIFICATIONS _ON　: Use (Note) Note) Palm vein data must satisfy the following condition when "Use" is specified. ・Enrollment format of palm vein data is "Non-compressed format". >See>　As for the "authentication result score notification function", refer to the "System Development Guide". Processing time for verification is slightly longer if this function is used in verification process.

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls "PvAPI_SetProfile" in Authentication library.

[Caution]

It is necessary to call this method after calling "JAVA_BioAPI_ModuleAttach" and before calling each method for capturing, enrollment, verification and identification.

>See>　For information on the "JAVA_BioAPI_ModuleAttach", refer to the "3.2.2.3　JAVA_BioAPI_ModuleAttach".

[Remark 1]

It is necessary to call this method for each setting item. If same item is set multiple times, final value becomes effective.

[Remark 2]

The following items specified in this method can also be specified in setting file of Authentication library.

・Guide mode

・"Authentication result score notification function"

In case these items are specified in the setting file and in this method too, value of this method becomes effective.

>See> For information on setting file of Authentication library, refer to the “Authentication Library Reference Guide”.

3.2.2.16 JAVA_PvAPI_SetProfile [StringType setting value]

[Function]

This method sets operation mode of Authentication library.

[Coding syntax]

```
public long JAVA_PvAPI_SetProfile
 (
   JAVA_uint32                     ModuleHandle,
   JAVA_uint32                     dwFlag,
   String                          dwParam1,
   JAVA_uint32                     dwParam2,
   JAVA_uint32                     dwReserve
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of attached module.
dwFlag	input	Create an instance of "JAVA_uint32" class and set flag for operation mode to field "value" of the instance. Then specify the instance to this parameter. >See> As for the flag for operation mode, refer to the "Flag" of [Definition].
dwParam1	input	Create an instance of String class and specify it. >See> As for the value for operation mode, refer to the "Setting value" of [Definition].
dwParam2	input	Unused
dwReserve	input	Unused

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls "PvAPI_SetProfile" in Authentication library.

[Definition]

Setting Item	Flag	Setting value
Individual encryption key	JAVA_PvAPI _PROFILE _CARD_INFO	For creation of enrollment data, specify individual encryption key in 16 bytes string (Note) if different encryption key is necessary to encrypt enrollment data. (For example, use individual encryption key in order to encrypt enrollment data using unique encryption key for each smart card.) Note) If the string is longer than 16 bytes when converted to ASCII language, only first 16 bytes becomes effective. For authentication, it is necessary to specify the same individual encryption key as enrollment in order to decrypt encrypted enrollment data. >See> For information on basic sequence for using individual encryption key, refer to the "Authentication Library Reference Guide".

3.2.2.17 JAVA_PvAPI_GetErrorInfo

[Function]

This method acquires error information.

[Coding syntax]

```
public long JAVA_PvAPI_GetErrorInfo
 (
   JAVA_PvAPI_ErrorInfo           ErrorInfo
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ErrorInfo	input/ output	Create and specify the instance of “JAVA_PvAPI_ErrorInfo” class. Error information is set to the instance.

[Description]

[Error information class]

```
public class JAVA_PvAPI_ErrorInfo
{
    public long    ErrorLevel;     Error level (0: Normal, 1: Cancellation,
                                               2: Recoverable, 3: Not recoverable)
    public long    ErrorCode;      Error type  (1: Device, 2: Resource,
                                               3: Application, 4: Others)
    public long    ErrorDetail;    Error details
    public long    ErrorModule;    Error detected module(1: Interface section,
                                                        2: Authentication library)
    public long    ErrorOptional1; Internal information 1
    public long    ErrorOptional2; Internal information 2
    public long[]  APIInfo;        Interface section information
                                   Specified array of long type(length is 4)
    public long    ErrorInfo1;     Error information 1
                                   notified in Authentication library
    public long    ErrorInfo2;     Error information 2
                                   notified in Authentication library
    public long[]  ErrorInfo3;     Error information 3
                                   notified in Authentication library
                                   Specified array of long type(length is 4)
}
```

★Tip　**“Internal information 1” – “Error information 3” notified in Authentication library**

Since these are internal information, it is not necessary to pay attention to the value at application.

>See>　As for error details, refer to the “Authentication Library Reference Guide".

[Description]

This method calls “PvAPI_GetErrorInfo” in Authentication library.

[Caution]

In order to acquire error information, call this method immediately after calling the method where error occurred. The content of error information is not guaranteed if this method is called at any other time.

3.2.2.18 JAVA_PvAPI_Sense

[Function]

This method detects if a hand is placed over sensor.

[Coding syntax]

```
public long JAVA_PvAPI_Sense
 (
   JAVA_uint32                        ModuleHandle,
   JAVA_uint32                        Timeout,
   JAVA_uint32                        Interval,
   JAVA_uint32                        CheckRetryInterval,
   JAVA_uint32                        CheckRetryCount
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of attached module.
Timeout	input	Create an instance of "JAVA_uint32" class and set timeout duration in millisecond to field "value" of the instance. Then specify the instance to this parameter. A range from 0 to 3,600,000 (Note1) can be specified. Recommended value is 10,000 or more. Specify 0 for not timing out. Note1) If specified value is under 10,000, this method may return with timeout before detecting a hand even if a hand is placed.
Interval	input	Create an instance of "JAVA_uint32" class and set "sensing interval before hand detections" in milliseconds to field "value" of the instance. (Note2) Then specify the instance to this parameter. A range from 50 to 1,000 can be specified. Recommended value is 100.
CheckRetryInterval	input	Create an instance of "JAVA_uint32" class and set "interval between first detection and re-detection of a hand" in milliseconds to field "value" of the instance. (Note2). A range from 50 to 1,000 can be specified. Recommended value is 50.

Name	I/O	Description
CheckRetryCount	input	Create an instance of “JAVA_uint32” class and set “number of times to perform re-detections after first detection of a hand” to field “value” of the instance. A range from 0 to 15 can be specified. Recommended value is 3. Sensor assumes “a hand is positioned” when it detects a hand for the number of specified times consecutively.

Note 2)

In case time of hand detection in Sensor exceeds specified detection or re-detection interval, next hand detection process starts after termination of hand detection in sensor.

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “PvAPI_Sense” in Authentication library.

The “PvAPI_Sense” performs the following processes for hand detection.

- While a hand is not detected, hand detection is performed at specified intervals.
- Once a hand is detected, re-detection is performed for specified interval.
- After a hand is detected for specified number of times consecutively, the method determines as “the hand is placed” and returns.

[Remark]

Use this method in order to call "JAVA_BioAPI_Identify" or "JAVA_BioAPI_Capture" at timing of placing a palm.

(It is not necessary to use this method when calling "JAVA_BioAPI_Identify" or "JAVA_BioAPI_Capture" at other timing such as pressing a button.)

3.2.2.19 JAVA_PvAPI_Cancel

[Function]

This method cancels a process.

[Coding syntax]

```
public long JAVA_PvAPI_Cancel
 (
   JAVA_uint32                     ModuleHandle,
   JAVA_PvAPI_ErrorInfo            ErrorInfo
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of attached module.
ErrorInfo	input/ output	Create and specify the instance of "JAVA_PvAPI_ ErrorInfo" class. Error information occurred in this method is set in the instance.

[Definition]

[Error information class]

>See>　Refer to the [Definition] of "3.2.2.17　JAVA_PvAPI_GetErrorInfo".

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls "PvAPI_Cancel" in Authentication library.

[Caution 1]

The process of target method is canceled when its status is ready for cancellation, and then the method terminates with error. In such case, do not use any output data even if it contains information because the contents are not guaranteed.

[Caution 2]

The following errors are notified if this method is called again during cancellation process.

- Cancellation not possible
- Sequence error

[Remark]

The following methods are target for cancellation.

- JAVA_BioAPI_Capture
- JAVA_BioAPI_VerifyMatch
- JAVA_BioAPI_IdentifyMatch
- JAVA_BioAPI_Enroll
- JAVA_BioAPI_Verify
- JAVA_BioAPI_Identify
- JAVA_PvAPI_Sense
- JAVA_PvAPI_PresetIdentifyPopulation

3.2.2.20 JAVA_PvAPI_PreSetProfile

[Function]

This method sets sensor information of multiple Sensor connection and information of multiple processing for identification, which are necessary to be set before module attachment.

[Coding syntax]

```
public long JAVA_PvAPI_PreSetProfile
 (
   JAVA_uint32                        uiFlag,
   Object                             lpvParamData,
   JAVA_uint32                        uiParamDataSize,
   Object                             lpvReserve
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
uiFlag	input	Create an instance of "JAVA_uint32" class and set flag to field "value" of the instance. Then specify the instance to this parameter. >See> As for the flag, refer to the "Flag" of [Definition].
lpvParamData	input	Specify the setting value. >See> In order to specify the setting value, refer to the "Setting value" of [Definition].
uiParamDataSize	input	Unused
lpvReserve	input	Unused

[Definition]

[Flag, setting value]

Name	Flag	Setting value
Sensor information	JAVA_PvAPI_PRE_PROFILE_IDENTIFYSENSOR	Create an instance of "JAVA_PvAPI_SensorInfo" class and set information of sensor to be connected to the instance. Then specify the instance to "lpvParamData". >See> In order to acquire sensor information, refer to the "3.2.2.22 JAVA_PvAPI_GetConnectSensorInfoEx".

Name	Flag	Setting value
Information of multiple processing for identification	JAVA_PvAPI_PRE_PROFILE_ MAX_MATCHTHREAD_NUM	Create an instance of “JAVA_uint32” class and set the following value to field “value” of the instance. 0 : Not use multiple processing (1 thread) 2-128 : Use multiple processing (2-128 threads) (Note) After the above setting, specify the instance to “lpvParamData”. Note) The number of specified threads has to be within the number of CPU core of operational environment. >See> As for the multiple processing for identification, refer to the “System Development Guide”.

[Sensor information class]

```
public class JAVA_PvAPI_SensorInfo
{
    public long    uiSerialNo;     Serial number
    public String  szUnitNo;       Model
    public long    uiSensor;       Reserved
    public Object  lpvReserve2;    Reserved
}
```

[Information to be set to Sensor information class]

Name	I/O	Description
uiSerialNo	input	Specify the serial number of connected sensor.
szUnitNo	input	Specify the model of connected sensor.
uiReserve1	input	Unused
lpvReserve2	input	Unused

★Tip **Serial number and model of Sensor**
Using Sensor maintenance tool, it is possible to confirm serial number and model of Sensor.

>See> For information on Sensor maintenance tool, refer to the “Sensor Maintenance Tool Operation Guide”.

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “PvAPI_PreSetProfile” in Authentication library.

[Caution]

As for multiple Sensor connection, please note the following.

- There are several notes to be considered when designing and developing application with multiple Sensor connection.

>See> As for the application design with multiple Sensors, refer to the "System Development Guide" and "Authentication Library Reference Guide".

- In order to connect multiple Sensors, it is necessary to call this method after calling "JAVA_BioAPI_ModuleLoad" and before calling "JAVA_ BioAPI_ ModuleAttach".

  Also, it is necessary to call this method again before calling "JAVA_BioAPI_ ModuleAttach" if "JAVA_BioAPI_ModuleDetach" or "JAVA_BioAPI_Module Unload" has been called, Subsequent operations are not guaranteed if this method is not called.
- Sensor information specified in this method is effective until "JAVA_BioAPI _ModuleDetach" is called.
- If Sensor information is set multiple times in this method, the latter setting becomes effective.
- Operations are not guaranteed if the same Sensor information is set in this method from multiple applications.
- Operations are not guaranteed if multiple Sensors are connected without setting Sensor information in this method.

3.2.2.21 JAVA_PvAPI_PresetIdentifyPopulation

[Function]

In order for identification, this method sets target palm vein data group to Authentication library in advance.

[Coding syntax]

```
public long JAVA_PvAPI_PresetIdentifyPopulation
 (
   JAVA_uint32                        ModuleHandle,
   JAVA_BioAPI_IDENTIFY_POPULATION    Population
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
ModuleHandle	input	Specify the handle of the attached module.
Population	input	Create an instance of "JAVA_BioAPI_IDENTIFY_ POPULATION" class and set enrollment data group to the instance. Then specify the instance to this parameter. As for the number of the enrollment data group, please keep it within 1,000 hands. In order to specify the data, refer to the following. >See> For information on the "JAVA_BioAPI_IDENTIFY _POPULATION" class, refer to the "Appendix A.2 JAVA_BioAPI_IDENTIFY_POPULATION Class".

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls "PvAPI_PresetIdentifyPopulation" in Authentication library.

[Caution 1]

This method can be called when enrollment format of palm vein data is "Non-compressed format".

>See> For information on enrollment format of palm vein data, refer to the " System Development Guide" and "3.2.2.15 JAVA_PvAPI_SetProfile [JAVA_uint32 Type setting value]".

[Caution 2]

It is necessary to call this method after calling "JAVA_BioAPI_ModuleAttach" and before calling "JAVA_BioAPI_ModuleDetach".
Palm vein data group enrolled in advance is effective until calling "JAVA_BioAPI_ModuleDetach".

[Caution 3]

Calling this method, enrollment data is preserved on memory as decrypted data for a long period. Therefore, call this method in secure environment taken security measures.

[Remark]

Using this method, it is possible to reduce processing time for identification.
This method is effective in case that the number of enrollment data is large and those data is fixed while identification. On the other hand, it is difficult to use this function in case enrollment data is updated frequently.

3.2.2.22 JAVA_PvAPI_GetConnectSensorInfoEx

[Function]

This method searches Sensors connected to target hardware to acquire Sensor information, and returns the information as a list.

Use this method in order to acquire setting information to switch Sensors when multiple Sensors are connected to the target hardware.

>See> For information on how to switch Sensors, refer to the “5.9.1 Sensor Switching Sequence”.

[Coding syntax]

```
public long JAVA_PvAPI_GetConnectSensorInfoEx
 (
   JAVA_uint32                       lpuiSensorNum,
   JAVA_PvAPI_SensorInfoEx[]         lptSensorInfo
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
lpuiSensorNum	input/ output	Create and specify the instance of “JAVA_unit32” class. The number of connected sensors is set in the range of 0 - 8.
lptSensorInfo	input/ output	Create “JAVA_PvAPI_SensorInfoEx” class array and specify it. Set the size of array to be greater than “JAVA_PvAPI _GET_SENSOR_INFO_MAX”. Interface module creates instance of ”JAVA_PvAPI_SensorInfoEx” class and sets sensor information to the instance. Also, Interface module sets instance for the number of “lpuiSensorNum” from index 0 of array.

[Definition]

[Maximum number of returned Sensor information items]

```
JAVA_PvAPI_GET_SENSOR_INFO_MAX; Maximum number of returned items (8)
```

[Sensor information class（Note 1）

```
public class JAVA_PvAPI_SensorInfoEx
{
    public long    uiSerialNo;                Serial number
    public String  szUnitNo;                  Model
    public long    uiSensor;                  Sensor type (Note 2)
    public long    uiFwVersion;               Firmware version
    public long    uiFwLevel;                 Firmware level
    public Object  szReserve;                 Reserved
}
```

Note 1) Sensor information is not set when sensor is not connected.

Note 2) The following value is set for sensor type.

JAVA_PvAPI_INFO_SENSOR_TYPE_2 :	PalmSecure Sensor or PalmSecure Sensor V2

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “PvAPI_GetConnectSensorInfoEx” in Authentication library.

[Caution 1]

Call this method after calling “JAVA_BioAPI_ModuleLoad” and before calling “JAVA_BioAPI_ModuleAttach”.

[Caution 2]

Sensor information is not set in case of each of the following.

・Sensor driver corresponding to connected Sensor is not installed.

・Connected Sensor is used by other process.

[Caution 3]

As for the number of sensors, although it can be set up to 8 sensors in the interface of this method, please connect up to 2 sensors. Operations are not tested if 3 or more sensors are connected.

In case of connecting 3 or more sensors, be sure to test in actual environment in customer’s responsibility.

3.2.2.23 JAVA_PvAPI_GetLibraryInfo

[Function]

This method acquires file version of Authentication library.

It also acquires sensor information when Sensor is connected.

[Coding syntax]

```
public long JAVA_PvAPI_GetLibraryInfo
 (
   JAVA_PvAPI_LBINFO                lptLBInfo
 ) throws PalmSecureException
```

[Parameters]

Name	I/O	Description
lptLBInfo	input/ output	Create and specify the instance of "JAVA_PvAPI_LBINFO" class. Version information of Authentication library is specified to the instance.

[Definition]

[Version information class]

```
public class JAVA_PvAPI_LBINFO
{
   public long    uiLibVersion;     Authentication library version         (Note1)
   public long    uiLibLevel;       Authentication library level           (Note1)
   public long    uiLibSubCounter;  Authentication library version counter (Note1)
   public String  szDrvVersion;     Sensor driver component version
   public long    uiFwVersion;      Firmware version
   public long    uiFwLevel;        Firmware level
   public long    uiSensorKind;     Sensor type 1 (Note2)
   public long    uiSensorExtKind;  Sensor type 2 (Note3)
   public long    uiSerialNo;       Serial number
   public String  szUnitNo;         Model
   public long    uiLoopMode;     Whether “Continuous capure function” is used (Note4)
   public long    uiCompressMode; Whether “Image compression function” is used (Note5)
   public long    uiDriverKind;     Driver type (Note 6)
   public long    uiEdition;        Authentication library Edition type (Note 7)
   public byte[]  szReserve;        Reserved
}
```

Note 1) The notified parameters of Authentication library is file version shown as the version information of property of those files of "PvFw.DLL" or “F3BC4BIOSV.DLL”.

Please note the information is different from Authentication library version information (shown in “pvfwvl.txt” or “pvfwvlSV.txt”).

Note 2) The following value is set in “Sensor type 1”.

JAVA_PvAPI_INFO_SENSOR_TYPE_2:

PalmSecure Sensor or PalmSecure Sensor V2

Note 3) The following value is set in “Sensor type 2”.

JAVA_PvAPI_INFO_SENSOR_MODE_COMPATIBLE:

PalmSecure Sensor

JAVA_PvAPI_INFO_SENSOR_MODE_EXTEND:

PalmSecure Sensor V2

Note 4) The following value is set in

“Whether ‘Continuous capture function’ is used”.

JAVA_PvAPI_INFO_LOOP_MODE_OFF:	Do not use (capture once)
JAVA_PvAPI_INFO_LOOP_MODE_ON:	Use (capture 2 to 5 times)

Note 5) The following value is set in

“Whether ‘Image compression function’ is used”.

JAVA_PvAPI_INFO_COMPRESS_MODE_OFF:	Do not use
JAVA_PvAPI_INFO_COMPRESS_MODE_ON:	Use

Note 6) The following value is set in “Driver type”.

JAVA_PvAPI_INFO_DRIVER_KIND_EXTENDED:

Sensor driver for extended function

JAVA_PvAPI_INFO_DRIVER_KIND_CONVENTIONAL:

Conventional Sensor driver

Note 7) The following value is set in “Authentication library Edition type”.

JAVA_PvAPI_INFO_EDITION_PE:	Professional Edition
JAVA_PvAPI_INFO_EDITION_EE:	Enterprise Edition

[Return value]

JAVA_BioAPI_OK	Successful
JAVA_BioAPI_ERRCODE_FUNCTION_FAILED	Error

[Description]

This method calls “PvAPI_GetLibraryInfo” in Authentication library.

[Caution]

Call this method after calling “JAVA_BioAPI_ModuleAttach” and before calling “JAVA_BioAPI_ModuleDetach”.

Only “Authentication library file version” is notified if this method is called before calling “JAVA_BioAPI_ModuleAttach”, after calling “JAVA_BioAPI_ModuleDetach” or Sensor is not connected.

3.2.2.24 JAVA_BioAPI_GUI_STATE_CALLBACK

[Function]

"Guidance for correct palm positioning" and "silhouette image of captured palm" is notified from Authentication library. Also, the followings are notified depending on function.

- When you use "detailed information notification function for guidance image display":
  Notified detailed information to show the guidance image for correct palm positioning
- When you use "enrolled data score notification function":
  Notified the quality of enrollment data as score value.
- When you use "shutter function":
  Notified message for the start of capturing before the second capturing at enrollment of palm vein data. In addition, the silhouette image is notified after the second capturing.

>See> As for the "detailed information notification function", "enrolled score notification function" and "shutter function", refer to the "Authentication Library Reference Guide".

[Coding syntax]

```
public long JAVA_BioAPI_GUI_STATE_CALLBACK
 (
   Object                          GuiStateCallbackCtx,
   long                            GuiState,
   short                           Response,
   long                            Message,
   short                           Progress,
   JAVA_BioAPI_GUI_BITMAP          SampleBuffer
) ;
```

[Parameters]

Name	I/O	Description
GuiStateCallbackCtx	input	Notified the object specified in the parameter of "JAVA_BioAPI_SetGUICallbacks".
GuiState	input	Notified notification type. • Silhouette image notification • Guidance notification. • Guidance image notification.
Response	output	Unused
Message	input/ optional	Notified message ID when the notification type is "Guidance notification".

Name	I/O	Description
Progress	input/ optional	Unused
SampleBuffer	input/ optional	Notified silhouette image when notification type is "Silhouette image notification".

[Definition]

[Notification type] (Note 1)

JAVA_BioAPI_SAMPLE_AVAILABLE	(0x0001)	Silhouette image notification
JAVA_BioAPI_MESSAGE_PROVIDED	(0x0002)	Guidance notification
JAVA_PvAPI_APPEND_STREAMING	(0x0008)	Guidance image notification (Note 2)

Note 1) Notification type is specified by bits.

Note 2) The guidance image notification notifies the guidance image corresponding to message ID notified by guidance notification as callback notification. When bits in the guidance image notification (0x0008) are on, also the bits in guidance notification (0x0002) is on. Therefore, a value "0x0000000A" is notified.

>See> As for the callback notification for guidance image, refer to the "3.2.2.25　JAVA_BioAPI_GUI_STREAMING_CALLBACK" method.

[Silhouette image]

```
public class JAVA_BioAPI_GUI_BITMAP
{
  public long             Width;           Width of image
  public long             Height;          Height of image
  public JAVA_BioAPI_DATA  Bitmap;          Bitmap data
}
```

[Return value]

[When not using detailed information notification function for guidance image display]

JAVA_BioAPI_OK	Successful (fixed)

[When using detailed information notification function for guidance image display]

JAVA_BioAPI_OK	Return from the function(Note)
JAVA_PvAPI_WAIT	Repeat status notification callback (Note)

>See> As for the "detailed information notification function for guidance image display", refer to the "Authentication Library Reference Guide".

Note) "JAVA_BioAPI_OK" and "JAVA_PvAPI_WAIT" can be specified when notification type is "guidance notification" and process completion type "JAVA_PvAPI_NOTIFY_API_END" (0x030002XX) is notified in message ID.

Specify the "JAVA_PvAPI_WAIT" in order to prevent capturing/enrollment method from returning while guidance image is shown in application. However, be sure to specify the "JAVA_BioAPI_OK" if the value of process completion type "JAVA_PvAPI_NOTIFY_API_END" is "Canceled" (0x03000210) or "Error" (0x03000220).

[Description]

This method is called by "BioAPI_GUI_STATE_CALLBACK" in Authentication library.

[Caution 1]

The event handlers for status notification are notified in short intervals. For example, when a hand is not positioned over Sensor, a guidance message "Please place your hand above the Sensor." is notified continuously until a hand is positioned. In this case, application should take appropriate actions such as to skip same guidance messages.

In case application completely ignores the guidance notification, problems can be occurred such as user gets confused with hand positioning. Therefore, application is necessary to notify user of the latest guidance notification.

[Caution 2]

Perform minimum necessary processes in the event handler for status notification and return as soon as possible.

[Caution 3]

Do not modify control mode of floating point in the event handler for status notification. Operations are not guaranteed if control mode for the floating point is modified.

[Remark 1]

The event handler for status notification is defined in "JAVA_BioAPI_GUI_STATE_CALLBACK_IF" interface (event listener for status notification)". In order to use this event handler, the following is necessary in advance; create an instance of the class which implements the "JAVA_BioAPI_GUI_STATE_CALLBACK_IF" interface in application, and then set it to the parameter of "JAVA_BioAPI_SetGUICallbacks" method.

>See> For information on the "JAVA_BioAPI_SetGUICallbacks", refer to the "3.2.2.7 JAVA_BioAPI_SetGUICallbacks".

[Remark 2]

The following table shows message ID to be notified when notification type is “guidance notification”, and whether messages are notified by each guide mode.

✔ : Notified　　- : Not notified

Message ID		Description (1st row: Meaning and Status 2nd row: Message example)	Guide mode	
Value	Definition		With guide mode	Without guide mode
0x020300MN (Note 1) (Note 2)	JAVA_PvAPI_NOTIFY_CAP _GUID_START	Capturing guidance is started.	✔	✔
0x02030200	JAVA_PvAPI_NOTIFY_CAP _GUID_BADIMAGE	Capturing guidance is in progress: Hand is still on Sensor	✔	✔
		Please move your hand away from the Sensor.		
0x02030201	JAVA_PvAPI_NOTIFY_CAP _GUID_NO_HANDS	Capturing guidance is in progress: Hand is not placed.	✔	✔
		Place your hand.		
0x02030203	JAVA_PvAPI_NOTIFY_CAP _GUID_MOVING	Capturing guidance is in progress: Hand is still moving.	✔	✔
		Don’t move your hand.		
0x02030204	JAVA_PvAPI_NOTIFY_CAP _GUID_LESSINFO	Capturing guidance is in progress: Shape of hand is not recognized.	✔	✔
		Spread your fingers, and place your hand to the correct position again.		
0x02030205	JAVA_PvAPI_NOTIFY_CAP _GUID_RIGHT	Capturing guidance is in progress: Hand is placed too far to the right.	-	✔
		Move your hand slightly to the left.		
0x02030206	JAVA_PvAPI_NOTIFY_CAP _GUID_LEFT	Capturing guidance is in progress: Hand is placed too far to the left.	-	✔
		Move your hand slightly to the right.		
0x02030207	JAVA_PvAPI_NOTIFY_CAP _GUID_DOWN	Capturing guidance is in progress: Hand is placed too backward.	-	✔
		Move your hand slightly forward.		
0x02030208	JAVA_PvAPI_NOTIFY_CAP _GUID_UP	Capturing guidance is in progress: Hand is placed too forward	-	✔
		Move your hand slightly backward.		
0x02030209	JAVA_PvAPI_NOTIFY_CAP _GUID_FAR	Capturing guidance is in progress: Hand is too far from the sensor.	✔	✔
		Move your hand slightly to the sensor.		
0x0203020A	JAVA_PvAPI_NOTIFY_CAP _GUID_NEAR	Capturing guidance is in progress: Hand is placed too close. (Capturing distance between a hand and Sensor is under approximately 40mm.)	✔	✔
		Place your hand slightly far from the sensor.		
0x0203020B	JAVA_PvAPI_NOTIFY_CAP _GUID_CAPTURING	Capturing guidance is in progress: Capturing	✔	✔
		Now capturing... Please don’t move you’re your hand.		
0x02030210	JAVA_PvAPI_NOTIFY_CAP _GUID_PITCH_DOWN	Capturing guidance is in progress: Fingers are placed downward.	-	✔
		Keep your hand flat.		

Message ID		Description (1st row: Meaning and Status 2nd row: Message example)	Guide mode	
Value	Definition		With guide mode	Without guide mode
0x02030211	JAVA_PvAPI_NOTIFY_CAP_GUID_PITCH_UP	Capturing guidance is in progress: Fingers are placed upward. Keep your hand flat.	-	✔
0x02030212	JAVA_PvAPI_NOTIFY_CAP_GUID_ROLL_RIGHT	Capturing guidance is in progress: Right side of a hand is downward. Keep your hand flat.	-	✔
0x02030213	JAVA_PvAPI_NOTIFY_CAP_GUID_ROLL_LEFT	Capturing guidance is in progress: Left side of a hand is downward. Keep your hand flat.	-	✔
0x02030214	JAVA_PvAPI_NOTIFY_CAP_GUID_YAW_RIGHT	Capturing guidance is in progress: Fingers are pointing to the right. Place your hand parallel to the sensor.	-	✔
0x02030215	JAVA_PvAPI_NOTIFY_CAP_GUID_YAW_LEFT	Capturing guidance is in progress: Fingers are pointing to the left. Place your hand parallel to the sensor.	-	✔
0x02030217	JAVA_PvAPI_NOTIFY_CAP_GUID_ROUND	Capturing guidance is in progress: Hand is cupping. Keep your hand flat and move it slightly to the sensor.	✔	✔
0x02030220	JAVA_PvAPI_NOTIFY_CAP_GUID_ADJUST_LIGHT	Capturing guidance is in progress: Start the capturing again after using new modified condition. Trying to capture again. Please don't move your hand.	✔	✔
0x02030221	JAVA_PvAPI_NOTIFY_CAP_GUID_ADJUST_NG	Capturing guidance is in progress: Start the capturing again because of out of range for capturing condition. Trying to capture again. Please don't move your hand.	✔	✔
0x02030300	JAVA_PvAPI_NOTIFY_CAP_GUID_PHASE_END	Capturing phase is completed which means "Starting verification" for "JAVA_BioAPI_VerifyMatch", "JAVA_BioAPI_IdentifyMatch". Now authenticating…	✔	✔
x02030222	JAVA_PvAPI_NOTIFY_CAP_GUID_CAPTURE_IMAGE	Capturing guidance is in progress: Guidance image is notified after returning of this message ID.	(Notified when using "detailed information notification fuction for guidance image display".)	
0x02040000	JAVA_PvAPI_NOTIFY_WAIT_FOR_TRIGGER	Notified as callback massage for the start of capturing before the second capturing at enrollment of palm vein data.	(Notified when using "shutter function")	

Message ID		Description (1st row: Meaning and Status 2nd row: Message example)	Guide mode	
Value	Definition		With guide mode	Without guide mode
0x030001XX (Note 3)	JAVA_PvAPI_NOTIFY_API_KIND	The process type is notified. ・JAVA_PvAPI_NOTIFY_APL_KIND + 0x01 : JAVA_BioAPI_Capture ・JAVA_PvAPI_NOTIFY_APL_KIND + 0x02 : JAVA_BioAPI_VerifyMatch ・JAVA_PvAPI_NOTIFY_APL_KIND + 0x03 : JAVA_BioAPI_IdentifyMatch ・JAVA_PvAPI_NOTIFY_APL_KIND + 0x04 : JAVA_BioAPI_Enroll ・JAVA_PvAPI_NOTIFY_APL_KIND + 0x05 : JAVA_BioAPI_Verify ・JAVA_PvAPI_NOTIFY_APL_KIND + 0x06 : JAVA_BioAPI_Identify	(Notified when using detailed information notification for guidance image display.)	
0x030002XX (Note 4)	JAVA_PvAPI_NOTIFY_API_END	The process completion type is notified. ・JAVA_PvAPI_NOTIFY_API_END ：Successful ・JAVA_PvAPI_NOTIFY_API_END + 0x10 ：Cancel ・JAVA_PvAPI_NOTIFY_API_END + 0x20 ：Error When this message ID is notified, specify return value (JAVA_BioAPI_OK or JAVA_PvAPI_WAIT) complying with application control.	(Notified when using detailed information notification for guidance image display.)	
0x04XXXXXX (Note 5)	JAVA_PvAPI_NOTIFY_MATCH_RESULT	Authentication result is notified. <Verification> ・JAVA_PvAPI_NOTIFY_MATCH_RESULT ：Authentication NG ・JAVA_PvAPI_NOTIFY_MATCH_RESULT + 0x000001：Authentication OK <Identification> ・JAVA_PvAPI_NOTIFY_MATCH_RESULT ：No similar palm vein data ・JAVA_PvAPI_NOTIFY_MATCH_RESULT + 0xXXXXXX：Found similar palm vein data (0xXXXXXX indicates enrollment number of the most similar palm vein data item.)	(Notified when using detailed information notification function for guidance image display.)	
0x05XXXXXX (Note 6)	JAVA_PvAPI_NOTIFY_REGIST_SCORE	Quality of enrollment data is notified as score value. ・JAVA_PvAPI_REGIST_SCORE_QUALITY_1 ：Quality level 1 (Quality of palm vein data is good.) ・JAVA_PvAPI_REGIST_SCORE_QUALITY_2 ：Quality level 2 (Quality of palm vein data is acceptable.) ・JAVA_PvAPI_REGIST_SCORE_QUALITY_3 ：Quality level 3 (Quality of palm vein data is deficient.)	(Notified when using enrolled data score notification function.)	

Note 1) “JAVA_PvAPI_NOTIFY_CAP_GUID_START (0x020300MN)” is defined as a value of “0x02030000”. Value of “MN” is “11” in case of authentication, and is “2N” (N means capturing number which starts from 1) in case of enrollment.

Note 2) Authentication library may automatically repeat capturing instruction for the first capture “JAVA_PvAPI_NOTIFY_CAP_GUID_START (0x02030021)” in case palm vein is not captured properly during enrollment.

Therefore, application should be designed to be able to process the multiple notifications of capturing instruction “JAVA_PvAPI_NOTIFY_CAP_GUID _START (0x02030021 or 0x02030022)”.

Note 3) “JAVA_PvAPI_NOTIFY_API_KIND (0x030001XX)” is defined as a value of “0x03000100”.

Note 4) “JAVA_PvAPI_NOTIFY_API_END (0x030002XX)” is defined as a value of “0x03000200”.

Note 5) “JAVA_PvAPI_NOTIFY_MATCH_RESULT (0x04XXXXXX)” is defined as a value of “0x04000000”.

Note 6) “JAVA_PvAPI_NOTIFY_REGIST_SCORE (0x05XXXXXX)” is defined as a value of “0x05000000”.

The value of “0x05XXXXXX” is to be one of the following depending on score value.

・JAVA_PvAPI_REGIST_SCORE_QUALITY_1: 0x05000001

・JAVA_PvAPI_REGIST_SCORE_QUALITY_2: 0x05000002

・JAVA_PvAPI_REGIST_SCORE_QUALITY_3: 0x05000003

>See> As for the Guide mode (With guide mode/Without guide mode), refer to the “Authentication Library Reference Guide” or “ 3.2.2.15 JAVA_PvAPI_SetProfile [JAVA_uint32 Type setting value]”.

>See> As for the “detailed information notification function for guidance image display” and the “enrolled data score notification function”, refer to the ”Authentication Library Reference Guide”.

[Remark 3]

When notification type is silhouette image notification, the following items are notified.

• Width of image:	640
• Height of image:	480
• Bitmap data:	Bitmap file format, monochrome, 256 gradations

At the enrollment of palm vein data, silhouette image is notified as follows.

- Not use “shutter function”:

  The silhouette image is notified only after first capturing.
- Use “shutter function”:

  The silhouette image is notified after each of the first and second capturing.

[Remark 4]

When notification type is “guidance image notification”, guidance for guiding palm to correct position is notified to Java application via “JAVA_BioAPI_GUI_ STREAMING_CALLBACK” event handler.

Guidance image is notified when the following settings are completed.

- ➢ Detailed information notification function for guidance image display “CBGUIMessageDetail” is set as “1” (Use) in setting file of Authentication library.
- ➢ The instance of the class implements the "JAVA_BioAPI_GUI_STREAMING_CALLBACK_IF” interface using "JAVA_BioAPI_SetGUICallbacks" is registered in Interface module.

>See> As for the setting file of Authentication library, refer to the “Authentication Library Reference Guide”.

>See> For information on the “JAVA_BioAPI_SetGUICallbacks”, refer to the ”3.2.2.7　JAVA_BioAPI_SetGUICallbacks”.

>See> For information on the “JAVA_BioAPI_GUI_STREAMING_CALLBACK”, refer to the “3.2.2.25　JAVA_BioAPI_GUI_STREAMING_CALLBACK”.

3.2.2.25 JAVA_BioAPI_GUI_STREAMING_CALLBACK

[Function]

Guidance image for guiding palm to correct position is notified from Authentication library.

[Coding syntax]

```
public long JAVA_BioAPI_GUI_STREAMING_CALLBACK
(
   Object                            GuiStreamingCallbackCtx,
   JAVA_BioAPI_GUI_BITMAP            Bitmap
) ;
```

[Parameters]

Name	I/O	Description
GuiStreamingCallback Ctx	input	Notified the object specified in the parameter of "JAVA_BioAPI_SetGUICallbacks".
Bitmap	input/ optional	Guidance image is notified.

[Definition]

[Guidance image]

```
public class JAVA_BioAPI_GUI_BITMAP
{
  public long              Width;           Width of image
  public long              Height;          Height of image
  public JAVA_BioAPI_DATA  Bitmap;          Bitmap data
}
```

[Return value]

JAVA_BioAPI_OK	Successful (fixed)

[Description]

This method is called by "BioAPI_GUI_STREAMING_CALLBACK" in Authentication library.

[Caution]

Perform minimum necessary processes in the "event handler for guidance image notification" and return as soon as possible.

[Remark 1]

The “event handler for guidance image notification” is defined in “JAVA_BioAPI_GUI_STATE_CALLBACK_IF interface (event listener for guidance image notification)”. In order to use this event handler, the following is necessary in advance; create an instance of the class implements the “JAVA_BioAPI_GUI_STATE_CALLBACK_IF” interface in application, and then set it to the parameter of “JAVA_BioAPI_SetGUICallbacks” method.

›See›　For information on the “JAVA_BioAPI_SetGUICallbacks”, refer to the ”3.2.2.7　JAVA_BioAPI_SetGUICallbacks”.

[Remark 2]

The following items are notified in guidance image.

•	Width of image:	640
•	Height of image:	480
•	Bitmap data:	Bitmap file format, monochrome, 256 gradations

3.3　PalmSecureHelper Class

3.3.1　List of Methods

“PalmSecureHelper” class provides the following methods.

No.	Method	Description	Reference
1	convertBIRToByte	Convert BIR class to byte array.	3.3.2.1
2	convertByteToBIR	Convert byte array to BIR class.	3.3.2.2

3.3.2　Method Reference

Method of “PalmSecureHelper” class converts BIR class to byte array, and make palm vein data appropriate format for file output and storing to database and so on.

>See>　As for the error information, refer to the
“ Chapter4　Error Information ”.

3.3.2.1 convertBIRToByte

[Function]

Convert palm vein data reserved in BIR class to byte array.

[Coding syntax]

```
public static byte[] convertBIRToByte
 (
   JAVA_BioAPI_BIR               bir
) throws PalmSecureException, IOException
```

[Parameters]

Name	I/O	Description
bir	input	Specify the instance of BIR class.

[Return value]

Palm vein data converted to byte array

3.3.2.2 convertByteToBIR

[Function]

Convert palm vein data converted to byte array to BIR class.

[Coding syntax]

```
public static JAVA_BioAPI_BIR convertByteToBIR
(
   byte[]                      data
) throws PalmSecureException, IOException
```

[Parameters]

Name	I/O	Description
data	input	Specify palm vein data converted to byte array.

[Return value]

Palm vein data converted to instance of BIR class

Chapter4　Error Information

4.1 Error Information

It is possible to acquire error information of Authentication library and of Interface module.

4.2 Error Information of Authentication Library

When method of Interface module returns value "JAVA_BioAPI_ERRCODE_ FUNCTION_FAILED"(Error), call "JAVA_PvAPI_GetErrorInfo” method immediately after calling the method in order to acquire error information ("JAVA_PvAPI_ErrorInfo" class).

>See> For information on the “JAVA_PvAPI_GetErrorInfo”, refer to the ”3.2.2.17 JAVA_PvAPI_GetErrorInfo”.

>See> For details of error information acquired by “JAVA_PvAPI_ErrorInfo” class, refer to the ”Authentication Library Reference Guide”.

There are descriptions for each combination of “Error level”, “Error type” and “Error detail”.
When referring to the error information on the “Authentication Library Reference Guide”, please translate the language to the corresponding one as follows.

- “Callback” → “Event handler”
- “Function” → “Method”

Also, please add “JAVA_” to each name of methods.
E.g. ”PvAPI_ApAuthenticate” → ”JAVA_PvAPI_ApAuthenticate”

4.3 Error Information of Interface module

When error occurs in Interface module, "PalmSecureException” class is thrown.

Error No. notified in "PalmSecureException” class is as follows.

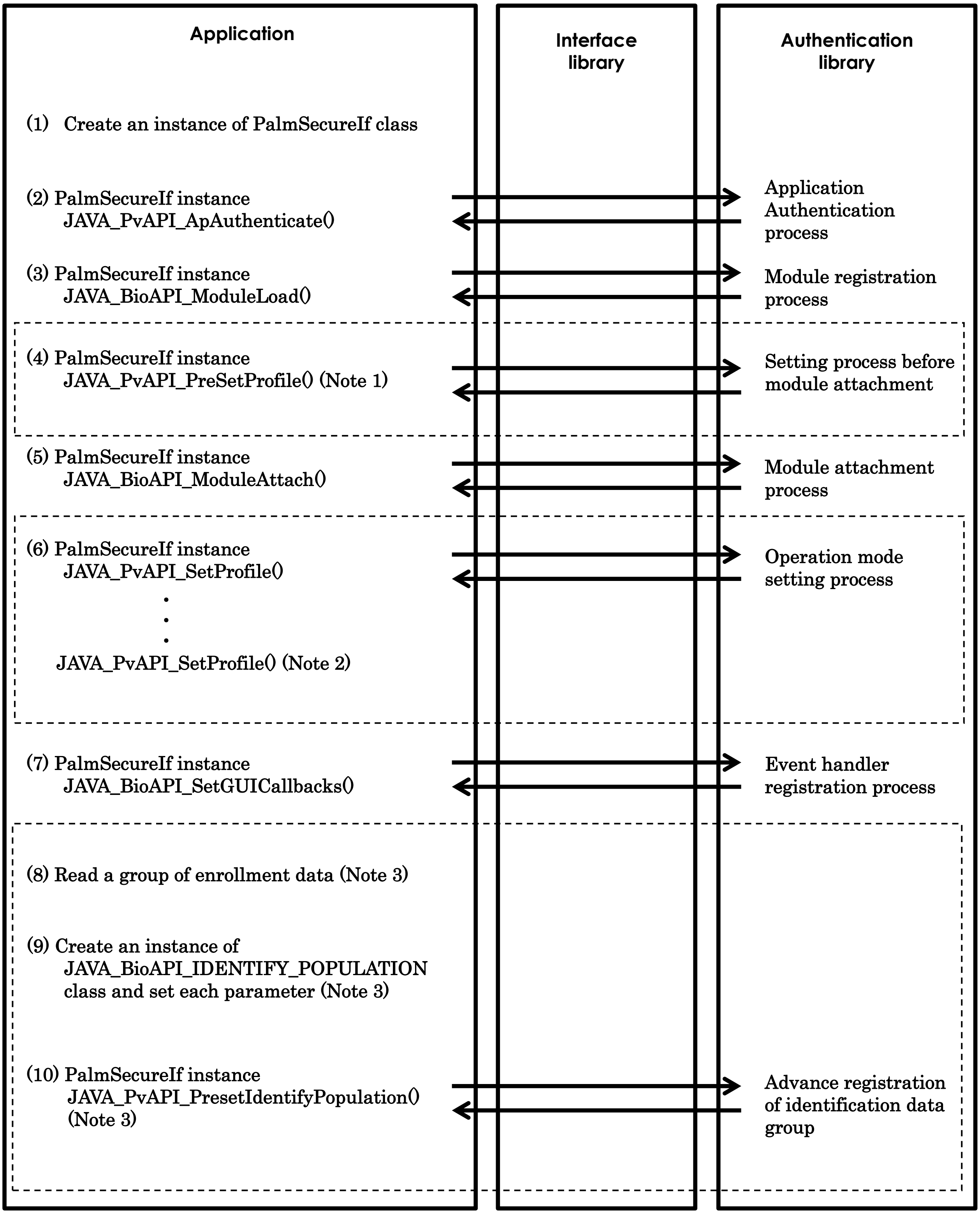
Error No. is defined in “PalmSecureConstant” class.

For detailed information, refer to "ErrCause" of "PalmSecureException” class.

Error No.	Constants	Description
100	SYSTEM_EXCEPTION	System error occurs.
200	ARGUMENT_EXCEPTION	Error in parameter for calling methods.
300	MEMORY_EXCEPTION	Failed to acquire memory.
400	ELSE_EXCEPTION	Error caused by other than the above.

Chapter5 Sequence

5.1 Initialization Sequence


Application
Interface library
Authentication library
(1) Create an instance of PalmSecureIf class
(2) PalmSecureIf instance JAVA_PvAPI_ApAuthenticate()
Application Authentication process
(3) PalmSecureIf instance JAVA_BioAPI_ModuleLoad()
Module registration process
(4) PalmSecureIf instance JAVA_PvAPI_PreSetProfile() (Note 1)
Setting process before module attachment
(5) PalmSecureIf instance JAVA_BioAPI_ModuleAttach()
Module attachment process
(6) PalmSecureIf instance JAVA_PvAPI_SetProfile()
・
・
・
JAVA_PvAPI_SetProfile() (Note 2)
Operation mode setting process
(7) PalmSecureIf instance JAVA_BioAPI_SetGUICallbacks()
Event handler registration process
(8) Read a group of enrollment data (Note 3)
(9) Create an instance of JAVA_BioAPI_IDENTIFY_POPULATION class and set each parameter (Note 3)
(10) PalmSecureIf instance JAVA_PvAPI_PresetIdentifyPopulation() (Note 3)
Advance registration of identification data group

Note 1) Call ”JAVA_PvAPI_PreSetProfile” for each setting item for multiple Sensor connection and/or multiple processing for identification.

Among the setting item of “JAVA_PvAPI_SetProfile”. the following item can also be set in setting file of Authentication library.

- Multiple processing for identification

>See> As for the multiple Sensor connection, refer to the “System Development Guide” and”5.9 When Connecting Multiple Sensors”.

>See> As for the setting file of Authentication library, refer to the “Authentication Library Reference Guide”.

Note 2) Call “JAVA_PvAPI_SetProfile” for each setting item when dynamically changing operation mode of Authentication library.

The following items specified in “JAVA_PvAPI_SetProfile” can also be specified in setting file of Authentication library.

- Guide mode
- Authentication result score notification function

!Caution **Authentication result score notification function**

Processing time for verification is slightly longer when “authentication result score notification function” is used for verification. Therefore, use this function to test authentication after enrolling palm vein data and identification process.

In such cases, specify “Do not use” in setting file and call “JAVA_PvAPI_SetProfile” before testing enrollment data or identification process with specifying “Use” each time.

>See> As for the setting file of Authentication library, refer to the “Authentication Library Reference Guide”.

Note 3) From (8) to (10) are effective in case that the number of enrollment data is large and those data is fixed while identification. Executing (8) to (10) reduces processing time for identification.

Using the method of (10), enrollment data is preserved on memory as decrypted data for a long period. Therefore, call the method in secure environment taken security measures.

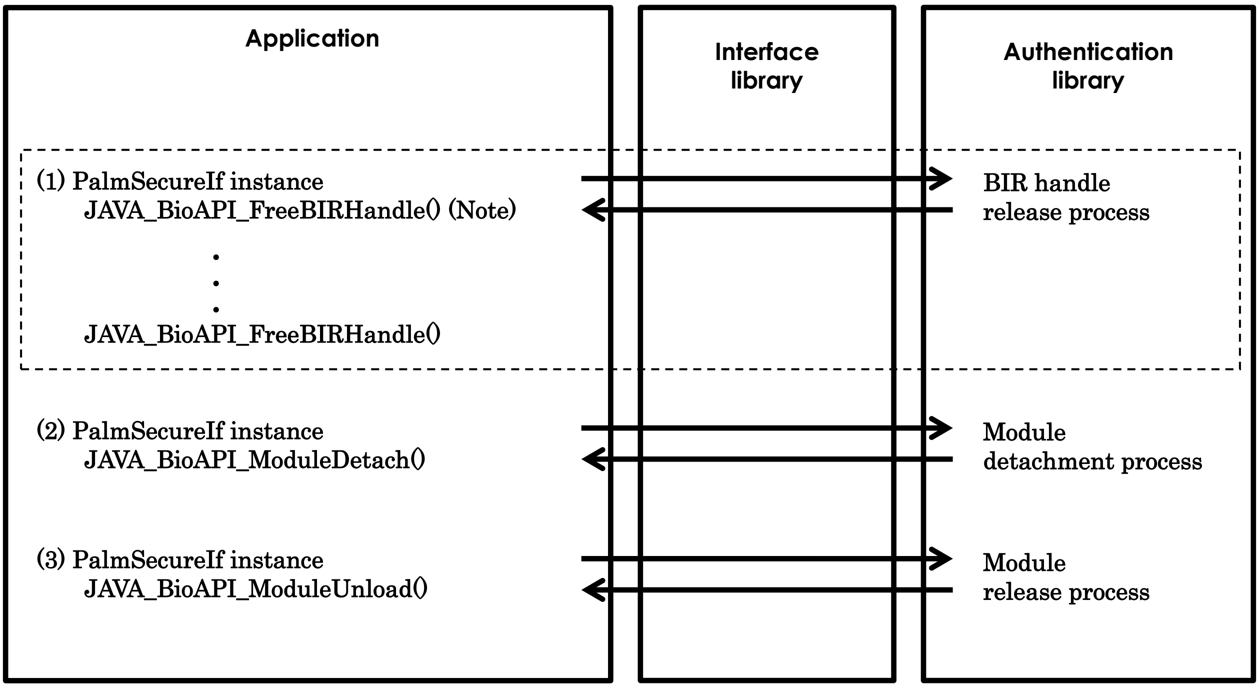
★Tip **Initialization sequence of Authentication library and starting sequence of Sensor**

Executing initialization sequence of Authentication library(from (1) to (5) in the above sequence), starting sequence of Sensor is also executed.

≻See≻ As for the processing time for starting Sensor sequence, refer to the "Authentication Library Reference Guide".

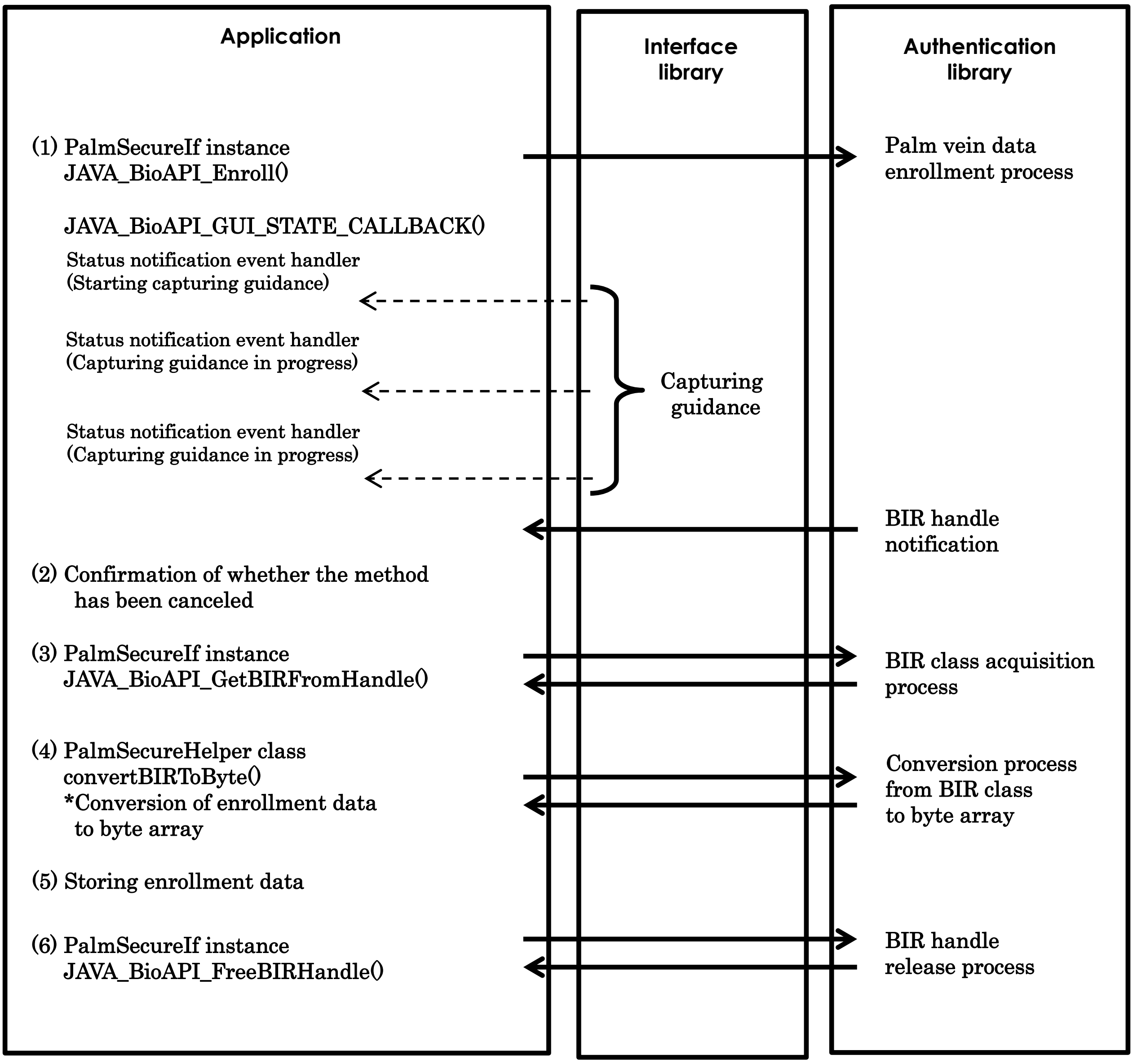
≻See≻ For information on the "JAVA_BioAPI_IDENTIFY_POPULATION" class refer to the "Appendix A.2　JAVA_BioAPI_IDENTIFY_POPULATION Class".

5.2 Termination Sequence


Application
Interface library
Authentication library
(1) PalmSecureIf instance
JAVA_BioAPI_FreeBIRHandle() (Note)
JAVA_BioAPI_FreeBIRHandle()
BIR handle release process
(2) PalmSecureIf instance
JAVA_BioAPI_ModuleDetach()
Module detachment process
(3) PalmSecureIf instance
JAVA_BioAPI_ModuleUnload()
Module release process

Note) Call “JAVA_BioAPI_FreeBIRHandle” when BIR handle notified by “JAVA_BioAPI_Enroll” or “JAVA_BioAPI_Capture” has not been released.

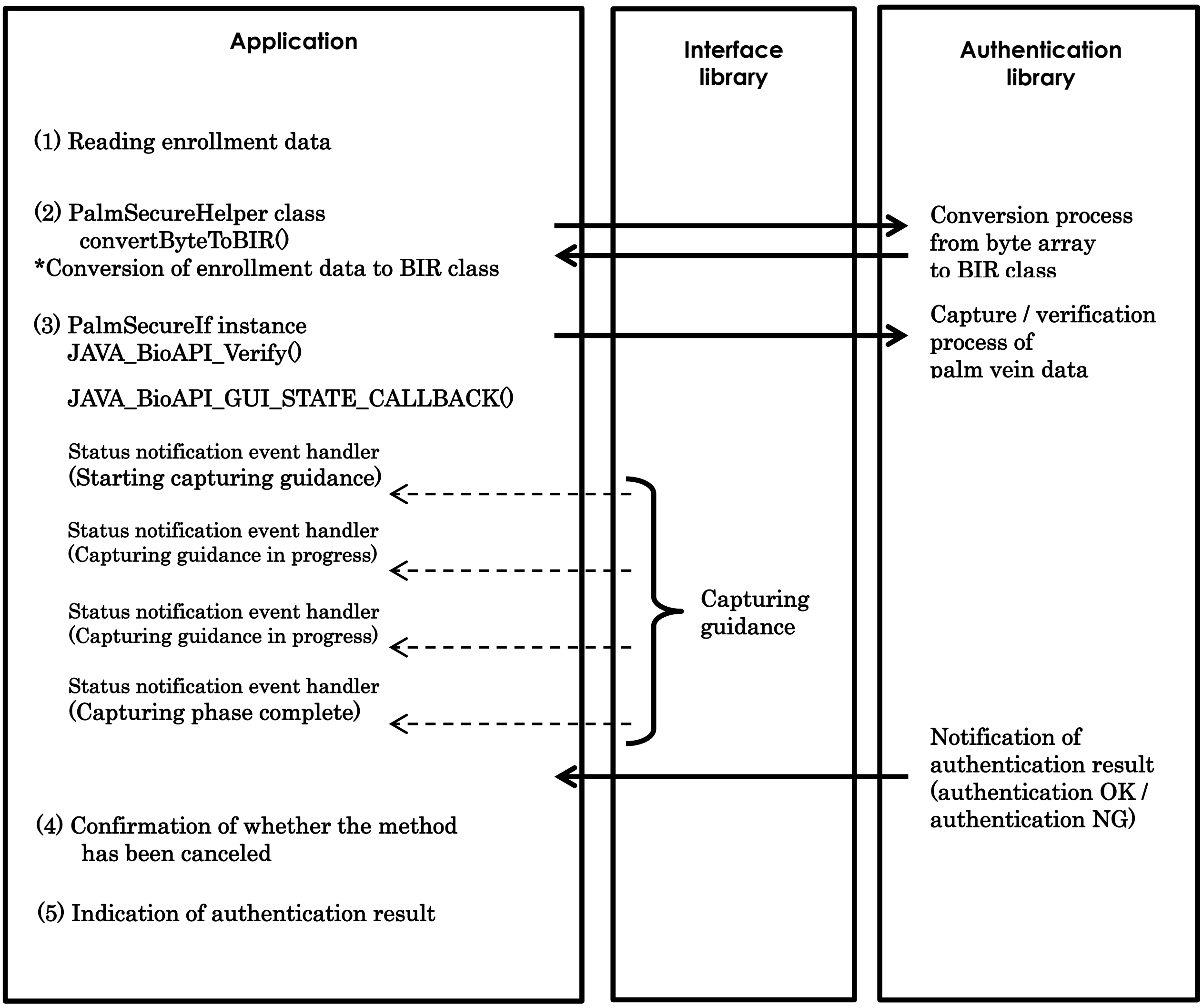
5.3 Enrollment Sequence


Application
Interface library
Authentication library
(1) PalmSecureIf instance
JAVA_BioAPI_Enroll()
Palm vein data enrollment process
JAVA_BioAPI_GUI_STATE_CALLBACK()
Status notification event handler (Starting capturing guidance)
Status notification event handler (Capturing guidance in progress)
Status notification event handler (Capturing guidance in progress)
Capturing guidance
BIR handle notification
(2) Confirmation of whether the method has been canceled
(3) PalmSecureIf instance
JAVA_BioAPI_GetBIRFromHandle()
BIR class acquisition process
(4) PalmSecureHelper class
convertBIRToByte()
*Conversion of enrollment data to byte array
Conversion process from BIR class to byte array
(5) Storing enrollment data
(6) PalmSecureIf instance
JAVA_BioAPI_FreeBIRHandle()
BIR handle release process

>See> In order to confirm whether the method has been canceled, refer to the "5.6 Cancellation Sequence".

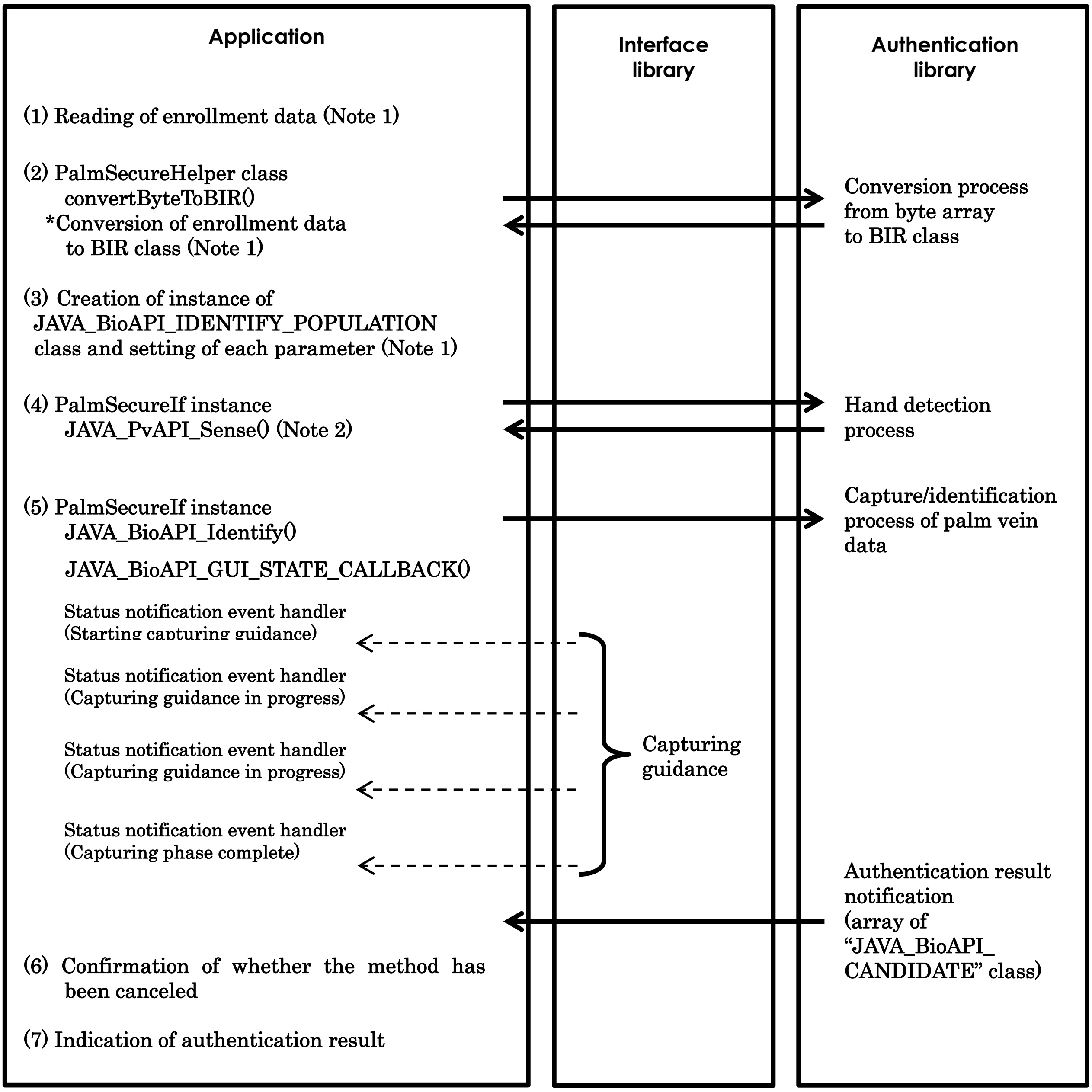
>See> As for the enrollment data, refer to the "Authentication Library Reference Guide".

5.4 Capture and Verification Sequence (for Stand Alone Configuration)


Application
Interface library
Authentication library
(1) Reading enrollment data
(2) PalmSecureHelper class
convertByteToBIR()
*Conversion of enrollment data to BIR class
Conversion process from byte array to BIR class
(3) PalmSecureIf instance
JAVA_BioAPI_Verify()
Capture / verification process of palm vein data
JAVA_BioAPI_GUI_STATE_CALLBACK()
Status notification event handler (Starting capturing guidance)
Status notification event handler (Capturing guidance in progress)
Status notification event handler (Capturing guidance in progress)
Status notification event handler (Capturing phase complete)
Capturing guidance
Notification of authentication result (authentication OK / authentication NG)
(4) Confirmation of whether the method has been canceled
(5) Indication of authentication result

>See> In order to confirm whether the method has been canceled, refer to the "5.6 Cancellation Sequence".

5.5 Capture and Identification Sequence (for Stand Alone Configuration)

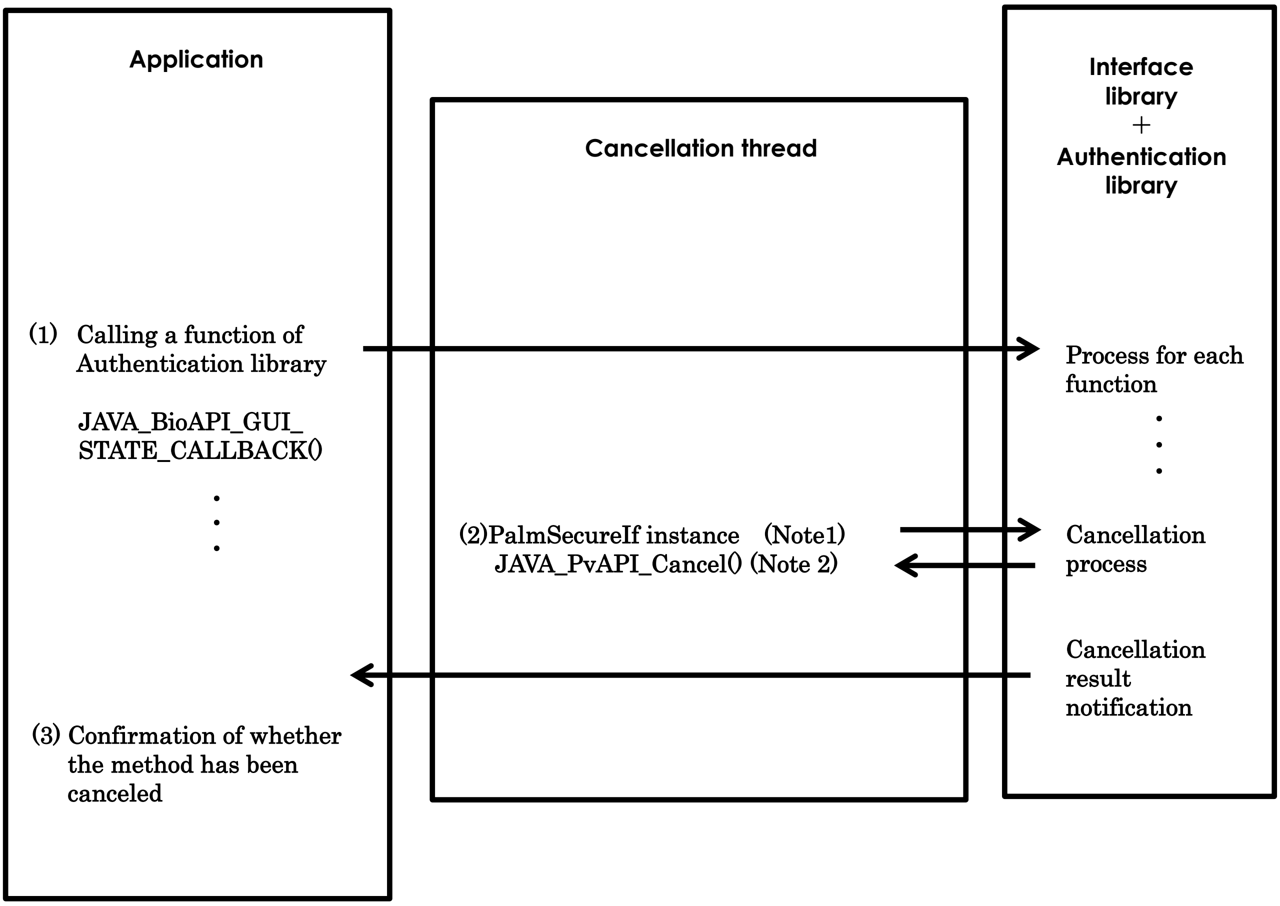

Application
Interface library
Authentication library
(1) Reading of enrollment data (Note 1)
(2) PalmSecureHelper class convertByteToBIR()
*Conversion of enrollment data to BIR class (Note 1)
Conversion process from byte array to BIR class
(3) Creation of instance of JAVA_BioAPI_IDENTIFY_POPULATION class and setting of each parameter (Note 1)
(4) PalmSecureIf instance JAVA_PvAPI_Sense() (Note 2)
Hand detection process
(5) PalmSecureIf instance JAVA_BioAPI_Identify()
Capture/identification process of palm vein data
JAVA_BioAPI_GUI_STATE_CALLBACK()
Status notification event handler (Starting capturing guidance)
Status notification event handler (Capturing guidance in progress)
Status notification event handler (Capturing guidance in progress)
Status notification event handler (Capturing phase complete)
Capturing guidance
Authentication result notification (array of "JAVA_BioAPI_ CANDIDATE" class)
(6) Confirmation of whether the method has been canceled
(7) Indication of authentication result

Note 1) From (1) to (3) are not necessary in case "advance registration of identification data group" (From (8) to (10) of Initialization Sequence) has been executed.

Note 2) Call "JAVA_PvAPI_Sense" when calling "JAVA_BioAPI_Identify" at a timing of placing a palm. (It is not necessary when calling "JAVA_BioAPI_Identify" at timing such as pressing a button.)

>See> In order to confirm whether the method has been canceled, refer to the "5.6 Cancellation Sequence".

5.6 Cancellation Sequence


Application
Cancellation thread
Interface library + Authentication library
(1) Calling a function of Authentication library
JAVA_BioAPI_GUI_STATE_CALLBACK()
Process for each function
(2)PalmSecureIf instance (Note1)
JAVA_PvAPI_Cancel() (Note 2)
Cancellation process
Cancellation result notification
(3) Confirmation of whether the method has been canceled

Note 1) As for "PalmSecureIf" instance for cancellation thread, use the same "PalmSecureIf" instance for main thread.

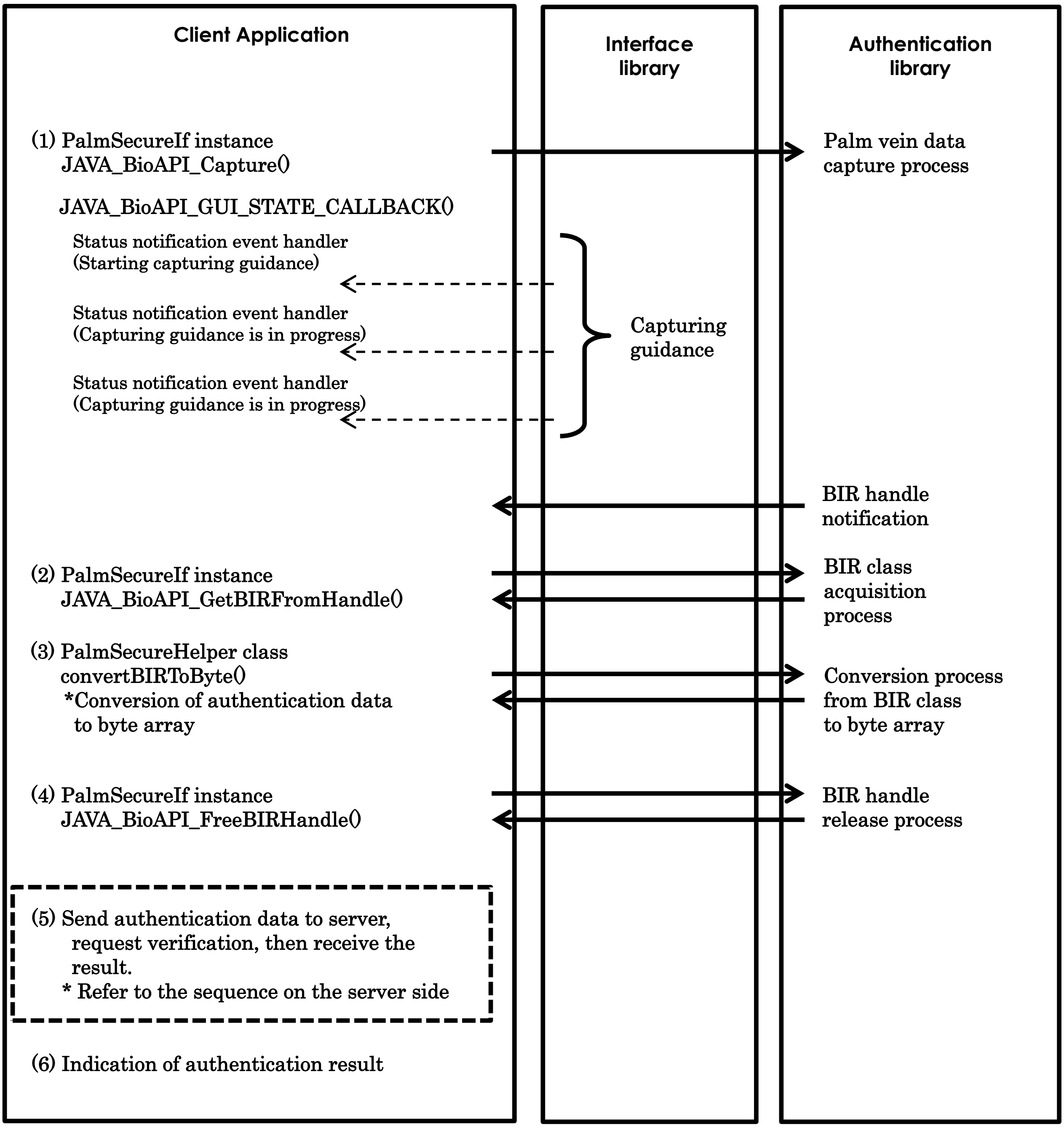
Note 2) "JAVA_PvAPI_Cancel" operates asynchronously from other methods.

>See> As for the target method of cancellation process, refer to the "3.2.2.19 JAVA_PvAPI_Cancel".

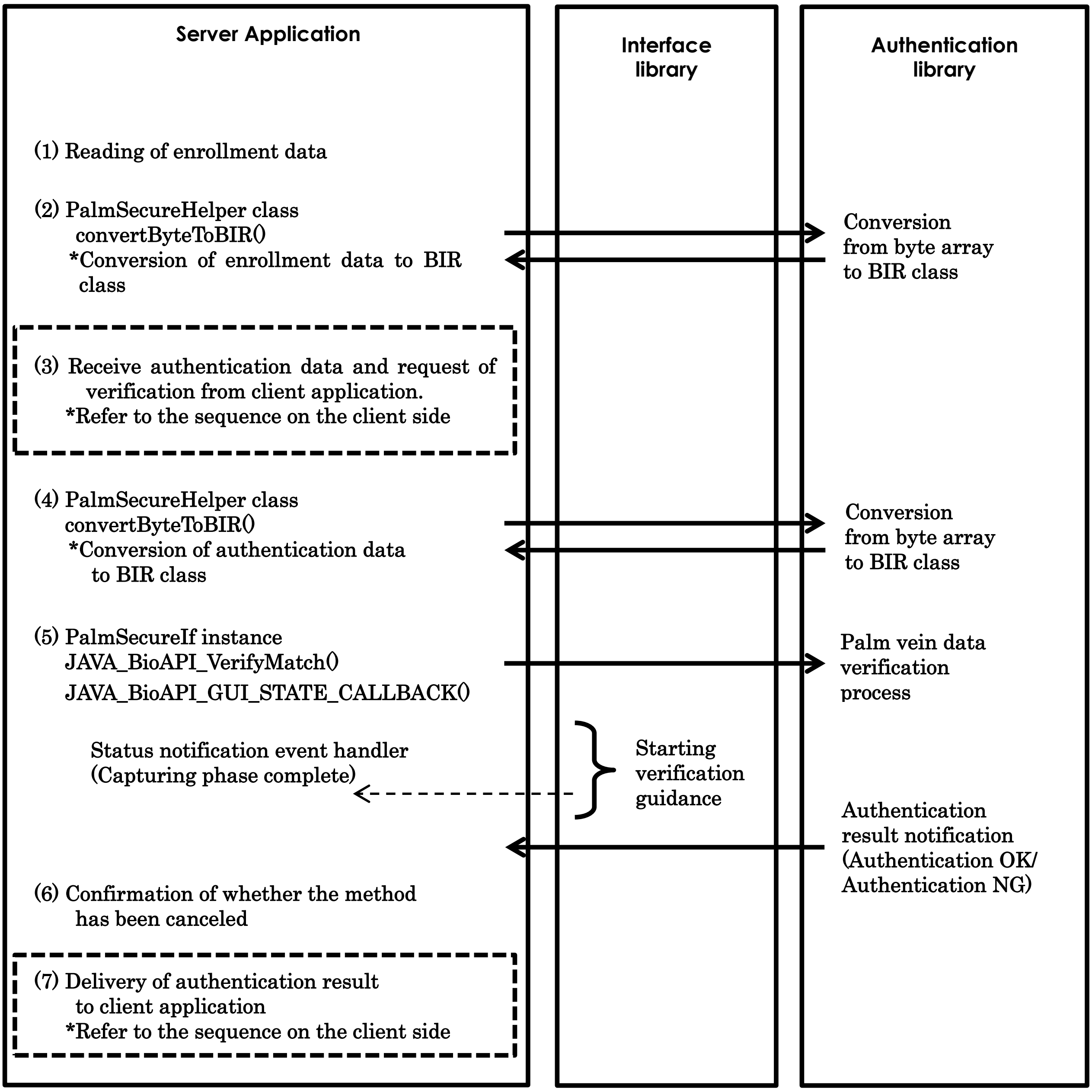
5.7 Client Server Configuration

5.7.1 Capture and Verification Sequence

◆ Client side


Client Application
Interface library
Authentication library
(1) PalmSecureIf instance
JAVA_BioAPI_Capture()
Palm vein data capture process
JAVA_BioAPI_GUI_STATE_CALLBACK()
Status notification event handler
(Starting capturing guidance)
Status notification event handler
(Capturing guidance is in progress)
Status notification event handler
(Capturing guidance is in progress)
Capturing guidance
BIR handle notification
(2) PalmSecureIf instance
JAVA_BioAPI_GetBIRFromHandle()
BIR class acquisition process
(3) PalmSecureHelper class
convertBIRToByte()
*Conversion of authentication data to byte array
Conversion process from BIR class to byte array
(4) PalmSecureIf instance
JAVA_BioAPI_FreeBIRHandle()
BIR handle release process
(5) Send authentication data to server, request verification, then receive the result.
* Refer to the sequence on the server side
(6) Indication of authentication result

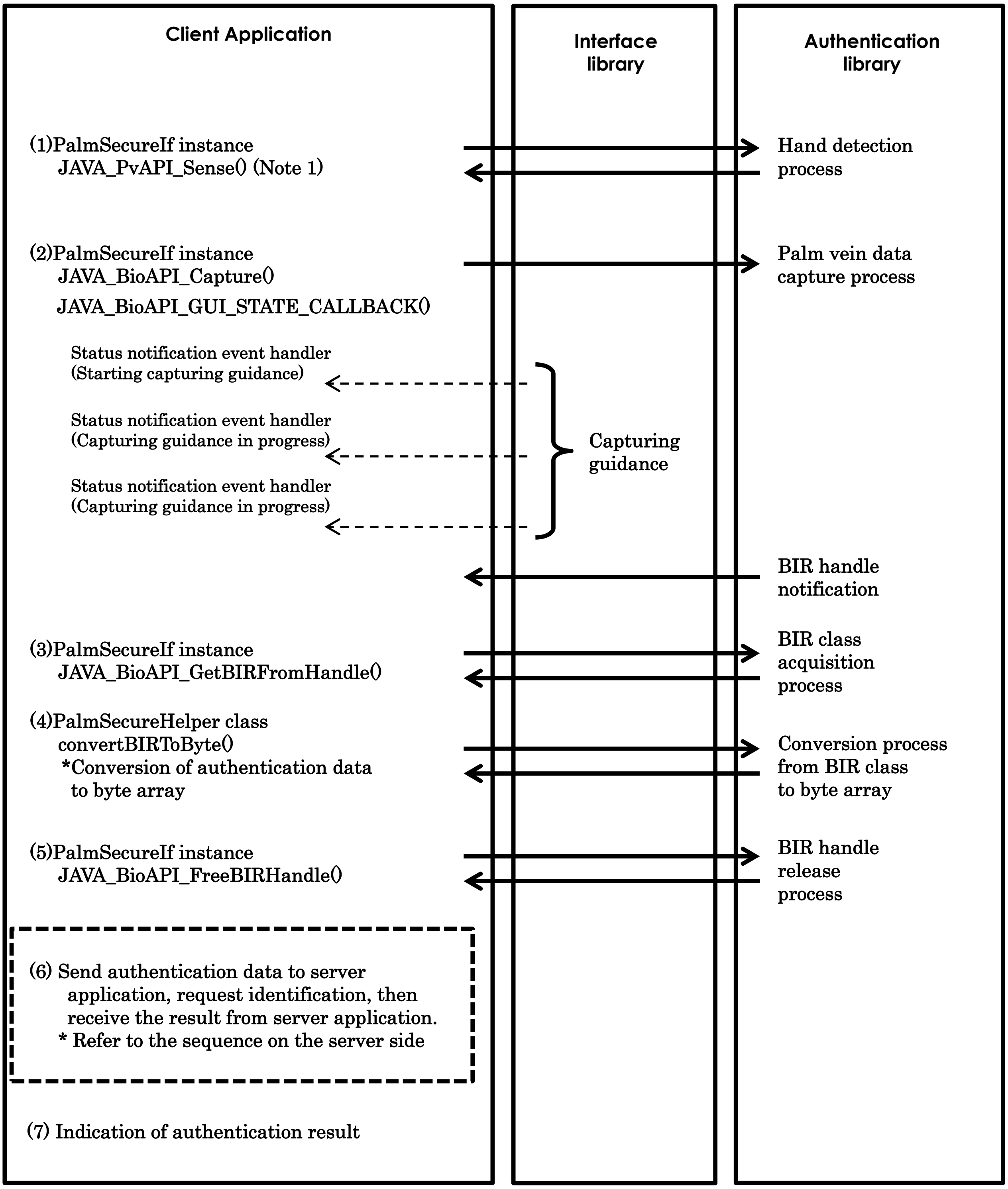
◆　Server side


Server Application
Interface library
Authentication library
(1) Reading of enrollment data
(2) PalmSecureHelper class
convertByteToBIR()
*Conversion of enrollment data to BIR class
Conversion from byte array to BIR class
(3) Receive authentication data and request of verification from client application.
*Refer to the sequence on the client side
(4) PalmSecureHelper class
convertByteToBIR()
*Conversion of authentication data to BIR class
Conversion from byte array to BIR class
(5) PalmSecureIf instance
JAVA_BioAPI_VerifyMatch()
JAVA_BioAPI_GUI_STATE_CALLBACK()
Palm vein data verification process
Status notification event handler
(Capturing phase complete)
Starting verification guidance
Authentication result notification (Authentication OK/ Authentication NG)
(6) Confirmation of whether the method has been canceled
(7) Delivery of authentication result to client application
*Refer to the sequence on the client side

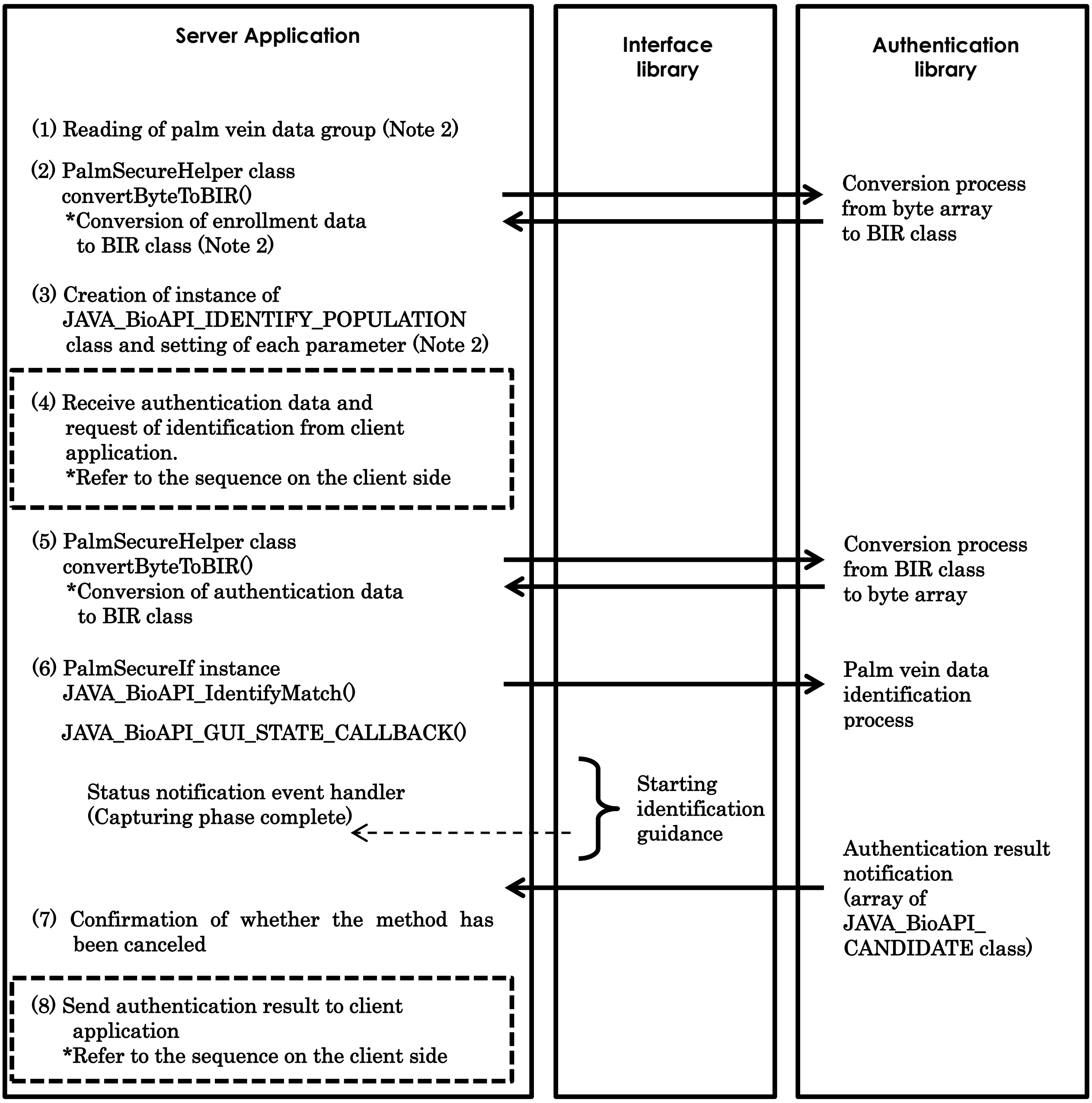
>See>　In order to confirm whether the method has been canceled, refer to the "5.6　Cancellation Sequence".

5.7.2 Capture and Identification Sequence

◆ Client side


Client Application
Interface library
Authentication library
(1)PalmSecureIf instance
JAVA_PvAPI_Sense() (Note 1)
Hand detection process
(2)PalmSecureIf instance
JAVA_BioAPI_Capture()
JAVA_BioAPI_GUI_STATE_CALLBACK()
Palm vein data capture process
Status notification event handler
(Starting capturing guidance)
Status notification event handler
(Capturing guidance in progress)
Status notification event handler
(Capturing guidance in progress)
Capturing guidance
BIR handle notification
(3)PalmSecureIf instance
JAVA_BioAPI_GetBIRFromHandle()
BIR class acquisition process
(4)PalmSecureHelper class
convertBIRToByte()
*Conversion of authentication data to byte array
Conversion process from BIR class to byte array
(5)PalmSecureIf instance
JAVA_BioAPI_FreeBIRHandle()
BIR handle release process
(6) Send authentication data to server application, request identification, then receive the result from server application.
* Refer to the sequence on the server side
(7) Indication of authentication result

◆　Server side



(1) Reading of palm vein data group (Note 2)

(2) PalmSecureHelper class convertByteToBIR()
*Conversion of enrollment data to BIR class (Note 2)

Conversion process from byte array to BIR class

(3) Creation of instance of JAVA_BioAPI_IDENTIFY_POPULATION class and setting of each parameter (Note 2)

(4) Receive authentication data and request of identification from client application.
*Refer to the sequence on the client side

(5) PalmSecureHelper class convertByteToBIR()
*Conversion of authentication data to BIR class

Conversion process from BIR class to byte array

(6) PalmSecureIf instance JAVA_BioAPI_IdentifyMatch()

JAVA_BioAPI_GUI_STATE_CALLBACK()

Palm vein data identification process

Status notification event handler (Capturing phase complete)

Starting identification guidance

Authentication result notification (array of JAVA_BioAPI_ CANDIDATE class)

(7) Confirmation of whether the method has been canceled

(8) Send authentication result to client application
*Refer to the sequence on the client side

Note 1)　Call "JAVA_PvAPI_Sense" when calling "JAVA_BioAPI_Capture" at a timing of placing a palm. (It is not necessary when calling "JAVA_BioAPI_Capture" at timing such as pressing a button.)

Note 2)　From (1) to (3) are not necessary in case "advance registration of identification data group" (From (8) to (10) of Initialization Sequence) has been executed.

>See>　In order to confirm whether the method has been canceled, refer to the "5.6　Cancellation Sequence".

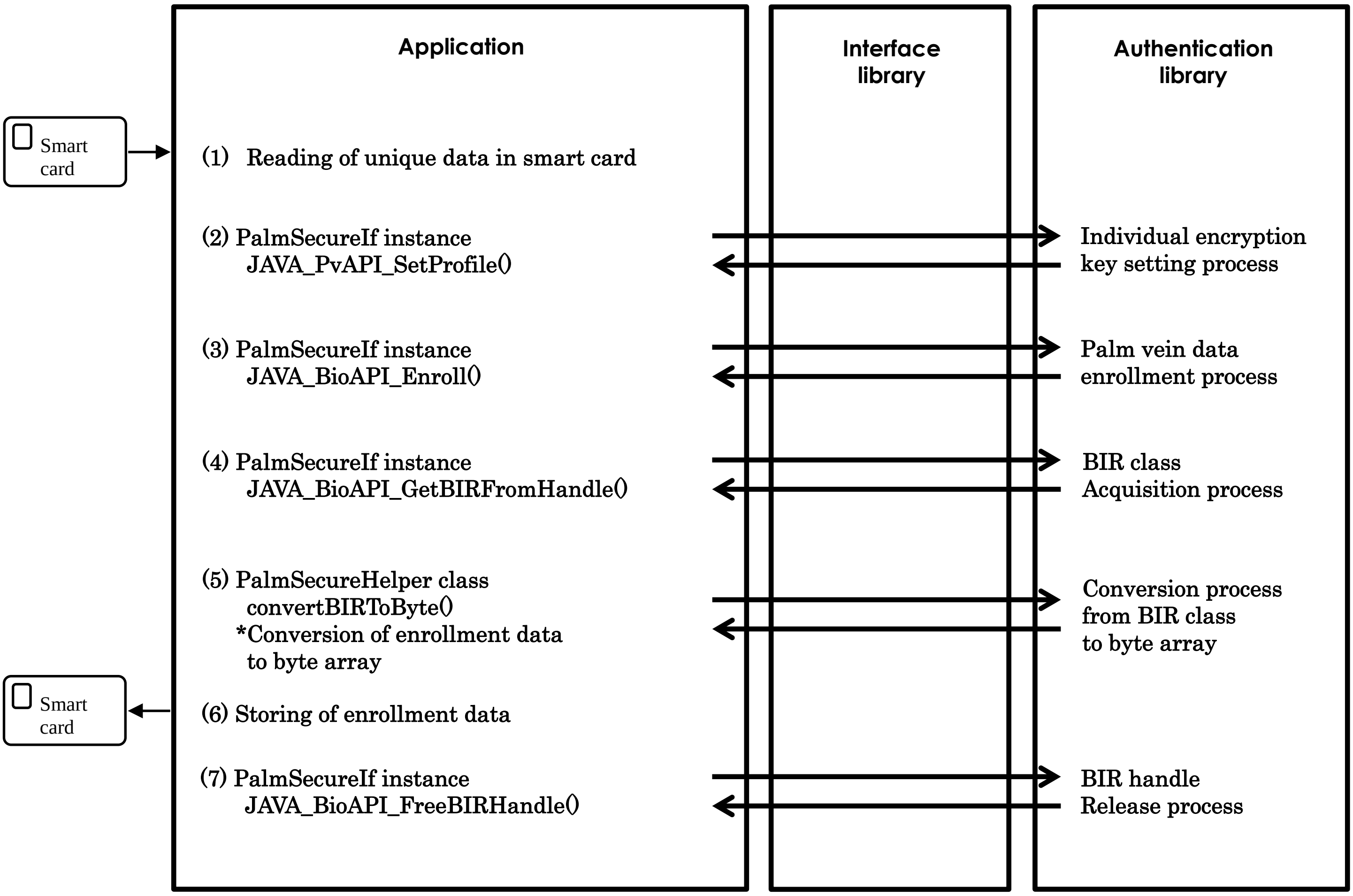
5.8 Individual Encryption Key Setting Sequence

The following shows a sequence of individual encryption key setting.

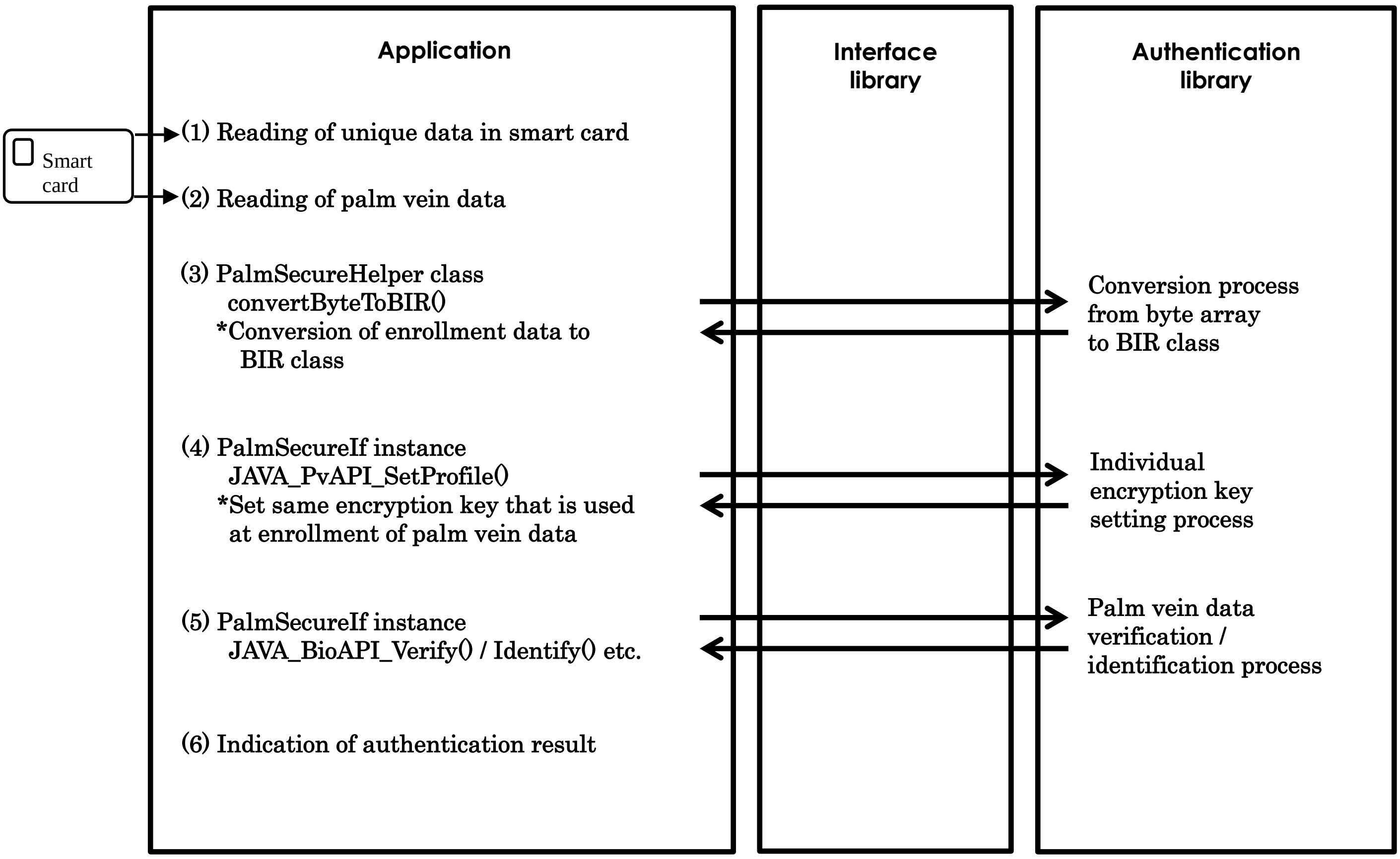
It describes an example of using different encryption keys for each IC card to encrypt palm vein data.

Also it describes only processes specific to the individual encryption key setting, and it omits basic process sequence.

5.8.1 Palm Vein Data Enrollment Sequence


Application
Interface library
Authentication library
Smart card
(1) Reading of unique data in smart card
(2) PalmSecureIf instance JAVA_PvAPI_SetProfile()
Individual encryption key setting process
(3) PalmSecureIf instance JAVA_BioAPI_Enroll()
Palm vein data enrollment process
(4) PalmSecureIf instance JAVA_BioAPI_GetBIRFromHandle()
BIR class Acquisition process
(5) PalmSecureHelper class convertBIRToByte() *Conversion of enrollment data to byte array
Conversion process from BIR class to byte array
Smart card
(6) Storing of enrollment data
(7) PalmSecureIf instance JAVA_BioAPI_FreeBIRHandle()
BIR handle Release process

5.8.2 Verification/Identification Sequence


Application
Interface library
Authentication library
Smart card
(1) Reading of unique data in smart card
(2) Reading of palm vein data
(3) PalmSecureHelper class
convertByteToBIR()
*Conversion of enrollment data to BIR class
Conversion process from byte array to BIR class
(4) PalmSecureIf instance
JAVA_PvAPI_SetProfile()
*Set same encryption key that is used at enrollment of palm vein data
Individual encryption key setting process
(5) PalmSecureIf instance
JAVA_BioAPI_Verify() / Identify() etc.
Palm vein data verification / identification process
(6) Indication of authentication result

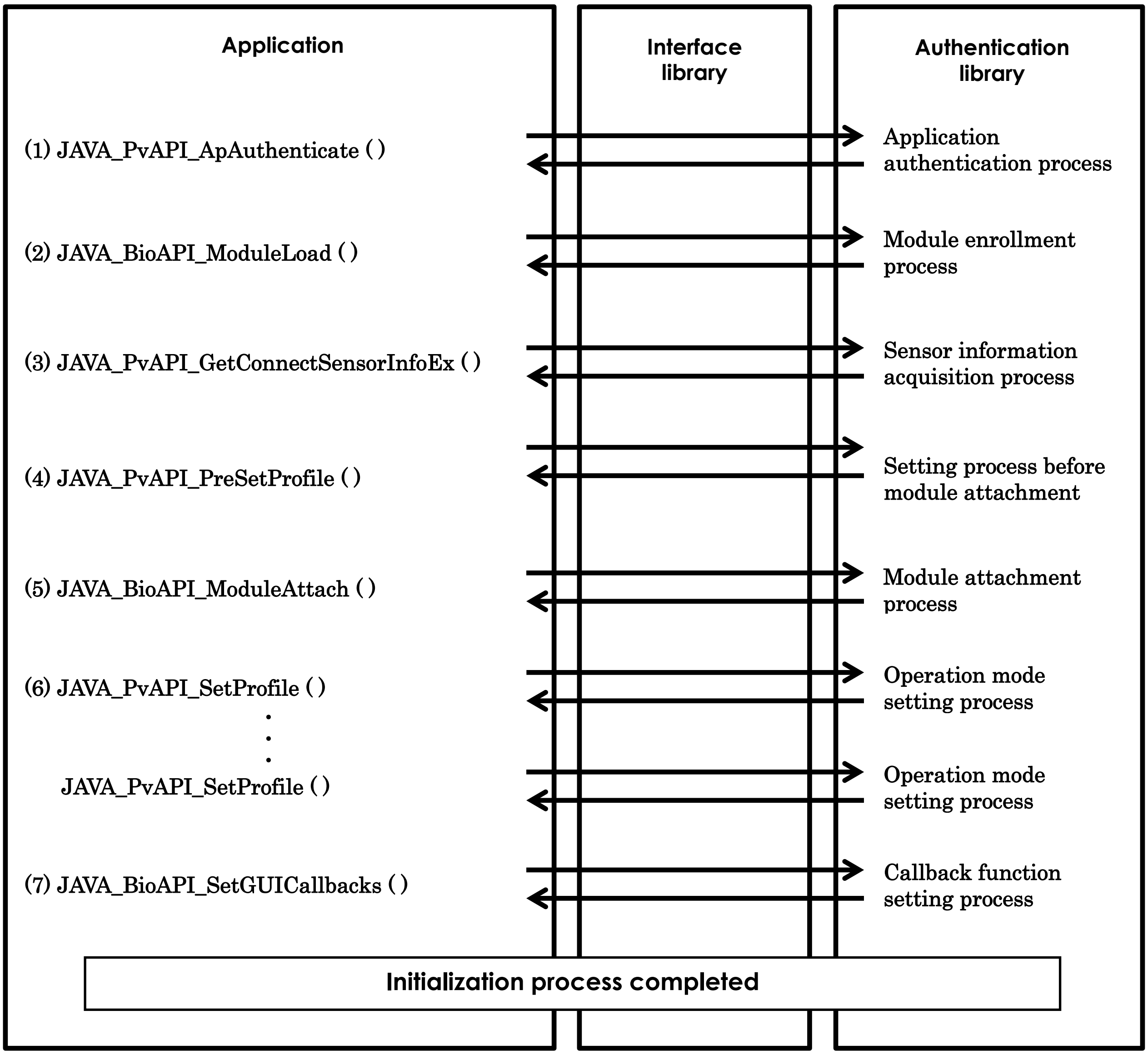
★Tip **In case the individual encryption key is not set**

Encryption is executed using same encryption key in Authentication library.

5.9 When Connecting Multiple Sensors

In order to use target hardware connecting multiple Sensors, it is necessary to switch sensors as follows.

5.9.1 Sensor Switching Sequence


Application
Interface library
Authentication library
(1) JAVA_PvAPI_ApAuthenticate ( )
Application authentication process
(2) JAVA_BioAPI_ModuleLoad ( )
Module enrollment process
(3) JAVA_PvAPI_GetConnectSensorInfoEx ( )
Sensor information acquisition process
(4) JAVA_PvAPI_PreSetProfile ( )
Setting process before module attachment
(5) JAVA_BioAPI_ModuleAttach ( )
Module attachment process
(6) JAVA_PvAPI_SetProfile ( )
Operation mode setting process
JAVA_PvAPI_SetProfile ( )
Operation mode setting process
(7) JAVA_BioAPI_SetGUICallbacks ( )
Callback function setting process
Initialization process completed

★Tip **In order to switch a Sensor again**
Executing termination sequence of Authentication library and repeat the above sensor switching sequence.

>See> As for the termination sequence, refer to the ”5.2 Termination Sequence”.

Appendix

Appendix A Other Class

A.1 JAVA_BioAPI_INPUT_BIR Class

This “JAVA_BioAPI_INPUT_BIR” class is provided in order to store enrollment and authentication data.

Use instance of this class as a parameter when calling the following method.

- JAVA_BioAPI_VerifyMatch
- JAVA_BioAPI_IdentifyMatch
- JAVA_BioAPI_Verify

Usage of “JAVA_BioAPI_INPUT_BIR” class

The following shows how to use “JAVA_BioAPI_ INPUT_BIR” class.

(1) Create an instance of “JAVA_BioAPI_INPUT_BIR” class.

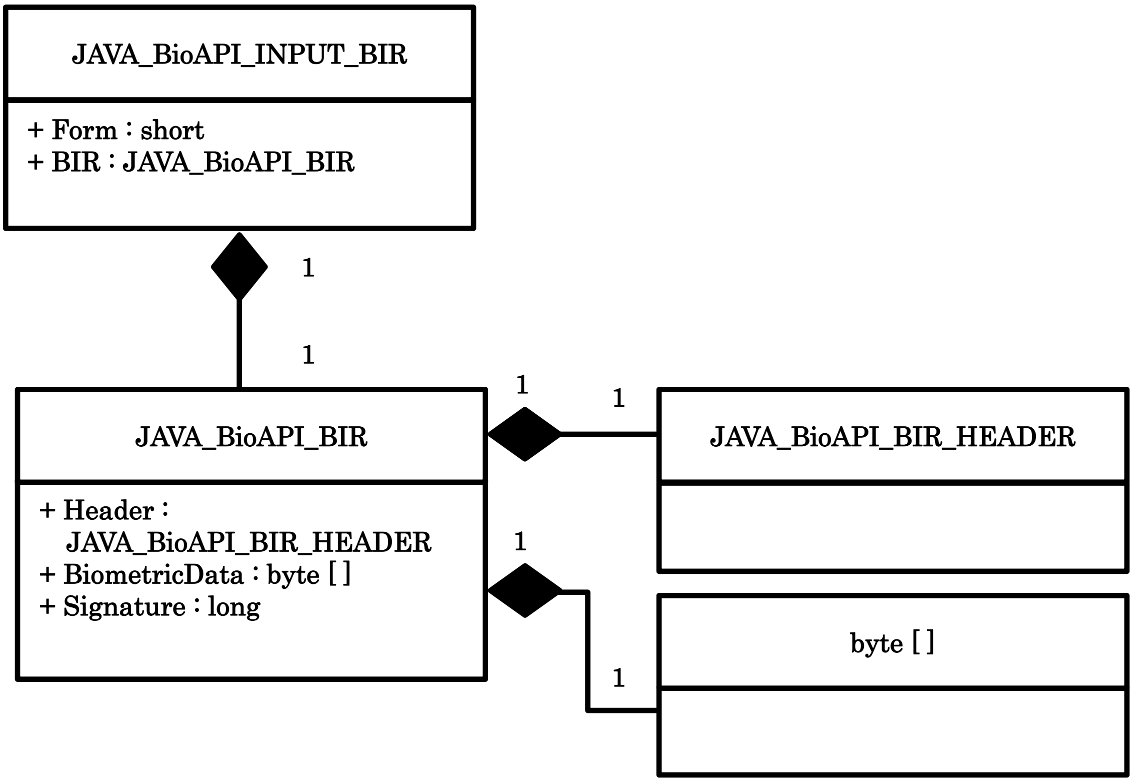
(2) Set values to the following two fields of the instance created in (1).
It is not necessary to set values to other fields.

- Set “JAVA_BioAPI_ FULLBIR_INPUT” of “PalmSecureConstant” class to field “Form”.
- Set instance of “JAVA_BioAPI_BIR” class to field “BIR”.

Create an instance of “JAVA_BioAPI_BIR” class by each of the followings.

- Acquire instance of “JAVA_BioAPI_BIR” class as a return value of “convertByteToBIR” method of “PalmSecureHelper” class.
- Create an instance of “JAVA_BioAPI_BIR” class at application and set palm vein data for enrollment/ authentication using “JAVA_BioAPI_GetBIRFromHandle” method.

>See> For detailed information on setting, refer to the source program of Sample application.


JAVA_BioAPI_INPUT_BIR
+ Form : short
+ BIR : JAVA_BioAPI_BIR
1
1
JAVA_BioAPI_BIR
+ Header :
JAVA_BioAPI_BIR_HEADER
+ BiometricData : byte [ ]
+ Signature : long
1
1
JAVA_BioAPI_BIR_HEADER
1
byte [ ]
1

A.2 JAVA_BioAPI_IDENTIFY_POPULATION Class

This "JAVA_BioAPI_IDENTIFY_POPULATION" class is provided in order to store enrollment data for identification.

Use instance of this class as a parameter when calling the following method.

- JAVA_BioAPI_Identify
- JAVA_BioAPI_IdentifyMatch
- JAVA_BioAPI_PresetIdentifyPopulation

!Caution **Keep the number of enrollment data to be stored within 1,000.**
When setting enrollment data with following steps, please set the data within 1,000.

Usage of "JAVA_BioAPI_IDENTIFY_POPULATION" class

The following shows how to use "JAVA_BioAPI_IDENTIFY_POPULATION" class.

◆ In order to use enrollment data group which has not been set using "JAVA_BioAPI_Identify" or "JAVA_BioAPI_IdentifyMatch" or in order to set enrollment data group in advance using "JAVA_PvAPI_PresetIdentifyPopulation".

(1) Create an array of "JAVA_BioAPI_BIR" class.
Set the size of array to be greater than the number of enrollment data to be used.

(2) Set instance of "JAVA_BioAPI_BIR" class to the array of "JAVA_BioAPI_BIR" class created in (1).
As for the instance of "JAVA_BioAPI_BIR" class, create it by each of the followings.

- Acquire instance of "JAVA_BioAPI_BIR" class as a return value of "convertByteToBIR" method of "PalmSecureHelper" class.

- Create an instance of "JAVA_BioAPI_BIR" class at application and set enrollment data using "JAVA_BioAPI_GetBIRFromHandle" method.

(3) Create an instance of "JAVA_BioAPI_BIR_ARRAY_POPULATION" class.

(4) Set value to the following two fields of the instance created in (3).

- ➢ Set the array of “JAVA_BioAPI_BIR” which is set in (2) to field “Members”.

- ➢ Set the number of instance of “JAVA_BioAPI_BIR” class which is set in (2) to field “NumberOfMembers”.

(5) Create an instance of “JAVA_BioAPI_IDENTIFY_POPULATION” class.

(6) Set values to the following two fields of the instance created in (5).
It is not necessary to set values to other fields.

- ➢ Set “JAVA_BioAPI_ ARRAY_TYPE” of “PalmSecureConstant” class to field “Type”.

- ➢ Set instance of “JAVA_BioAPI_BIR_ ARRAY_POPULATION” class which is set in (5) to field “BIRArray”.

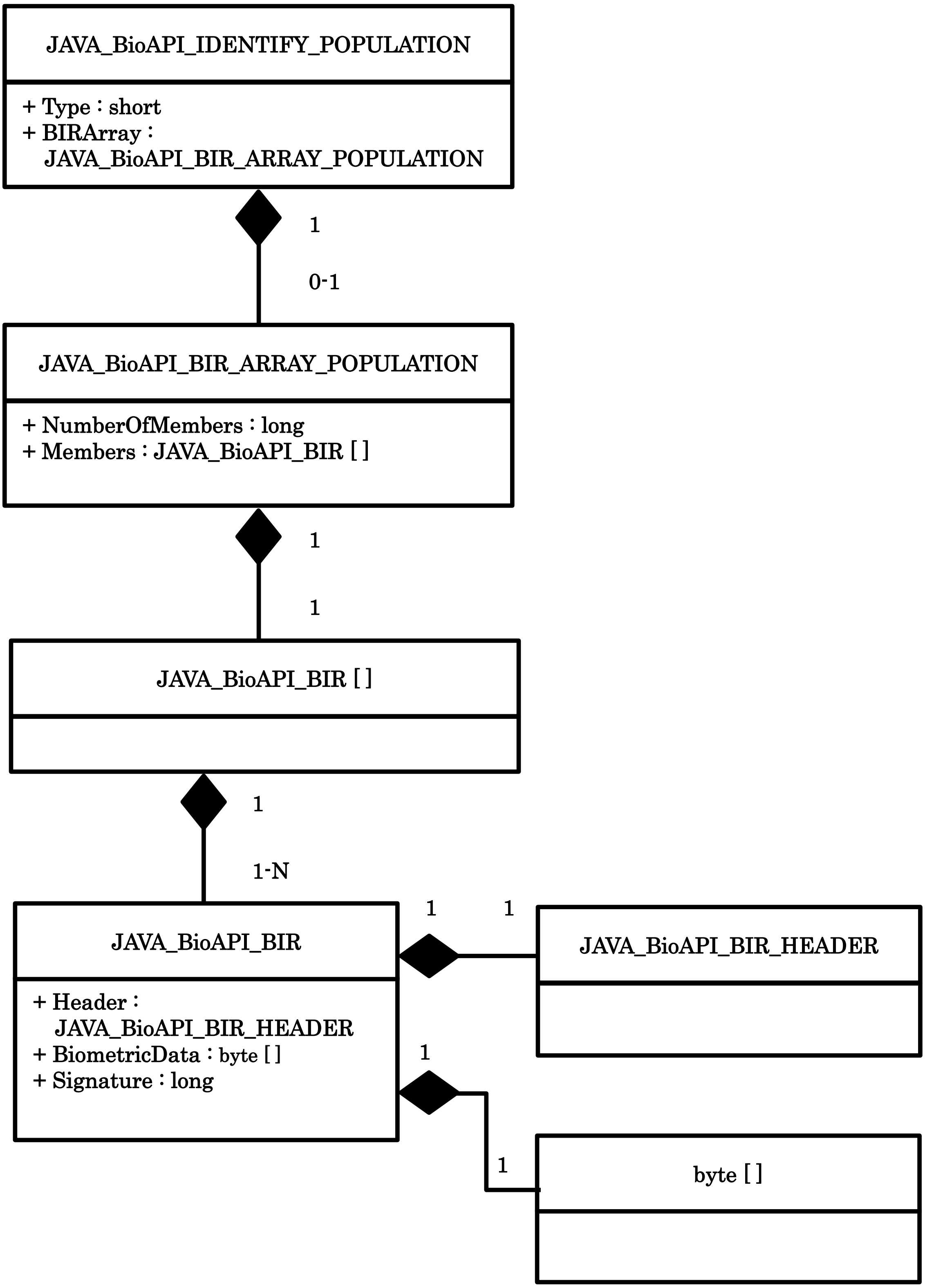
◆ In order to use enrollment data group which has been set using “JAVA_BioAPI_Identify” or “JAVA_BioAPI_IdentifyMatch”

(1) Create an instance of “JAVA_BioAPI_IDENTIFY_POPULATION” class.

(2) Set values to the following two fields of the instance created in (1).
It is not necessary to set values to other fields.

- ➢ Set “JAVA_PvAPI_ PRESET_ARRAY_TYPE” of “PalmSecureConstant” class to field “Type”.
- ➢ Set “null” to field “BIRArray”.

>See> For detailed informaton on setting, refer to the source program of Sample application.


JAVA_BioAPI_IDENTIFY_POPULATION
+ Type : short
+ BIRArray :
JAVA_BioAPI_BIR_ARRAY_POPULATION
1
0-1
JAVA_BioAPI_BIR_ARRAY_POPULATION
+ NumberOfMembers : long
+ Members : JAVA_BioAPI_BIR [ ]
1
1
JAVA_BioAPI_BIR [ ]
1
1-N
JAVA_BioAPI_BIR
+ Header :
JAVA_BioAPI_BIR_HEADER
+ BiometricData : byte [ ]
+ Signature : long
1
1
JAVA_BioAPI_BIR_HEADER
1
1
byte [ ]

A.3 JAVA_BioAPI_CANDIDATE Class

This “JAVA_BioAPI_CANDIDATE” class is provided in order to store candidate items selected by a result of identification.

Use array of this class as a parameter when calling the following methods.

- JAVA_BioAPI_Identify
- JAVA_BioAPI_IdentifyMatch

Usage of “JAVA_BioAPI_CANDIDATE” class

The following shows how to use “JAVA_BioAPI_CANDIDATE” class.

(1) Create an array of “JAVA_BioAPI_CANDIDATE” class.
Set the size of array to be greater than the value of the parameter “MaxNumberOfResults” which is set in “JAVA_BioAPI_Identify” or “JAVA_BioAPI_IdentifyMatch” to be used.

Setting value of array of “JAVA_BioAPI_CANDIDATE” class

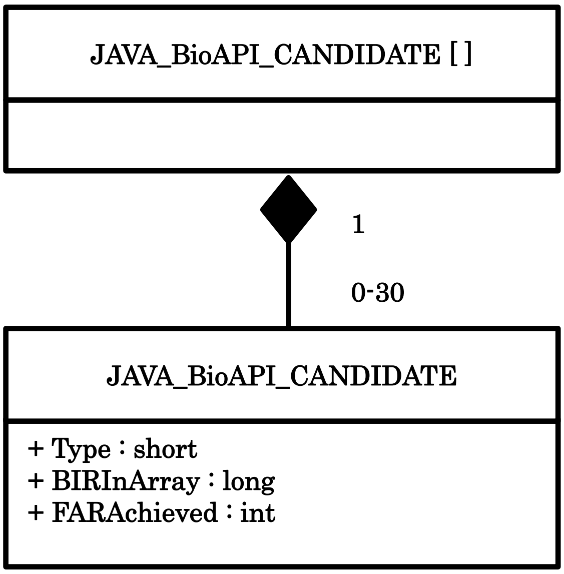
The following shows value to be set in array of “JAVA_BioAPI_CANDIDATE” class.

As for the array of “JAVA_BioAPI_CANDIDATE” class, which is set in (1), the less than the number of instance set in “NumberOfResults” of “JAVA_BioAPI_Identify” or “JAVA_BioAPI_IdentifyMatch” is stored.

Instances are stored in order of decreasing authentication score (similarity level) from index 0.

- For field “Type”, “JAVA_BioAPI_ARRAY_TYPE” of “PalmSecureConstant” class is always set.
- For field "BIRInArray", index of enrollment data is set.
The index shows position of "JAVA_BioAPI_BIR" array where the enrollment data is stored.
- For field “FARAchieved”, a value is set when using “authentication score notification function”. Score value against similar enrollment data is set in a range between 1,000 and 10,000, by 1,000 units. The larger the value, the more the similarity.

>See>　As for authentication result score notification function, refer to the "Authentication Library Reference Guide" or "3.2.2.15 JAVA_PvAPI_SetProfile [JAVA_uint32 Type setting value]".



Appendix B Compatibility of Palm Vein Data

Since the data structure is different, it is not possible to use the palm vein data created by Old Interface library for this Interface module as it is.

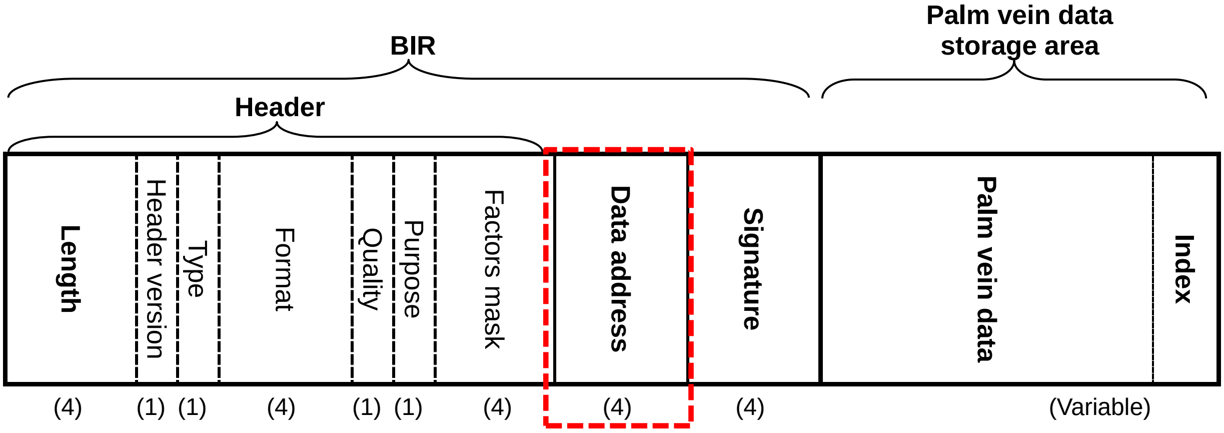
The following shows the difference between the data structure of this Interface module and Old Interface library.

Numbers in "( )" indicate the size of each field (bytes).

>See> For information on palm vein data structure, refer to "Authentication Library Reference Guide".

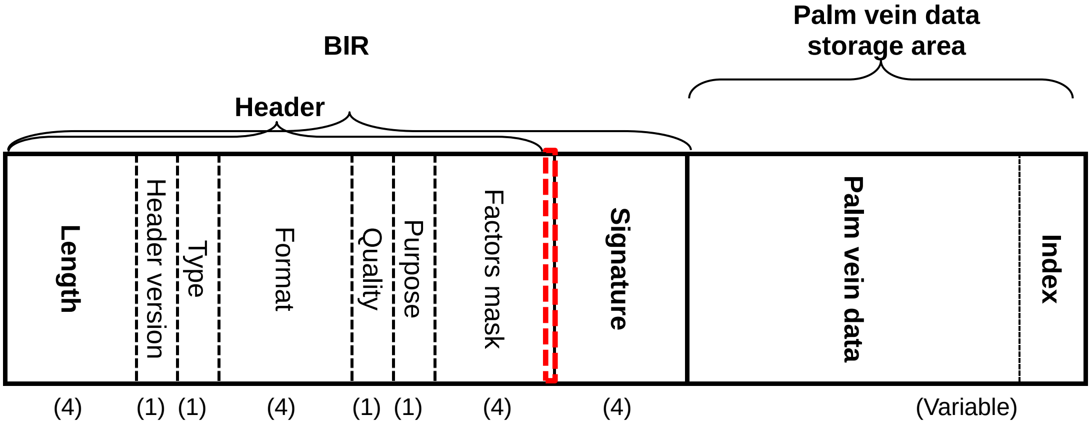
B.1 Structure of Palm Vein Data

Interface module uses the same data structure as Authentication library.



B.2 Structure of Palm Vein Data (Old Interface library)

"Data address" is omitted from the data structure of B.1.



PalmSecure™