35 ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ. МАШИНЫ КОНТОРСКИЕ

ОКС 35.040

Изменение № 1 ГОСТ Р ИСО/МЭК 19784-1—2007 Автоматическая идентификация. Идентификация биометрическая. Биометрический программный интерфейс. Часть 1. Спецификация биометрического программного интерфейса

Утверждено и введено в действие Приказом Федерального агентства по техническому регулированию и метрологии от 23.11.2010 № 491-ст

Дата введения 2012—07—01

Введение. Первый абзац изложить в новой редакции:

«Настоящий стандарт определяет высокоуровневую обобщенную модель биометрического распознавания, пригодную для использования в любой биометрической технологии. Настоящий стандарт не обеспечивает поддержку мультимодальной биометрии в явном виде»;

дополнить абзацами (после третьего):

«Настоящий стандарт определяет поведение инфраструктуры БиоАПИ, когда приложения и ПБУ находятся в одной системе. Другие взаимодействующие стандарты (см. 4.29) определяют разновидности поведения, которые позволяют ПБУ и графическому интерфейсу пользователя работать удаленно от системы, содержащей приложение.

(Продолжение см. с. 30)

(Продолжение Изменения № 1 к ГОСТ Р ИСО/МЭК 19784-1—2007)

Примечание — ИСО/МЭК 24708 [6], определяющий протокол межсетевого обмена биометрического программного интерфейса (ПМО БиоАПИ), является примером стандарта межсетевого обмена».

Раздел 3 дополнить ссылкой:

«IETF RFC 3987 Интернационализированные идентификаторы ресурсов (ИИР)».

Раздел 4 дополнить подразделами — 4.29 — 4.31:

«4.29 **стандарты межсетевого обмена** (interworking standards): Стандарты, которые преобразовывают действия инфраструктуры БиоАПИ (и требования для соответствия БиоАПИ) для поддержки использования коммуникационных каналов связи с целью предоставления возможности приложению взаимодействовать с удаленным ПБУ с использованием стандартизированного протокола.

4.30 **тестировать-верифицировать/тест-верификация** (test-verify/test-verification): Процесс сравнения «один к одному» тестового образца с биометрическим шаблоном при регистрации для определения соответствия тестового образца биометрическому шаблону.

4.31 **тип регистрации** (enroll type): Значение, которое указывает шаблон подопераций, выполняемых ПБУ в процессе операции регистрации».

Раздел 5 после сокращения «ИД» дополнить сокращением: «ИИР — интернационализированный идентификатор ресурса (см. RFC 3987);

(Продолжение см. с. 31)

ПМО БиоАПИ — протокол межсетевого обмена биометрического программного интерфейса».

Пункт 6.1.8 дополнить перечислением — е):

«е) предоставление приложению графической информации об активной операции регистрации, верификации или идентификации».

Пункт 6.3.6. Перечисления а), b), с) после обозначения ***«BioAPI_Init»*** дополнить словами: «или ***BioAPI_InitEndpoint*** (только в БиоАПИ 2.1)».

Подраздел 7.8. Заголовок изложить в новой редакции:

«7.8 Тип BioAPI_BIR_BIOMETRIC_TYPE (БиоАПИ 2.0)»;

дополнить абзацем (перед первым):

«Данный подраздел применяется только при использовании версии БиоАПИ 2.0».

Подраздел 7.14. Заголовок изложить в новой редакции:

«7.14 Тип BioAPI_BIR_SUBTYPE (БиоАПИ 2.0)»;

дополнить абзацем (перед первым):

«Данный подраздел применяется только при использовании версии БиоАПИ 2.0».

Подраздел 7.16. Заголовок изложить в новой редакции:

«7.16 Тип BioAPI_BSP_SCHEMA (БиоАПИ 2.0)»;

дополнить абзацем (перед первым):

«Данный подраздел применяется только при использовании версии БиоАПИ 2.0».

Подраздел 7.27. Примечание. Заменить слова: «Иногда невозможно определить» на «Когда используется БиоАПИ версии 2.0, иногда невозможно определить».

Пункт 7.28.2 изложить в новой редакции:

«7.28.2 Определения

BSPUuid(входной) — УУИД ПБУ, инициирующего событие;

UnitID (входной) — ИД модуля БиоАПИ, связанного с событием;

AppNotifyCallbackCtx (входной) — общий указатель на контекстную информацию, которая предоставляется при вызове функции ***BioAPI_BSPLoad,*** устанавливающей обработчик событий;

UnitSchema (входной) — указатель на схему модуля БиоАПИ, связанного с данным событием;

EventType (входной) — тип произошедшего события BioAPI_ EVENT».

Подраздел 7.31. Заголовок изложить в новой редакции:

«7.31 Тип BioAPI_GUI_BITMAP (БиоАПИ 2.0)»;

дополнить абзацем (перед первым):

«Данный подраздел применяется только при использовании версии БиоАПИ 2.0».

Подраздел 7.32. Заголовок изложить в новой редакции:

(Продолжение см. с. 32)

«7.32 Тип BioAPI_GUI_MESSAGE (БиоАПИ 2.0)»;

дополнить абзацем (перед первым):

«Данный подраздел применяется только при использовании версии стандарта БиоАПИ 2.0».

Подраздел 7.33. Заголовок изложить в новой редакции:

«7.33 Тип BioAPI_GUI_PROGRESS (БиоАПИ 2.0)»;

дополнить абзацем (перед первым):

«Данный подраздел применяется только при использовании версии БиоАПИ 2.0».

Подраздел 7.34. Заголовок изложить в новой редакции:

«7.34 Тип BioAPI_GUI_RESPONSE (БиоАПИ 2.0)»;

дополнить абзацем (перед первым):

«Данный подраздел применяется только при использовании версии БиоАПИ 2.0».

Подраздел 7.35. Заголовок изложить в новой редакции:

«7.35 Тип BioAPI_GUI_STATE (БиоАПИ 2.0)»;

дополнить абзацем (перед первым):

«Данный подраздел применяется только при использовании версии БиоАПИ 2.0».

Подраздел 7.36. Заголовок изложить в новой редакции:

«7.36 Тип BioAPI_GUI_STATE_CALLBACK (БиоАПИ 2.0)»;

дополнить абзацем (перед первым):

«Данный подраздел применяется только при использовании версии БиоАПИ 2.0».

Подраздел 7.37. Заголовок изложить в новой редакции:

«7.37 Тип BioAPI_GUI_STREAMING_CALLBACK (БиоАПИ 2.0)»;

дополнить абзацем (перед первым):

«Данный подраздел применяется только при использовании номера версии БиоАПИ 2.0».

Подраздел 7.38 дополнить абзацами:

«При использовании БиоАПИ версии 2.1 нулевое значение не должно использоваться в качестве действительного значения дескриптора присоединенной сессии ПБУ. Все остальные 32-битовые беззнаковые целочисленные значения разрешены.

#define BioAPI_INVALID_BSP_HANDLE (0)

Данное определение применимо только при использовании БиоАПИ версии 2.1».

Подраздел 7.46 после абзаца «#define BioAPI_SETGUICALLBACKS (0x00000002)»

дополнить абзацем:

«#define BioAPI_SUBSCRIBETOGUIEVENTS (0x00000002)»;

(Продолжение см. с. 33)

после абзаца «#defineBioAPI_CONTROLUNIT (0X00400000)»

дополнить абзацем:

«#define BioAPI_TRANSFORM (0x00800000)»;

дополнить абзацами (перед примечанием):

«Определение BioAPI_SETGUICALLBACKS применяется только при использовании версии БиоАПИ 2.0. Определение BioAPI_SUBSCRIBETOGUIEVENTS применяется только при использовании версии БиоАПИ 2.1.

Данное определение BioAPI_TRANSFORM применимо только при использовании БиоАПИ версии 2.1».

Подраздел 7.47. Заменить слова и дополнить абзацами: «Если установлено, то ПБУ предоставляет потоковые данные ГИП» на

«Если задано, то ПБУ предоставляет потоковые данные ГИП. Данное определение применяется только при использовании версии БиоАПИ 2.0.

#define BioAPI_GUI_PROGRESS_EVENTS (0x00000020)

Если задано, то ПБУ предоставляет потоковые данные ГИП. Данное определение применяется только при использовании версии БиоАПИ 2.1.

#define BioAPI_IDENTIFYINDICATOR (0x00200000)

Если задано, то ПБУ поддерживает индикацию выполнения в процессе идентификации. Данное определение применяется только при использовании версии БиоАПИ 2.1».

Подраздел 7.57 (7.57.1, 7.57.2*) изложить в новой редакции (сноску * исключить):

«7.57 Тип BioAPI_VERSION

Данный тип используется для представления версии БиоАПИ или определения ИПФ (Интерфейс Поставщика Функции), для которой реализованы компоненты или данные. Данный тип используется в функции BioAPI_Init, в заголовке ЗБИ и в схемах реестра компонентов.

Следующие два значения определены в настоящем стандарте:

(1) 32 десятеричное (20 шестнадцатеричное), соответствующее значению номера редакции 2 и значению номера поправки 0 и представляющее БиоАПИ 2.0; и

(2) 33 десятеричное (21 шестнадцатеричное), соответствующее значению номера редакции 2 и значению номера поправки 1 и представляющее БиоАПИ 2.1.

typedef uint8_tBioAPI_VERSION;

Примечание 1 — Данный тип не используется для версий продукта, которые обычно представлены строками.

Примечание 2 — Версия БиоАПИ, используемая в рамках заголовка ЗБИ, соответствует «CBEFF_patron_header_version» в ИСО/МЭК 19785-1.

(Продолжение см. с. 34)

Версия БиоАПИ является соединением значения номера редакции и значения номера поправки так, что первое шестнадцатеричное число представляет номер редакции, а второе шестнадцатеричное число — номер поправки:

BioAPI_VERSION 0xnm,

где n — номер редакции,

m — номер поправки».

Раздел 7 дополнить подразделами и пунктами — 7.58; 7.59, 7.59.1—7.59.10; 7.60, 7.60.1—7.60.4; 7.61, 7.61.1—7.61.3; 7.62; 7.63; 7.64; 7.65; 7.66; 7.67; 7.68; 7.69; 7.70; 7.71, 7.71.1—7.71.3; 7.72:

«7.58 Тип BioAPI_BIR_BIOMETRIC_TYPE (БиоАПИ 2.1)

Данный подраздел применяется только при использовании версии БиоАПИ 2.1.

Маска, которая описывает набор биометрических типов (факторов), содержащихся в ЗБИ БиоАПИ или поддерживаемых ПБУ.

```
typedef uint32_t BioAPI_BIR_BIOMETRIC_TYPE;
 #define BioAPI_NO_BIOTYPE_AVAILABLE                (0x00000000)
 #define BioAPI_TYPE_MULTIPLE_BIOMETRIC_TYPES
                                                    (0x00000001)
 #define BioAPI_TYPE_FACE                           (0x00000002)
 #define BioAPI_TYPE_VOICE                          (0x00000004)
 #define BioAPI_TYPE_FINGER                         (0x00000008)
 #define BioAPI_TYPE_IRIS                           (0x00000010)
 #define BioAPI_TYPE_RETINA                         (0x00000020)
 #define BioAPI_TYPE_HAND_GEOMETRY                  (0x00000040)
 #define BioAPI_TYPE_SIGNATURE_SIGN                 (0x00000080)
 #define BioAPI_TYPE_KEYSTROKE                      (0x00000100)
 #define BioAPI_TYPE_LIP_MOVEMENT                   (0x00000200)
 #define BioAPI_TYPE_GAIT                           (0x00001000)
 #define BioAPI_TYPE_VEIN                           (0x00002000)
 #define BioAPI_TYPE_DNA                            (0x00004000)
 #define BioAPI_TYPE_EAR                            (0x00008000)
 #define BioAPI_TYPE_FOOT                           (0x00010000)
 #define BioAPI_TYPE_SCENT                          (0x00020000)
 #define BioAPI_TYPE_OTHER                          (0x40000000)
 #define BioAPI_TYPE_PASSWORD                       (0x80000000)
```

Примечание 1 — BioAPI_TYPE_MULTIPLE_BIOMETRIC_TYPE используется для обозначения того, что биометрические образцы, содержащиеся в ББД (ЗБИ), включают в себя образцы, полученные от биометрических сканеров разных типов (например, данные отпечатков пальцев и изображения лица). Расположение индивидуальных образцов в ББД определяет владелец формата и идентифицируется значением типа формата.

(Продолжение см. с. 35)

Примечание 2 — Значение NO VALUE AVAILABLE указывается установкой нулевого значения. Данное значение должно использоваться в том случае, если для ЗБИ, которые первоначально не были созданы ПБУ БиоАПИ, а были преобразованы в ЗБИ БиоАПИ, информация о биометрическом типе недоступна в записи первоначального источника. Преобразованные ЗБИ, чьи биометрические типы не соответствуют ни одному из определенных типов, должны использовать значение OTHER.

Примечание 3 — Биометрический тип ЗБИ БиоАПИ соответствует «CBEFF_BDB_biometric_type» по ИСО/МЭК 19785-1.

Примечание 4 — Несмотря на то, что password не является биометрической характеристикой, BioAPI_TYPE_PASSWORD включен как действительный BioAPI_BIR_BIOMETRIC_TYPE для поддержки использования password: a) для разработки и тестирования, b) как дополнительный фактор аутентификации в рамках приложения БиоАПИ.

Примечание 5 — Наименования нескольких биометрических типов в версии БиоАПИ 2.1 отличается от имен в версии БиоАПИ 2.0, но без изменений в семантике. Также в версию БиоАПИ 2.1 включено несколько новых биометрических типов. Наконец, «температурные» типы присутствуют в версии БиоАПИ 2.0, но отсутствуют в версии БиоАПИ 2.1. Данные различия внесены для соответствия с ИСО/МЭК 19785-1.

Примечание 6 — Наименования следующих значений не совпадают в версии БиоАПИ 2.0 и БиоАПИ 2.1, но их семантика не изменена. Наименования в версии БиоАПИ 2.1 соответствуют приведенным в ИСО/МЭК 19785-1 (ЕСФОБД).

Наименование в версии БиоАПИ 2.0	Наименование в версии БиоАПИ 2.1	Кодировка
MULTIPLE	MULTIPLE_BIOMETRIC_TYPES	0x00000001
FACIAL_FEATURES (ЧЕРТЫ ЛИЦА)	FACE (ЛИЦО)	0x00000002
FINGERPRINT (ОТПЕЧАТОК ПАЛЬЦА)	FINGER (ПАЛЕЦ)	0x00000008
SIGNATURE_DYNAMICS (ДИНАМИКА ПОДПИСИ)	SIGNATURE_SIGN (ЗНАК ПОДПИСИ)	0x00000080
KEYSTROKE_DYNAMICS (ДИНАМИКА РАБОТЫ С КЛАВИАТУРЫ)	KEYSTROKE (РАБОТА С КЛАВИАТУРОЙ)	0x00000100

Примечание 7 — Следующие значения не присутствуют в версии БиоАПИ 2.0, но присутствуют в версии БиоАПИ 2.1 для совместимости с ИСО/МЭК 19785-1 (ЕСФОБД).

(Продолжение см. с. 36)

Добавленное значение	Кодировка
NO_BIOTYPE_AVAILABLE	0x00000000
VEIN (ВЕНА)	0x00002000
DNA (ДНК)	0x00004000
EAR (УХО)	0x00008000
FOOT (СТУПНЯ)	0x00010000
SCENT (ЗАПАХ)	0x00020000

7.59 Тип BioAPI_BIR_SUBTYPE (БиоАПИ 2.1)

7.59.1 Данный подраздел применяется только при использовании версии БиоАПИ 2.1.

7.59.2 BioAPI_BIR_SUBTYPE идентифицирует подтип в рамках типа ББД (блок биометрических данных) (определенный в BioAPI_BIR_ BIOMETRIC_TYPE). Его значения и содержание определены дескриптором типа ББД.

Примечание — В определении данного типа в версии БиоАПИ 2.1 могут быть установлены несколько битов. Данное определение соответствует ИСО/МЭК 19785-1.

```
typedef uint8_t BioAPI_BIR_SUBTYPE;
 #define BioAPI_BIR_SUBTYPE_VEIN_ONLY_MASK        (0x80)
 #define BioAPI_BIR_SUBTYPE_LEFT_MASK             (0x01)
 #define BioAPI_BIR_SUBTYPE_RIGHT_MASK            (0x02)
 #define BioAPI_BIR_SUBTYPE_THUMB                 (0x04)
 #define BioAPI_BIR_SUBTYPE_POINTERFINGER         (0x08)
 #define BioAPI_BIR_SUBTYPE_MIDDLEFINGER          (0x10)
 #define BioAPI_BIR_SUBTYPE_RINGFINGER            (0x20)
 #define BioAPI_BIR_SUBTYPE_LITTLEFINGER          (0x40)
 #define BioAPI_BIR_SUBTYPE_VEIN_PALM             (0x04)
 #define BioAPI_BIR_SUBTYPE_VEIN_BACKOFHAND       (0x08)
 #define BioAPI_BIR_SUBTYPE_VEIN_WRIST            (0x10)
 #define BioAPI_NO_SUBTYPE_AVAILABLE              (0x00)
```

7.59.3 Данная структура является битовой маской с битами, определенными в таблице, приведенной ниже. Бит 7 используется для указания интерпретации битов более низкого порядка. Позиция бита ноль (0) или один (1) всегда указывают право и лево соответственно. Если позиция бита 7 не установлена, то данные позиции битов могут применяться к любому биометрическому типу; однако позиции битов 2—6 характерны

(Продолжение см. с. 37)

для биометрических типов «палец» и «вена». Если позиция бита 7 установлена, то остальные позиции битов используются только для биометрического типа «вена».

Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0
0 (любой)	Мизинец	Безымянный	Средний	Указательный	Большой	Правый	Левый
1 (только вена)	Зарезервирован	Зарезервирован	Запястье	Тыльная сторона ладони	Ладонь	Правый	Левый

Примечание — Биометрические данные рисунка вен могут быть извлечены из изображений пальцев (Бит 7 задан равным нулю), или от запястья, ладони, или тыльной стороны ладони правой или левой руки (Бит 7 задан равным 1).

7.59.4 Может быть установлено ноль или больше битов. Абстрактное значение NO VALUE AVAILABLE указывается установкой всех битов на ноль.

Примечание — Абстрактное значение NO VALUE AVAILABLE может быть использовано ЗБИ, которые не были созданы ПБУ БиоАПИ, а были преобразованы из другого формата данных. Данное значение также может быть использовано, когда ни одно другое значение не применимо или информация недоступна.

7.59.5 Позиции битов с BioAPI_BIR_SUBTYPE_THUMB по BioAPI_BIR_SUBTYPE_LITTLEFINGER могут быть использованы для идентификации экземпляра(ов) биометрического типа, относящихся к пальцам.

Примечание — Подтип ЗБИ БиоАПИ соответствует «CBEFF_BDB_biometric_subtype» (ЕСФОБД_ББД_биометрический_подтип) в ИСО/МЭК 19785-1.

7.59.6 Если задана одна или более позиций бита с BioAPI_BIR_SUBTYPE_THUMB по BioAPI_BIR_SUBTYPE_LITTLEFINGER, то должна быть задана одна из двух позиций бита или обе позиции битов BioAPI_BIR_SUBTYPE_LEFT_MASK и BioAPI_BIR_SUBTYPE_RIGHT_MASK.

Если задан только бит в первой позиции, то значение подтипа обозначает один или более пальцев левой руки. Если задан только бит в последней позиции, то значение подтипа обозначает один или более паль-

(Продолжение см. с. 38)

цев правой руки. Если заданы биты в обеих позициях, то значение обозначает один или более пальцев обеих рук (один и тот же палец на обеих руках).

Примечание — Невозможно обозначить, к примеру, набор отпечатков пальцев, состоящий из указательного пальца левой руки и среднего пальца правой руки.

7.59.7 Если значение с набором битов нескольких пальцев присутствует в заголовке ЗБИ (или использовано в качестве входного параметра функции БиоАПИ, осуществляющей захват), то не обязательно, чтобы все указанные экземпляры пальцев действительно присутствовали в ББД ЗБИ (или были действительно захвачены). Однако рекомендуется, чтобы все обозначенные экземпляры пальцев присутствовали в ББД, кроме случая отсутствия пальца или подобных исключительных ситуаций. Таким же образом, если в поле подтипа ЗБИ обозначено несколько пальцев, то допускается, чтобы ББД ЗБИ включал данные о дополнительном пальце, если у субъекта шесть пальцев на руке.

7.59.8 Если ни одна позиция бита с BioAPI_BIR_SUBTYPE_THUMB по BioAPI_BIR_SUBTYPE_LITTLEFINGER не была задана, то позиции битов BioAPI_BIR_SUBTYPE_LEFT_MASK и BioAPI_BIR_SUBTYPE_RIGHT_MASK могут быть использованы для идентификации экземпляра биометрического типа, для которого существует один правый экземпляр и один левый экземпляр (например, радужная оболочка глаза), или идентификации биометрического типа, для которого существует лишь один экземпляр (например, лицо).

7.59.9 Данные рисунка вен, полученные из изображения одного или более пальцев, идентифицируются установкой бита в позиции 7 в ноль, установкой битов в одной или двух позициях 0 и 1, и установкой битов в одной или более позициях 3—7. Данные рисунка вен, полученные из других источников (запястье, ладонь или тыльная сторона ладони), идентифицируются установкой бита в позиции 7 в единицу, установкой битов в одной или двух позициях 0 и 1 и установкой одного из битов в позициях 3—5 соответствующим образом.

Примечание 1 — ЗБИ, которые не были созданы ПБУ БиоАПИ, но были преобразованы из другого формата данных, и для которых информация о подтипе недоступна, могут использовать значение NO VALUE AVAILABLE.

Примечание 2 — Подтип ЗБИ БиоАПИ соответствует «CBEFF_BDB_biometric_subtype» (ЕСФОБД_ББД_биометрический_подтип) в ИСО/МЭК 19785-1.

Примечание 3 — Данная структура, в первую очередь, используется в рамках заголовка ЗБИ; однако она также используется как входной параметр для функций, захватывающих биометрические данные. Значение BioAPI_NO_

(Продолжение см. с. 39)

SUBTYPE_AVAILABLE используется в заголовке ЗБИ, когда информация о подтипе недоступна. Значение BioAPI_NO_SUBTYPE_AVAILABLE также используется как параметр функции, когда приложение позволяет ЗБИ определять подтип, который должен быть захвачен.

7.59.10 Не рекомендуется использование позиции бита BioAPI_BIR_SUBTYPE_MULTIPLE, когда номер использующейся версии БиоАПИ 2.1.

7.60 Тип BioAPI_BSP_SCHEMA (БиоАПИ 2.1)

7.60.1 Данный подраздел применяется только при использовании версии БиоАПИ 2.1.

7.60.2 Данный тип включает в себя информацию о ПБУ, содержащуюся в реестре компонентов.

```
typedef struct bioapi_bsp_schema {
  BioAPI_UUID BSPUuid;
  BioAPI_STRING  BSPDescription;
  uint8_t *Path;
  BioAPI_VERSION SpecVersion;
  BioAPI_STRING ProductVersion;
  BioAPI_STRING Vendor;
  BioAPI_BIR_BIOMETRIC_DATA_FORMAT*BSPSupportedFormats;
  uint32_t NumSupportedFormats;
  BioAPI_BIR_BIOMETRIC_TYPE FactorsMask;
  BioAPI_OPERATIONS_MASK Operations;
  BioAPI_OPTIONS_MASK Options;
  BioAPI_FMR PayloadPolicy;
  uint32_t MaxPayloadSize;
  int32_t DefaultVerifyTimeout;
  int32_t DefaultIdentifyTimeout;
  int32_t DefaultCaptureTimeout;
  int32_t DefaultEnrollTimeout;
  int32_t DefaultCalibrateTimeout;
  uint32_t MaxBSPDbSize;
  uint32_t MaxIdentify;
  uint32_t MaxNumEnrollInstances;
  uint8_t *HostingEndpointIRI;
  BioAPI_UUID BSPAccessUUID;
}BioAPI_BSP_SCHEMA;
```

7.60.3 Определения

BSPUuid — УУИД ПБУ.

BSPDescription — строка с нулевым символом на конце, содержащая текстовое описание ПБУ.

(Продолжение см. с. 40)

Path — указатель на строку с нулевым символом на конце, содержащую путь к файлу, содержащему исполняемый код ПБУ, включая название файла. Путь к файлу может быть записан в виде адреса страницы URL. Символьная строка должна содержать символы, закодированные в формате UTF-8 в соответствии с ИСО/МЭК 10646 (приложение D).

Примечание — Если BioAPI_BSP_SCHEMA используется в вызове функции, компонент, получающий вызов, выделяет память для элемента схемы *Path* (путь к файлу), а вызывающий компонент освобождает память.

SpecVersion — номер редакции и номер поправки данной редакции спецификации БиоАПИ, для которой был разработан ПБУ.

ProductVersion — строка версии программного обеспечения ПБУ.

Vendor — строка с нулевым символом на конце, содержащая название изготовителя ПБУ.

BSPSupportedFormats — указатель на структуру BioAPI_BIR_BIOMETRIC_DATA_FORMAT, определяющую поддерживаемые форматы ББД.

NumSupportedFormats — число поддерживаемых форматов, содержащихся в BSPSupportedFormats.

FactorMask — маска, указывающая биометрические типы, поддерживаемые ПБУ.

Operations — маска, указывающая операции, поддерживаемые ПБУ.

Options — маска, указывающая опции, поддерживаемые ПБУ.

PayloadPolicy — пороговое значение (минимальное значение вероятности ложного совпадения), используемое для принятия решения о выдаче полезной информации после успешной верификации.

MaxPayloadSize — максимальный размер полезной информации (в байтах), которую может принять ПБУ.

DefaultVerifyTimeout — заданное по умолчанию значение времени ожидания в миллисекундах, используемое ПБУ для функции верификации ***BioAPI_Verify*** в случае, когда время ожидания не определено приложением.

DefaultIdentifyTimeout — заданное по умолчанию значение времени ожидания в миллисекундах, используемое ПБУ для функции идентификации ***BioAPI_Identify*** и ***BioAPI_IdentifyMatch*** в случае, когда время ожидания не определено приложением.

DefaultCaptureTimeout — заданное по умолчанию значение времени ожидания в миллисекундах, используемое ПБУ для функции захвата ***BioAPI_Capture*** в случае, когда время ожидания не определено приложением.

DefaultEnrollTimeout — заданное по умолчанию значение времени ожидания в миллисекундах, используемое ПБУ для функции регистрации

(Продолжение см. с. 41)

BioAPI_Enroll в случае, когда время ожидания не определено приложением.

DefaultCalibrateTimeout — заданное по умолчанию значение времени ожидания в миллисекундах, используемое ПБУ для операций калибровки датчика в случае, когда время ожидания не определено приложением.

MaxBSPDbSize — максимальный размер управляемой ПБУ базы данных ЗБИ.

Примечание 1 — Применяется только в том случае, когда ПБУ способен непосредственно управлять отдельным модулем архива.

Примечание 2 — Нулевое значение означает, что информация о размере базы данных не должна быть предоставлена по следующим трем причинам:

a) базы данных не поддерживаются;

b) существует возможность управления несколькими модулями (непосредственно или с использованием интерфейса ПБФ), каждый из которых может иметь различную максимальную длину, и информация о данном модуле будет предоставлена как часть уведомления о подключении (часть схемы модуля); или

c) поддерживается один модуль архива, но в данном параметре информация не предоставлена — она будет доступна в уведомлении о подключении.

MaxIdentify — максимальное число людей, поддерживаемых функцией идентификации. Значение FFFFFFFF указывает на отсутствие ограничения.

MaxNumEnrollInstances — максимальное число разных экземпляров, для которых ПБУ может создать контрольный шаблон в течение одной операции регистрации. Данная информация может быть полезной для приложения, использующего опцию ГИП, управляемого приложением.

HostingEndpointIRI — интернационализированный идентификатор ресурса (ИИР), идентифицирующий структуру, реестр компонентов которой содержит регистрацию ПБУ. Данный параметр должен быть проигнорирован инфраструктурами, соответствующими настоящему стандарту, и приложение должно установить его на пустой указатель. Это необходимо для поддержки взаимодействующих стандартов, которые могут определить порядок использования идентичных ПБУ, присутствующих на нескольких компьютерах, приложением, которое работает на том же или другом компьютере.

BSPAccessUUID — УУИД, уникальный в рамках области видимости приложения, с помощью которого приложение может сослаться на ПБУ вместо использования УУИД ПБУ. Данный параметр должен быть проигнорирован инфраструктурами, соответствующими настоящему стандарту, и приложение может ему задать любое значение УУИД. Это необходимо для поддержки взаимодействующих стандартов, которые могут

(Продолжение см. с. 42)

определить порядок использования идентичных ПБУ, присутствующих на нескольких компьютерах, приложением, которое работает на том же или другом компьютере.

Примечание — «BSPAccess_UUID» и «HostingendpointIRI» являются частью определения типа BioAPI_BSP_SCHEMA, но не являются частью информации о схеме ПБУ, хранящейся в реестре компонентов (см. таблицу 3).

7.60.4 Более подробное описание приведенных элементов и порядка записи информации ПБУ в реестр компонентов БиоАПИ приведено в 10.1.2 и 10.2.1.

7.61 Тип BioAPI_GUI_BITMAP (БиоАПИ 2.1)

7.61.1 Данный подраздел применяется только при использовании версии БиоАПИ 2.1.

7.61.2 Данная структура представляет собой графическое изображение для его отображения приложением.

```
typedef struct bioapi_gui_bitmap {
  BioAPI_BIR_SUBTYPE_MASK SubtypeMask;
  /* Подтип (или подтипы) образца, представленного на изображении */
  uint32_t Width;
  /* Ширина изображения в точках (число точек в горизонтальной линии) */
  uint32_t Height;
  /* Высота изображения в точках (число горизонтальных линий) */
  BioAPI_DATA Bitmap;
} BioAPI_GUI_BITMAP;
```

7.61.3 Определения

Bitmap — множество 8-битовых полутоновых точек (где «00» — черный и «FF» — белый), в котором данные считываются слева направо и сверху вниз следующим образом (таблица 1-1).

Таблица 1-1 — Формат битовой карты графического интерфейса пользователя

Позиция байта	Значение	Описание
0	Строка 0, пиксель 0	Первый пиксель первой строки
1	Строка 0, пиксель 1	Второй пиксель первой строки
...	...	...

(Продолжение см. с. 43)

Окончание табл. 1-1

Позиция байта	Значение	Описание
Ширина — 1	Строка 0, пиксель (ширина — 1)	Последний пиксель первой строки
Ширина	Строка 1, пиксель 0	Первый пиксель второй строки
...	...	...
(Ширина*Высота) — 1	Строка (ширина — 1), пиксель (высота — 1)	Последняя строка, последний пиксель

Примечание — Данная структура используется вместе с опцией ГИП, управляемого приложением. Описание BioAPI_GUI_STATE_HANDLER и BioAPI_GUI_STREAMING_HANDLER приведено в 7.71.

7.62 Тип BioAPI_GUI_ENROLL_TYPE (БиоАПИ 2.1)

Данный подраздел применяется только при использовании версии БиоАПИ 2.1.

Тип BioAPI_GUI_ENROLL_TYPE является типом регистрации ПБУ и определяется таким образом:

```
typedef uint32_t BioAPI_GUI_ENROLL_TYPE;
  #define BioAPI_GUI ENROLL_TYPE TEST_VERIFY
  (0x00000001)
  #define BioAPl_GUI ENROLL_TYPE MULTIPLE_CAPTURE
  (0x00000002)
```

Тип регистрации ПБУ указывает схему подопераций, представленных ПБУ в течение операции регистрации.

Позиция бита BioAPI_GUI ENROLL_TYPE TEST_VERIFY указывает (если задана) на то, что ПБУ поддерживает тест-верификацию при регистрации. В течение операции регистрации приложение получает обратные вызовы уведомления о событии изменения состояния ГИП для двух подопераций захвата, обратные вызовы уведомления о событии изменения состояния ГИП для подоперации обработки, обратные вызовы уведомления о событии изменения состояния ГИП для подоперации верификации — сравнения между «обработанной» ЗБИ и соответствующим шаблоном.

Позиция бита BioAPI_GUI ENROLL_TYPE TEST_VERIFY указывает (если задана) на то, что ПБУ поддерживает несколько подопераций захвата при регистрации. В течение операции регистрации приложение полу-

(Продолжение см. с. 44)

чает обратные вызовы уведомления о событии изменения состояния ГИП для нескольких подопераций захвата, за которыми следуют обратные вызовы уведомления о событии изменения состояния ГИП для подоперации создания шаблона.

Две позиции бита не могут быть заданы одновременно.

Примечание — Данное ограничение введено из-за того, что приложению для поддержки сочетания Тестировать-Верифицировать и Многократный-Захват потребуется больше информации о временных рамках подопераций для выбора разметки экрана в процессе регистрации.

7.63 Тип BioAPI_GUI_BITMAP_ARRAY (БиоАПИ 2.1)

Данный подраздел применяется только при использовании версии БиоАПИ 2.1.

Данный тип BioAPI_GUI_BITMAP_ARRAY определяется следующим образом:

```
typedef struct bioapi_gui_bitmap_array {
  uint32_t NumberOfMembers;
  BioAPI_GUI_BITMAP *Bitmaps;
  /* Указатель на массив BioAPI_GUI_BITMAPs */
} BioAPI_GUI_BITMAP_ARRAY;
```

7.64 Тип BioAPI_BIR_SUBTYPE_MASK (БиоАПИ2.1)

Данный подраздел применяется только при использовании версии БиоАПИ 2.1.

Данный тип BioAPI_GUI_SUBTYPE_MASK определяется следующим образом:

```
typedef uint32_t BioAPI_BIR_SUBTYPE_MASK;
  #define BioAPI_BIR_SUBTYPE_LEFT_BIT                    (0x0001)
  #define BioAPI_BIR_SUBTYPE_RIGHT_BIT                   (0x0002)
  #define BioAPI_BIR_SUBTYPE_LEFT_THUMB_BIT              (0x0004)
  #define BioAPI_BIR_SUBTYPE_LEFT_POINTERFINGER_BIT
                                                         (0x0008)
  #define BioAPI_BIR_SUBTYPE_LEFT_MIDDLEFINGER_BIT
                                                         (0x0010)
  #define BioAPI_BIR_SUBTYPE_LEFT_RINGFINGER_BIT
                                                         (0x0020)
  #define BioAPI_BIR_SUBTYPE_LEFT_LITTLEFINGER_BIT
                                                         (0x0040)
  #define BioAPI_BIR_SUBTYPE_RIGHT_THUMB_BIT             (0x0080)
  #define BioAPI_BIR_SUBTYPE_RIGHT_POINTERFINGER_BIT
                                                         (0x0100)
```

(Продолжение см. с. 45)

```
#define BioAPI_BIR_SUBTYPE_RIGHT_MIDDLEFINGER_BIT
                                                    (0x0200)
#define BioAPI_BIR_SUBTYPE_RIGHT_RINGFINGER_BIT
                                                    (0x0400)
#define BioAPI_BIR_SUBTYPE_RIGHT_LITTLEFINGER_BIT
                                                    (0x0800)
#define BioAPI_BIR_SUBTYPE_LEFT_VEIN_PALM_BIT
                                                    (0x00001000)
#define BioAPI_BIR_SUBTYPE_LEFT_VEIN_BACKOFHAND_BIT
                                                    (0x00002000)
#define BioAPI_BIR_SUBTYPE_LEFT_VEIN_WRIST_BIT
                                                    (0x00004000)
#define BioAPI_BIR_SUBTYPE_RIGHT_VEIN_PALM_BIT
                                                    (0x00008000)
#define BioAPI_BIR_SUBTYPE_RIGHT_VEIN_BACKOFHAND_BIT
                                                    (0x00010000)
#define BioAPI_BIR_SUBTYPE_RIGHT_VEIN_WRIST_BIT
                                                    (0x00020000)
```

Примечание 1 — Значения VEIN не присутствуют в версии БиоАПИ 2.0.

Позиции битов BioAPI_BIR_SUBTYPE_LEFT_BIT и BioAPI_BIR_SUBTYPE_RIGHT_BIT могут быть использованы для идентификации экземпляра биометрической модальности, для которой есть один «левый» экземпляр и один «правый» экземпляр.

Примечание 2 — Геометрия радужной оболочки глаза и руки является примером таких модальностей.

Другие позиции битов (с BioAPI_BIR_SUBTYPE_LEFT_THUMB_BIT по BioAPI_BIR_SUBTYPE_RIGHT_LITTLEFINGER_BIT) могут быть использованы для идентификации экземпляров биометрической модальности, касающейся пальцев.

Примечание 3 — Отпечаток пальца является примером такой модальности.

Значение ноль (позиция бита не задана) может быть использовано с биометрическими модальностями с единственным экземпляром.

Примечание 4 — Подпись является примером такой модальности.

7.65 Тип BioAPI_GUI_EVENT_SUBSCRIPTION (БиоАПИ 2.1)

Данный подраздел применяется только при использовании номера версии БиоАПИ 2.1.

(Продолжение см. с. 46)

Данный тип хранит информацию о существующей именованной подписке на события ГИП. Он идентифицирует приложение-подписчик и именованную подписку и указывает, какие типы событий ГИП находятся в области действий подписки. Данный тип предназначен для использования в функции ***BioAPI_QueryGUIEventSubscriptions.***

Именованная подписка на событие ГИП создана вызовом ***BioAPI_SubscribeToGUIEvents,*** определяющим непустой УУИД подписки на событие ГИП. Инфраструктура вызывает обработчики события, определенные именованной подписке, для уведомления о событиях ГИП, генерируемых ПБУ, которые перенаправлены на данную именованную подписку (см. 8.3.7), и для уведомления о событиях ГИП, генерируемых приложением (см. 8.3.3, 8.3.4, 8.3.5), которые направлены на данную именованную подписку.

```
typedef struct _bioapi_gui_event_subscription {
  const uint8_t *SubscriberEndpointIRI;
  BioAPI_UUID GUIEventSubscriptionUuid;
  BioAPI_BOOL GUISelectEventSubscribed;
  BioAPI_BOOL GUIStateEventSubscribed;
  BioAPI_BOOL GUIProgressEventSubscribed;
} BioAPI_GUI_EVENT_SUBSCRIPTION;
```

SubscriberEndpointIRI — ИИР — см. RFC 3987 — (первоначально предоставлен инфраструктурой), который идентифицирует приложение, создавшее именованную подписку на событие ГИП. Параметр должен быть установлен на пустой указатель, если приложение-подписчик то же, что и текущее приложение.

GUIEventSubscriptionUuid — УУИД (первоначально предоставлен приложением-подписчиком), который идентифицирует именованную подписку на событие ГИП.

GUISelectEventSubscribed — указывает на то, находятся ли события выбора ГИП в области действий подписки (адрес обратного вызова, установленный на непустой указатель, первоначально предоставлен приложением-подписчиком для обработчика события выбора ГИП).

GUIStateEventSubscribed — указывает на то, находятся ли события изменения состояния ГИП в области действий подписки (адрес обратного вызова, установленный на непустой указатель, первоначально предоставлен приложением-подписчиком для обработчика события изменения состояния ГИП).

GUIProgressEventSubscribed — определяет то, находятся ли события выполнения ГИП в области действий подписки (адрес обратного вызова, установленный на непустой указатель, первоначально предоставлен приложением-подписчиком для обработчика события выполнения ГИП).

(Продолжение см. с. 47)

7.66 Тип BioAPI_GUI_MOMENT (БиоАПИ 2.1)

Данный пункт применяется только при использовании версии БиоАПИ 2.1.

Перечень моментов, в которых:

a) событие выбора ГИП может быть сгенерировано в отношении цикла подоперации, или

b) событие изменения состояния или выполнения может быть сгенерировано в отношении подоперации.

```
typedef uint32_t BioAPI_GUI_MOMENT;
#define BioAPI_GUI_MOMENT_BEFORE_START (1)
#define BioAPI_GUI_MOMENT_DURING (2)
#define BioAPI_GUI_MOMENT_AFTER_END (3)
```

Значения BioAPI_GUI_MOMENT_BEFORE_START и BioAPI_GUI_MOMENT_AFTER_END могут появиться во всех типах уведомлений о событиях ГИП, в то время как значение BioAPI_GUI_MOMENT_DURING может появиться только в уведомлениях о событии выполнения ГИП.

Если значение BioAPI_GUI_MOMENT_BEFORE_START встречается в уведомлении о событии выбора ГИП, то это означает, что ПБУ готов запустить новый цикл подоперации. Цикл подоперации заключается в единственном выполнении последовательности подопераций, составляющих операцию (см. 7.68). Число циклов подопераций (успешных или нет), выполнение которых может быть затребовано приложением в рамках операции, не ограничено. Однако все полученные данные в течение каждого цикла должны быть отвергнуты с запуском последующего цикла, поэтому результатом операции (включая возвращаемое значение и полученные данные) всегда является результат ее последнего (или единственного) цикла подоперации (кроме случая отмены).

Если значение BioAPI_GUI_MOMENT_AFTER_END встречается в обратном вызове уведомления о событии выбора ГИП, то это означает, что текущий цикл подоперации завершен. Данное значение не указывает на то, успешно ли выполнение цикла подопераций или нет, так как информация предоставляется отдельным параметром уведомления о событии выбора ГИП, содержащего значение результата цикла подопераций.

Если значение BioAPI_GUI_MOMENT_BEFORE_START встречается в обратном вызове уведомления о событии изменения состояния ГИП, то это означает, что ПБУ готов запустить новую подоперацию в рамках текущего цикла подопераций.

Если значение BioAPI_GUI_MOMENT_AFTER_END встречается в обратном вызове уведомления о событии изменения состояния ГИП, то это означает, что текущая подоперация, выполняемая ПБУ, завершена.

(Продолжение см. с. 48)

Данное значение не указывает на то, успешно ли выполнение подоперации или нет, так как информация предоставляется отдельным параметром уведомления о событии изменения состояния ГИП, содержащего значение результата цикла подопераций.

Если значение BioAPI_GUI_MOMENT_BEFORE_START встречается в обратном вызове уведомления о событии выполнения ГИП, то это означает, что ПБУ готов запустить новую подоперацию в рамках текущего цикла подопераций. Событие выполнения ГИП с данным моментом может быть излишним, если также произведено событие изменения состояния ГИП, и может быть исключено. Однако подобное событие выполнения ГИП применимо в тех операциях (таких как идентификация), которые не производят событие изменения состояния ГИП.

Если значение BioAPI_GUI_MOMENT_DURING встречается в обратном вызове уведомления о событии выполнения ГИП, то это означает, что подоперация находится в процессе выполнения.

Если значение BioAPI_GUI_MOMENT_AFTER_END встречается в обратном вызове уведомления о событии выполнения ГИП, то это означает, что текущая подоперация, выполняемая ПБУ, завершена. Событие выполнения ГИП с данным моментом может быть излишним, если также произведено событие изменения состояния ГИП, и может быть исключено. Однако подобное событие выполнения ГИП применимо в тех операциях (таких как идентификация), которые не производят событие изменения состояния ГИП.

Примечание — Данный тип используется с опцией ГИП, управляемой приложением (см. 7.71).

7.67 Тип BioAPI_GUI_PROGRESS (БиоАПИ 2.1)

Данный подраздел применяется только при использовании версии БиоАПИ 2.1.

Значение в диапазоне от 0 до 100 означает процент выполнения (завершения) подоперации, предназначенный для отображения пользователю или оператору.

```
typedef uint8_t BioAPI_GUI_PROGRESS;
```

Примечание — Используется с опцией ГИП, управляемого приложением. См. описание BioAPI_GUI_PROGRESS_EVENT в 7.71.

7.68 Тип BioAPI_GUI_OPERATION (БиоАПИ 2.1)

Данный подраздел применяется только при использовании версии БиоАПИ 2.1.

Перечень всех функций (операций) интерфейса ПБУ, в процессе которых ПБУ может сгенерировать событие ГИП.

(Продолжение см. с. 49)

```
typedef uint8_t BioAPI_GUI_OPERATION;
  #define BioAPI_GUI_OPERATION_CAPTURE (1)
  #define BioAPI_GUI_OPERATION_PROCESS (2)
  #define BioAPI_GUI_OPERATION_CREATETEMPLATE (3)
  #define BioAPI_GUI_OPERATION_VERIFYMATCH (4)
  #define BioAPI_GUI_OPERATION_IDENTIFYMATCH (5)
  #define BioAPI_GUI_OPERATION_VERIFY (6)
  #define BioAPI_GUI_OPERATION_IDENTIFY (7)
  #define BioAPI_GUI_OPERATION_ENROLL (8)
```

Значения данного типа встречаются в уведомлениях о событии ГИП и идентифицируют тип функции интерфейса ПБУ, который выполнялся, когда ПБУ сгенерировал событие ГИП, и на который ссылается событие ГИП.

В таблице 1-2 определяется соответствие между функциями и операциями интерфейса ПБУ.

Таблица 1-2 — Функции и операции интерфейса ПБУ

Функции интерфейса ПБУ	Значение BioAPI_GUI_OPERATION	Общее наименование операции
BioSPI_Capture	BioAPI_GUI_OPERATION_ CAPTURE	Операция захвата
BioSPI_Process	BioAPI_GUI_OPERATION_ PROCESS	Операция обработки
BioSPI_Create-Template	BioAPI_GUI_OPERATION_ CREATETEMPLATE	Операция создания шаблона
BioSPI_Verify-Match	BioAPI_GUI_OPERATION_ VERIFYMATCH	Операция верификации и сопоставления
BioSPI_Identify-Match	BioAPI_GUI_OPERATION_ IDENTIFYMATCH	Операция идентификации и сопоставления
BioSPI_Verify	BioAPI_GUI_OPERATION_ VERIFY	Операция верификации

(Продолжение см. с. 50)

Окончание таблицы 1-2

Функции интерфейса ПБУ	Значение BioAPI_GUI_OPERATION	Общее наименование операции
BioSPI_Identify	BioAPI_GUI_OPERATION_ IDENTIFY	Операция идентификации
BioSPI_Enroll	BioAPI_GUI_OPERATION_ ENROLL	Операция регистрации

В таблице 1-3 определяется, какое из событий ГИП может быть сгенерировано ПБУ в процессе выполнения каждого типа операции.

Таблица 1-3 — События ГИП, генерируемые в каждой операции

Операция	События выбора ГИП	События изменения состояния ГИП	События обработки ГИП
Операция захвата	X	X	X
Операция обработки			X
Операция создания шаблона			X
Операция верификации и сопоставления			X
Операция идентификации и сопоставления			X
Операция верификации	X	X	X
Операция идентификации	X	X	X
Операция регистрации	X	X	X

7.69 Тип BioAPI_GUI_RESPONSE (БиоАПИ 2.1)

Данный подраздел применяется только при использовании версии БиоАПИ2.1

Перечень возможных действий, которые должны быть выполнены ПБУ после того как обратный вызов уведомления о событии ГИП, произведенный ПБУ, возвращает управление ПБУ. Приложение задает выходящему параметру обратного вызова (Response) значение данного типа перед возвратом от уведомления обратного вызова.

(Продолжение см. с. 51)

```
typedef uint8_t BioAPI_GUI_RESPONSE;
  #define BioAPI_GUI_RESPONSE_DEFAULT (0)
  #define BioAPI_GUI_RESPONSE_OP_COMPLETE (1)
  #define BioAPI_GUI_RESPONSE_OP_CANCEL (2)
  #define BioAPI_GUI_RESPONSE_CYCLE_START (3)
  #define BioAPI_GUI_RESPONSE_CYCLE_RESTART (4)
  #define BioAPI_GUI_RESPONSE_SUBOP_START (5)
  #define BioAPI_GUI_RESPONSE_SUBOP_NEXT (6)
  #define BioAPI_GUI_RESPONSE_PROGRESS_CONTINUE (7)
  #define BioAPI_GUI_RESPONSE_PROGRESS_ABORT (8)
  #define BioAPI_GUI_RESPONSE_RECAPTURE (9)
```

Значение BioAPI_GUI_RESPONSE_OP_COMPLETE может быть возвращено только в качестве ответа обратному вызову уведомления о событии выбора ГИП со значением момента BioAPI_GUI_MOMENT_AFTER_END. Оно означает то, что ПБУ должен считать операцию завершенной и должен инициировать вызов первоначальной функции для возвращения кода результата текущего цикла подопераций (который может быть либо BioAPI_OK, либо кодом ошибки). После данного ответа приложения в отношении той же операции обратные вызовы уведомления о событии ГИП не ожидаются.

Значение BioAPI_GUI_RESPONSE_OP_CANCEL может быть возвращено в качестве ответа на любой обратный вызов уведомления о событии ГИП. Оно означает то, что ПБУ должен отменить всю операцию и инициировать вызов первоначальной функции для возвращения кода ошибки. После данного ответа приложения в отношении той же операции обратные вызовы уведомления о событии ГИП не ожидаются.

Значение BioAPI_GUI_RESPONSE_CYCLE_START может быть возвращено только в качестве ответа обратному вызову уведомления о событии выбора ГИП со значением момента BioAPI_GUI_MOMENT_BEFORE_START. Оно означает то, что ПБУ должен запустить цикл подопераций.

Значение BioAPI_GUI_RESPONSE_CYCLE_RESTART может быть возвращено только в качестве ответа обратному вызову уведомления о событии выбора ГИП со значением момента BioAPI_GUI_ MOMENT_AFTER_END или в качестве ответа обратному вызову уведомления о событии изменения состояния или выполнения с любым значением момента. Оно означает то, что ПБУ должен отвергнуть все данные, полученные в течение текущего цикла подопераций, и должен запустить новый цикл подопераций. После данного ответа приложения от ПБУ ожидается обратный вызов уведомления о событии выбора ГИП со значением момента BioAPI_GUI_MOMENT_BEFORE_START.

(Продолжение см. с. 52)

Значение BioAPI_GUI_RESPONSE_SUBOP_START может быть возвращено только в качестве ответа обратному вызову уведомления о событии изменения состояния ГИП со значением момента BioAPI_GUI_MOMENT_BEFORE_START. Оно означает то, что ПБУ должен запустить подоперацию.

Значение BioAPI_GUI_RESPONSE_SUBOP_NEXT может быть возвращено только в качестве ответа обратному вызову уведомления о событии изменения состояния ГИП со значением момента BioAPI_GUI_MOMENT_AFTER_END. Оно означает то, что ПБУ должен либо выполнить следующую подоперацию в текущем цикле подопераций (если существуют еще подоперации, которые необходимо выполнить), либо выйти из цикла подопераций (если все подоперации цикла выполнены). После данного ответа приложения от ПБУ ожидается обратный вызов уведомления о событии изменения состояния ГИП со значением момента BioAPI_GUI_MOMENT_BEFORE_START или обратный вызов уведомления о событии выбора со значением момента BioAPI_GUI_MOMENT_AFTER_END (соответственно).

Значение BioAPI_GUI_RESPONSE_PROGRESS_CONTINUE может быть возвращено только в качестве ответа на обратный вызов уведомления о событии выполнения ГИП. Оно означает то, что ПБУ должен продолжить выполнение подоперации.

Значение BioAPI_GUI_RESPONSE_PROGRESS_ABORT может быть возвращено только в качестве ответа на обратный вызов уведомления о событии выполнения ГИП. Оно означает то, что ПБУ должен отменить подоперацию и выдать код ошибки. После данного ответа приложения от ПБУ ожидается обратный вызов уведомления о событии изменения ГИП со значением момента BioAPI_GUI_MOMENT_AFTER_END.

Значение BioAPI_GUI_RESPONSE_RECAPTURE может быть возвращено только в качестве ответа на обратный вызов уведомления о событии выполнения ГИП со значением момента BioAPI_GUI_MOMENT_AFTER_END. Оно означает то, что ПБУ должен отвергнуть один из полученных в текущем цикле подопераций образцов (возможно, отвергая и любой контрольный шаблон и любой обработанный образец, созданные из отвергаемого образца) и выполнить другую подоперацию захвата для создания нового образца. После данного ответа приложения от ПБУ ожидается обратный вызов уведомления о событии изменения состояния ГИП со значением момента BioAPI_GUI_MOMENT_BEFORE_START (для подоперации захвата с целью регистрации). По завершении подоперации захвата ПБУ должен выполнить любую другую подоперацию, необходимую для завершения текущего цикла подопераций.

(Продолжение см. с. 53)

Значение BioAPI_GUI_RESPONSE_DEFAULT должно пониматься как одно из следующих значений, которое относится к особой ситуации, при которой оно возвращено: BioAPI_GUI_RESPONSE_CYCLE_START, BioAPI_GUI_RESPONSE_SUBOP_START, BioAPI_GUI_RESPONSE_PROCESS_CONTINUE, BioAPI_GUI_RESPONSE_SUBOP_NEXT, или BioAPI_GUI_RESPONSE_OP_COMPLETE.

Если приложение возвращает не то ответное значение, которое допустимо в каждой отдельной ситуации, то ПБУ должен отменить операцию и вызвать первоначальную функцию для возвращения кода ошибки. В отношении той же операции обратные вызовы уведомления о событии ГИП не ожидаются.

В таблице 1-4 сведена информация о совместимости между типами событий ГИП, значениями момента и ответами.

Таблица 1-4 — Ответы ГИП и события ГИП

Ответ приложения	Событие выбора ГИП (перед запуском)	Событие изменения состояния ГИП (перед запуском)	Событие выполнения ГИП (перед запуском)	Событие выполнения ГИП (в процессе)	Событие выполнения ГИП (после завершения)	Событие изменения состояния ГИП (после завершения)	Событие выбора ГИП (после завершения)
BioAPI_GUI_RESPONSE_DEFAULT	X	X	X	X	X	X	X
BioAPI_GUI_RESPONSE_CYCLE_START	X						
BioAPI_GUI_RESPONSE_SUBOP_START		X					
BioAPI_GUI_RESPONSE_PROGRESS_CONTINUE			X	X	X		
BioAPI_GUI_RESPONSE_SUBOP_NEXT						X	
BioAPI_GUI_RESPONSE_OP_COMPLETE							X

(Продолжение см. с. 54)

Окончание таблицы 1-4

Ответ приложения	Событие выбора ГИП (перед запуском)	Событие изменения состояния ГИП (перед запуском)	Событие выполнения ГИП (перед запуском)	Событие выполнения ГИП (в процессе)	Событие выполнения ГИП (после завершения)	Событие изменения состояния ГИП (после завершения)	Событие выбора ГИП (после завершения)
BioAPI_GUI_RESPONSE_PROGRESS_ABORT			X	X	X		
BioAPI_GUI_RESPONSE_RECAPTURE						X	
BioAPI_GUI_RESPONSE_CYCLE_RESTART		X	X	X	X	X	X
BioAPI_GUI_RESPONSE_OP_CANCEL	X	X	X	X	X	X	X

7.70 Тип BioAPI_GUI_SUBOPERATION (БиоАПИ 2.1)

Данный подраздел применяется только при использовании версии БиоАПИ 2.1.

Перечень возможных типов подопераций, выполняемых ПБУ в процессе операции, о которых может быть сообщено приложению в уведомлении о событии ГИП.

```
typedef uint8_t BioAPI_GUI_SUBOPERATION;
  #define BioAPI_GUI_SUBOPERATION_CAPTURE (1)
  #define BioAPI_GUI_SUBOPERATION_PROCESS (2)
  #define BioAPI_GUI_SUBOPERATION_CREATETEMPLATE (3)
  #define BioAPI_GUI_SUBOPERATION_VERIFYMATCH (4)
  #define BioAPI_GUI_SUBOPERATION_IDENTIFYMATCH (5)
```

Значения данного типа встречаются в уведомлениях о событии ГИП и идентифицируют тип подоперации текущей операции, для которой сгенерировано событие ГИП.

В таблице 1-5 показаны типы подопераций (и их число), которые выполняются ПБУ в процессе каждого типа операции (по порядку). Для операций BioAPI_GUI_OPERATION_CAPTURE, BioAPI_GUI_OPERATION_VERIFY, BioAPI_GUI_OPERATION_IDENTIFY и BioAPI_GUI_OPERATION_ENROLL число подопераций соответствует полному циклу подопераций (см. 7.66), и в нем не учитываются возможные повторные захваты, которые совершаются в процессе цикла (см. 7.69).

(Продолжение см. с. 55)

Т а б л и ц а 1-5 — Подоперации, выполняемые в каждой операции

Операции	Подоперации
BioAPI_GUI_OPERATION_CAPTURE с целью верификации или идентификации	**BioAPI_GUI_SUBOPERATION_CAPTURE** (одна) **BioAPI_GUI_SUBOPERATION_PROCESS** (ни одной или одна)
BioAPI_GUI_OPERATION_CAPTURE с целью регистрации (если тип регистрации тестировать-верифицировать)	**BioAPI_GUI_SUBOPERATION_CAPTURE** (две) **BioAPI_GUI_SUBOPERATION_CREATETEMPLATE** (одна) **BioAPI_GUI_SUBOPERATION_PROCESS** (одна) **BioAPI_GUI_SUBOPERATION_VERIFYMATCH** (одна)
BioAPI_GUI_OPERATION_CAPTURE с целью регистрации (если тип регистрации многократный-захват)	**BioAPI_GUI_SUBOPERATION_CAPTURE** (несколько) **BioAPI_GUI_SUBOPERATION_CREATETEMPLATE** (одна)
BioAPI_GUI_OPERATION_PROCESS	**BioAPI_GUI_SUBOPERATION_PROCESS** (одна)
BioAPI_GUI_OPERATION_CREATET EMPLATE	**BioAPI_GUI_SUBOPERATION_CREATETEMPLATE** (одна)
BioAPI_GUI_OPERATION_VERIFYM ATCH	**BioAPI_GUI_SUBOPERATION_VERIFYMATCH** (одна)
BioAPI_GUI_OPERATION_IDENTIFY MATCH	**BioAPI_GUI_SUBOPERATION_IDENTIFYMATCH** (одна)
BioAPI_GUI_OPERATION_VERIFY	**BioAPI_GUI_SUBOPERATION_CAPTURE** (одна) **BioAPI_GUI_SUBOPERATION_PROCESS** (одна) **BioAPI_GUI_SUBOPERATION_VERIFYMATCH** (одна)

(Продолжение см. с. 56)

Окончание таблицы 1-5

Операции	Подоперации
BioAPI_GUI_OPERATION_IDENTIFY	**BioAPI_GUI_SUBOPERATION_CAPTURE** (одна) **BioAPI_GUI_SUBOPERATION_PROCESS** (одна) **BioAPI_GUI_SUBOPERATION_IDENTIFYMATCH** (одна)
BioAPI_GUI_OPERATION_ENROLL (если тип регистрации тестировать-верифицировать)	**BioAPI_GUI_SUBOPERATION_CAPTURE** (две) **BioAPI_GUI_SUBOPERATION_CREATETEMPLATE** (одна) **BioAPI_GUI_SUBOPERATION_PROCESS** (одна) **BioAPI_GUI_SUBOPERATION_VERIFYMATCH** (одна)
BioAPI_GUI_OPERATION_ENROLL (если тип регистрации многократный-захват)	**BioAPI_GUI_SUBOPERATION_CAPTURE** (несколько) **BioAPI_GUI_SUBOPERATION_CREATETEMPLATE** (одна)

(Продолжение см. с. 57)

7.71 События ГИП

Данный подраздел применяется только при использовании версии БиоАПИ 2.1.

7.71.1 BioAPI_GUI_SELECT_EVENT_HANDLER (БиоАПИ 2.1)

7.71.1.1 Функция обратного вызова

```
typedef BioAPI_RETURN (BioAPI *BioAPI_GUI_SELECT_EVENT_
HANDLER)
(const BioAPI_UUID *BSPUuid,
BioAPI_UNIT_ID UnitID,
const BioAPI_HANDLE *BSPHandle,
BioAPI_GUI_ENROLL_TYPE EnrollType,
const void *GUISelectEventHandlerCtx,
BioAPI_GUI_OPERATION Operation,
BioAPI_GUI_MOMENT Moment,
BioAPI_RETURN ResultCode,
int32_t MaxNumEnrollSamples,
BioAPI_BIR_SUBTYPE_MASK SelectableInstances,
BioAPI_BIR_SUBTYPE_MASK *SelectedInstances,
BioAPI_BIR_SUBTYPE_MASK CapturedInstances,
const uint8_t *Text,
BioAPI_GUI_RESPONSE *Response);
```

7.71.1.2 Описание

Это тип указателя на функцию для функции обработчика событий ГИП приложения, который обрабатывает обратные вызовы уведомления о событии выбора ГИП, исходящие от ПБУ через инфраструктуру БиоАПИ. Для получения уведомлений о событии выбора ГИП приложение должно зарегистрировать функцию обратного вызова типа BioAPI_GUI_SELECT_EVENT_HANDLER путем предоставления адреса обратного вызова функции вместе с адресом контекста в вызове BioAPI_SubscribeToGUIEvents (см. 8.3.8).

Инфраструктура производит вызов функции приложения данного типа каждый раз, когда получает входной обратный вызов функции BioSPI_GUI_SELECT_EVENT_HANDLER, которую объявляет для ПБУ. Параметры обратного вызова (кроме GUISelectEventHandlerCtx) должны быть заданы из параметров входящего обратного вызова с теми же именами; параметр GUISelectEventHandlerCtx должен быть задан из адреса контекста выбора ГИП, предоставленного первоначально приложением в его вызове BioAPI_SubscribeToGUIEvents; адрес обратного вызова должен быть задан из адреса обратного вызова выбора ГИП, предоставленного первоначально приложением в вызове BioAPI_SubscribeToGUIEvents.

(Продолжение см. с. 58)

ПБУ может генерировать события выбора ГИП только в процессе выполнения вызова BioAPI_Capture, BioAPI_Verify, BioAPI_Identify или BioAPI_Enroll. Событие выбора ГИП со значением момента BioAPI_GUI_MOMENT_BEFORE_START генерируется перед запуском каждого цикла подопераций, а событие выбора ГИП со значением момента BioAPI_GUI_MOMENT_AFTER_END генерируется после завершения каждого цикла подопераций (см. 7.66). Основное назначение уведомления о событии выбора ГИП, которое генерируется перед запуском цикла подопераций:

1) сообщить приложению о том, что новый цикл подопераций готов к запуску;

2) разрешить приложению выбрать экземпляр(ы) для захвата (где возможность подобного выбора поддерживается ПБУ и соответствует используемой биометрической модальности).

Основное назначение уведомления о событии выбора ГИП, которое генерируется после завершения цикла подопераций: сообщить приложению о результатах цикла подопераций. Приложение должно ответить каждому уведомлению о событии выбора ГИП путем указания на то, должен ли цикл подопераций запуститься, должна ли операция считаться полностью завершенной, должен ли запуститься новый цикл подопераций или операция должна быть отменена (см. 7.69).

Если приложение определяет один или больше экземпляров в параметре Subtype вызова первоначальной функции (BioAPI_Capture, BioAPI_Enroll, BioAPI_Verify или BioAPI_Identify), и этот параметр поддерживается ПБУ, то параметр SelectableInstances задается исходя из значения параметра Subtype.

Функция обработки события выбора ГИП и любые другие вызванные функции из этой функции не должны вызывать (непосредственно или косвенно) ни одну функцию БиоАПИ. Возврат значения, отличающегося от BioAPI_OK, приводит к немедленному возврату вызванной функции к вызывающему коду с передачей ему данного значения в качестве кода ошибки.

7.71.1.3 Параметры

BSPUuid (входной) — УУИД, идентифицирующий ПБУ, связанный с событием ГИП.

UnitID (входной) — ИД модуля датчика ПБУ, связанного с событием ГИП.

BSPHandle (входной) — дескриптор присоединенной сессии ПБУ, связанный с событием ГИП.

(Продолжение см. с. 59)

EnrollType (входной) — тип регистрации ПБУ. Данный параметр имеет значение, только если *Operation* является BioAPI_GUI_OPERATION_ENROLL.

GUISelectEventHandlerCtx (входной) — контекстный адрес, первоначально предоставленный приложением-подписчиком как часть подписки на событие ГИП.

Operation (входной) — значение, которое указывает на тип операции, для которой сгенерировано событие ГИП. Допустимы только следующие значения:

- BioAPI_GUI_OPERATION_CAPTURE;
- BioAPI_GUI_OPERATION_VERIFY;
- BioAPI_GUI_OPERATION_IDENTIFY;
- BioAPI_GUI_OPERATION_ENROLL.

Moment (входной) — значение, которое указывает на то — сгенерировано ли событие ГИП перед запуском цикла подопераций или после завершения цикла подопераций. Допустимы только следующие значения:

- BioAPI_GUI_MOMENT_BEFORE_START;
- BioAPI_GUI_MOMENT_AFTER_END.

ResultCode (входной) — данный параметр является значащим только тогда, когда значением параметра Moment является BioAPI_GUI_MOMENT_AFTER_END, в противном случае ему должно быть задано значение, равное нулю. Если параметр является значащим, то он должен содержать код результата только что завершенного цикла подопераций. Код результата может быть либо BioAPI_OK, либо код ошибки БиоАПИ.

MaxNumEnrollSamples (входной) — данный параметр является значащим только тогда, когда значением параметра Moment является BioAPI_GUI_MOMENT_BEFORE_START. Если параметр является значащим, то он должен содержать максимальное число образцов, которые ПБУ может захватить в течение цикла подопераций, который скоро запустится.

SelectableInstances (входной) — данный параметр является значащим только тогда, когда значением параметра Moment является BioAPI_GUI_MOMENT_BEFORE_START. Если параметр является значащим, то он должен содержать битовую маску, которая определяет какой экземпляр(ы) биометрической модальности может(гут) быть выбран(ы) приложением для захвата в течение цикла подопераций, который скоро запустится. Некоторые ПБУ поддерживают выбор нескольких экземпляров модальности, потому что они могут создать в единственной подоперации захвата единственную ЗБИ, содержащую данные, касающиеся нескольких экземпляров.

(Продолжение см. с. 60)

SelectedInstances (выходной) — данный параметр является значащим только тогда, когда значением параметра Moment является BioAPI_GUI_MOMENT_BEFORE_START. Если параметр является значащим, то он должен содержать битовую маску, которая определяет какой(ие) экземпляр(ы) биометрической модальности выбран(ы) приложением для захвата в течение цикла подопераций, который скоро запустится.

CapturedInstances (входной) — данный параметр является значащим только тогда, когда значением параметра Moment является BioAPI_GUI_MOMENT_AFTER_END, в противном случае ему должно быть задано значение, равное нулю. Если параметр является значащим, то он должен содержать битовую маску, которая определяет какой(ие) экземпляр(ы) успешно захвачены ПБУ в течение цикла подопераций, который только что завершился.

Text (входной) — текстовое сообщение, состоящее из строк символов, закодированных в формате UTF-8 в соответствии с ИСО/МЭК 10646, предназначенное для демонстрации пользователю или оператору. Содержание сообщения и язык, в котором оно выражено, не стандартизированы.

Response (выходной) — значение, определяющее ответ приложения на уведомление о событии выбора ГИП (см. 7.69).

Примечание — См. также С.7.

7.71.1.4 Возвращаемое значение

Значение BioAPI_RETURN указывает на то, что функция была выполнена успешно или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

7.71.1.5 Ошибки:

BioAPIERR_USER_CANCELLED

BioAPIERR_FUNCTION_FAILED

7.71.2 BioAPI_GUI_STATE_EVENT_HANDLER (БиоАПИ 2.1)

7.71.2.1 Функция обратного вызова

```
typedef BioAPI_RETURN (BioAPI *BioAPI_GUI_STATE_EVENT_
HANDLER)
(const BioAPI_UUID *BSPUuid,
BioAPI_UNIT_ID UnitID,
const BioAPI_HANDLE *BSPHandle,
const void *GUIStateEventHandlerCtx,
BioAPI_GUI_OPERATION Operation,
BioAPI_GUI_SUBOPERATION Suboperation,
```

(Продолжение см. с. 61)

BioAPI_BIR_PURPOSE Purpose,
BioAPI_GUI_MOMENT Moment,
BioAPI_RETURN ResuilCode,
int32_t EnrollSampleIndex,
const BioAPI_GUI_BITMAP_ARRAY *Bitmaps,
const uint8_t *Text,
BioAPI_GUI_RESPONSE *Response,
int32_t *EnrollSampleIndexToRecapture);

7.71.2.2 Описание

Данный указатель указывает на функцию обработчика событий ГИП приложения, которая обрабатывает обратные вызовы уведомления о событиях изменения состояния ГИП, исходящие от ПБУ через инфраструктуру БиоАПИ. Для того чтобы получить уведомления о событии изменения состояния ГИП, приложение должно зарегистрировать функцию обратного вызова типа BioAPI_GUI_STATE_EVENT_HANDLER путем предоставления адреса обратного вызова функции вместе с контекстным адресом в вызове ***BioAPI_SubscribeToGUIEvents*** (см. 8.3.8).

Инфраструктура производит обратный вызов функции приложения данного типа каждый раз, когда получает входящий обратный вызов функции типа BioSPI_GUI_STATE_EVENT_HANDLER, которую она открывает ПБУ. Параметры обратного вызова (кроме *GUIStateEventHandlerCtx*) должны быть заданы из параметров входящего обратного вызова с теми же наименованиями; параметр *GUIStateEventHandlerCtx* должен быть задан из контекстного адреса изменения состояния ГИП, предоставленного приложением в его вызове ***BioAPI_SubscribeToGUIEvents;*** адрес обратного вызова должен быть задан из адреса обратного вызова изменения состояния ГИП, предоставленного приложением в вызове ***BioAPI_SubscribeToGUIEvents.***

ПБУ может генерировать события изменения состояния ГИП только в процессе выполнения вызова ***BioAPI_Capture, BioAPI_Verify, BioAPI_Identify*** или ***BioAPI_Enroll.*** Событие изменения состояния ГИП со значением момента BioAPI_GUI_MOMENT_BEFORE_START генерируется перед запуском каждой подоперации, а событие изменения состояния ГИП со значением момента BioAPI_GUI_MOMENT_AFTER_END генерируется после завершения каждой подоперации (см. 7.66).

Основное назначение уведомления о событии изменения состояния ГИП, которое генерируется перед запуском подоперации: сообщить приложению о том, что определенная подоперация готова к запуску. Основное назначение уведомления о событии изменения состояния ГИП, кото-

(Продолжение см. с. 62)

рое генерируется после завершения подоперации: сообщить приложению о результатах подоперации.

Приложение должно ответить каждому уведомлению о событии изменения состояния ГИП путем указания на то, должна ли подоперация запуститься, должен ли цикл подопераций продолжиться следующей подоперацией, должен ли запуститься новый цикл подопераций или операция должна быть отменена (см. 7.69).

Функция обработки события выбора ГИП и любые другие вызванные функции из этой функции не должны вызывать (непосредственно или косвенно) ни одну функцию БиоАПИ.

Возврат значения, отличающегося от BioAPI_OK, приводит к немедленному возврату вызванной функции к вызывающему коду с передачей ему данного значения в качестве кода ошибки.

7.71.2.3 Параметры

BSPUuid (входной) — УУИД, идентифицирующий ПБУ, связанный с событием ГИП.

UnitID (входной) — ИД модуля датчика ПБУ, связанного с событием ГИП.

BSPHandle (входной) — дескриптор присоединенной сессии ПБУ, связанный с событием ГИП.

GUIStateEventHandlerCtx (входной) — контекстный адрес, первоначально предоставленный приложением-подписчиком как часть подписки на событие ГИП.

Operation (входной) — значение, которое указывает на тип операции, для которой сгенерировано событие ГИП. Допустимы только следующие значения:

- BioAPI_GUI_OPERATION_CAPTURE;
- BioAPI GUI_OPERATION_VERIFY;
- BioAPI_GUI_OPERATION_IDENTIFY;
- BioAPI_GUI_OPERATION_ENROLL.

Suboperation (входной) — значение, которое указывает на тип подоперации, для которой сгенерировано событие ГИП (подоперация, которая скоро запустится или только что завершилась). Подоперация должна быть совместима с операцией (см. таблицу 1-5).

Purpose (входной) — данный параметр является значащим только тогда, когда значением параметра Suboperation является BioAPI_GUI_SUBOPERATION_CAPTURE, в противном случае ему должно быть задано значение, равное нулю. Если параметр является значащим, то он должен указывать на назначение подоперации захвата (любое значение типа BioAPI_BIR_PURPOSE).

(Продолжение см. с. 63)

Moment (входной) — значение, которое указывает на то, сгенерировано ли событие ГИП перед запуском подоперации или после завершения подоперации. Допустимы только следующие значения:

- BioAPI_GUI_MOMENT_BEFORE_START;
- BioAPI_GUI_MOMENT_AFTER_END.

ResultCode (входной) — данный параметр является значащим только тогда, когда значением параметра Moment является BioAPI_GUI_MOMENT_AFTER_END, в противном случае ему должно быть задано значение, равное нулю. Если параметр является значащим, то он должен содержать код результата только что завершенного цикла подопераций. Код результата может быть либо BioAPI_OK, либо код ошибки БиоАПИ.

EnrollSampleIndex (входной) — данный параметр является значащим только тогда, когда значением параметра Suboperation является BioAPI_GUI_SUBOPERATION_CAPTURE, значением параметра Moment является BioAPI_GUI_MOMENT_BEFORE_START, и образец для регистрации скоро будет захвачен. В противном случае ему должно быть задано значение, равное нулю. Если параметр является значащим, то он должен содержать индекс (целое число от 1 и более) захватываемого для регистрации образца в подоперации захвата, которая скоро запустится.

Bitmaps (входной) — последовательность битовых изображений, содержащая изображение(я) образцов, полученных в процессе подоперации, которые предназначены для демонстрации пользователю или оператору. Данная последовательность в основном не содержит битовые изображения или содержит одно битовое изображение, но может содержать несколько битовых изображений, если у ПБУ есть возможность захватывать несколько экземпляров биометрической модальности в течение одной операции захвата.

Text (входной) — текстовое сообщение, состоящее из строк символов, закодированных в формате UTF-8 в соответствии с ИСО/МЭК 10646, предназначенное для демонстрации пользователю или оператору. Содержание сообщения и язык, в котором оно выражено, не стандартизированы.

Response (выходной) — значение, определяющее ответ приложения на уведомление о событии выбора ГИП (см. 7.69).

EnrollSampleIndexToRecapture (выходной) — данный параметр является значащим только тогда, когда значением параметра Suboperation является BioAPI_GUI_SUBOPERATION_CREATETEMPLATE, значением параметра Moment является BioAPI_GUI_MOMENT_AFTER_END, и значением параметра *Response* является BioAPI_GUI_RESPONSE_

(Продолжение см. с. 64)

RECAPTURE. Если параметр является значащим, то он должен содержать индекс (положительное целое число), полученный ранее для регистрации образца, перезахват которого требует приложение.

Пример — Если ПБУ предоставляет функции BioAPI_GUI_SELECT_EVENT_HANDLER тип регистрации многократный-захват, то в каждой операции регистрации функция может захватить три биометрических образца. ПБУ генерирует события изменения состояния перед запуском и после завершения каждой подоперации захвата (когда EnrollSampleIndex заданы значения 1, 2 и 3), за которыми следует событие изменения состояния ГИП перед запуском и после завершения подоперации создания шаблона. Если приложение получает уведомление о событии изменения состояния ГИП перед запуском подоперации создания шаблона, то приложение спрашивает у пользователя, какой из образцов должен быть перемещен. Пользователь выбирает второй образец после просмотра битовых изображений на экране. Приложение задает параметру Response значение BioAPI_GUI_RESPONSE_RECAPTURE, EnrollSampleIndexToRecapture значение 2 и выходит из функции обратного вызова. После этого ПБУ генерирует другое событие изменения состояния ГИП перед запуском и после завершения новой подоперации захвата, когда EnrollSampleIndex задано значение 2, и, наконец, другое событие изменения состояния ГИП перед запуском и после завершения новой подоперации создания шаблона.

Примечание — См. также С.7.

7.71.2.4 Возвращаемое значение

Значение BioAPI_RETURN указывает на то, что функция была выполнена успешно или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

7.71.2.5 Ошибки:

BioAPIERR_USER_CANCELLED

BioAPIERR_FUNCTION_FAILED

7.71.3 BioAPI_GUI_PROGRESS_EVENT_HANDLER (БиоАПИ 2.1)

7.71.3.1 Функция обратного вызова

```
typedef BioAPI_RETURN (BioAPI *BioAPI_GUI_PROGRESS_EVENT_
HANDLER)
(const BioAPI_UUID *BSPUuid,
BioAPI_UNIT_ID UnitID,
const BioAPI_HANDLE *BSPHandle,
const void *GUIProgressEventHandlerCtx,
BioAPI_GUI_OPERATION Operation,
```

(Продолжение см. с. 65)

BioAPI_GUI_SUBOPERATION Suboperation,
BioAPI_BIR_PURPOSE Purpose,
BioAPI_GUI_MOMENT Moment,
uint8_t SuboperationProgress,
const BioAPI_GUI_BITMAP_ARRAY *Bitmaps,
const uint8_t *Text,
BioAPI_GUI_RESPONSE *Response);

7.71.3.2 Описание

Данный указатель указывает на функцию обработчика событий ГИП приложения, которая обрабатывает обратные вызовы уведомления о событиях выполнения ГИП, исходящие от ПБУ через инфраструктуру БиоАПИ.

Для того чтобы получить уведомления о событии выполнения ГИП приложение должно зарегистрировать функцию обратного вызова типа BioAPI_GUI_PROGRESS_EVENT_HANDLER путем предоставления адреса обратного вызова функции вместе с контекстным адресом в вызове ***BioAPI_SubscribeToGUIEvents*** (см. 8.3.8).

Инфраструктура производит обратный вызов функции приложения данного типа каждый раз, когда получает входящий обратный вызов функции типа BioSPI_GUI_PROGRESS_EVENT_HANDLER, которую она открывает ПБУ. Параметры обратного вызова (кроме *GUIProgressEventHandlerCtx*) должны быть заданы из параметров входящего обратного вызова с теми же наименованиями; параметр *GUIProgressEventHandlerCtx* должен быть задан из контекстного адреса ГИП, предоставленного приложением в его вызове ***BioAPI_SubscribeToGUIEvents;*** адрес обратного вызова должен быть задан из адреса обратного вызова выполнения ГИП, предоставленного приложением в вызове ***BioAPI_SubscribeToGUIEvents.***

ПБУ может генерировать события выполнения ГИП только в процессе выполнения вызова ***BioAPI_Capture, BioAPI_Process, BioAPI_CreateTemplate, BioAPI_VerifyMatch, BioAPI_IdentifyMatch, BioAPI_Verify, BioAPI_Identify*** или ***BioAPI_Enroll.*** События могут быть сгенерированы в любое время, даже многократно в течение любой подоперации.

Основное назначение уведомления о событии выполнения ГИП: сообщить приложению о степени выполнения любой продолжительной подоперации в процентах (таких как подоперации захвата или идентификации-сопоставления) и отправить приложению потоковые данные (в виде набора битовых изображений), собранные датчиком. Один из возможных способов использования данной информации приложением заключается в демонстрации битовых изображений и индикатора выполнения пользо-

(Продолжение см. с. 66)

вателю или оператору. Приложение должно отвечать каждому уведомлению о событии выполнения ГИП путем указания на то, должна ли подоперация продолжиться или быть прекращена (см. 7.69).

Функция обработки события выбора ГИП и любые другие вызванные функции из этой функции не должны вызывать (непосредственно или косвенно) ни одну функцию БиоАПИ.

Возврат значения, отличающегося от BioAPI_OK, приводит к немедленному возврату вызванной функции к вызывающему коду с передачей ему данного значения в качестве кода ошибки.

7.71.3.3 Параметры

BSPUuid (входной) — УУИД, идентифицирующий ПБУ, связанный с событием ГИП.

UnitID (входной) — ИД модуля датчика ПБУ, связанного с событием ГИП.

BSPHandle (входной) — дескриптор присоединенной сессии ПБУ, связанный с событием ГИП.

GUIProgressEventHandlerCtx (входной) — контекстный адрес, первоначально предоставленный приложением-подписчиком как часть подписки на событие ГИП.

Operation (входной) — значение, которое указывает на тип операции, для которой сгенерировано событие ГИП.

Suboperation (входной) — значение, которое указывает на тип подоперации, для которой сгенерировано событие ГИП. Подоперация должна быть совместима с операцией (см. таблицу 1-5).

Purpose (входной) — данный параметр является значащим только тогда, когда значением параметра *Suboperation* является BioAPI_GUI_SUBOPERATION_CAPTURE, в противном случае ему должно быть задано значение, равное нулю. Если параметр является значащим, то он должен указывать на назначение подоперации захвата (любое значение типа BioAPI_BIR_PURPOSE).

Moment (входной) — значение, которое указывает на то, сгенерировано ли событие ГИП перед запуском подоперации, в течение подоперации или после завершения подоперации.

SuboperationProgress (входной) — степень выполнения подоперации в процентах.

Bitmaps (входной) — последовательность битовых изображений, содержащая изображение(я) образцов, полученных в процессе подоперации, которые предназначены для демонстрации пользователю или оператору. Данная последовательность в основном не содержит битовые изображения или содержит одно битовое изображение, но может содержать несколько битовых изображений, если у ПБУ есть возможность захваты-

(Продолжение см. с. 67)

вать несколько экземпляров биометрической модальности в течение одной операции захвата.

Text (входной) — текстовое сообщение, состоящее из строк символов, закодированных в формате UTF-8 в соответствии с ИСО/МЭК 10646, предназначенное для демонстрации пользователю или оператору. Содержание сообщения и язык, в котором оно выражено, не стандартизированы.

Response (выходной) — значение, определяющее ответ приложения на уведомление о событии выбора ГИП (см. 7.69).

Примечание — См. также С.7.

7.71.3.4 Возвращаемое значение

Значение BioAPI_RETURN указывает на то, что функция была выполнена успешно, или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

7.71.3.5 Ошибки:

BioAPIERR_USER_CANCELLED

BioAPIERR_FUNCTION_FAILED

7.72 BioAPI_ERROR_INFO (БиоАПИ 2.1)

Данный подраздел применяется только при использовании БиоАПИ версии 2.1.

Данная структура используется функцией ***BioAPI_GetLastErrorInfo*** и содержит информацию о последнем типе ошибки, инициировавшем функцию БиоАПИ возвратить код ошибки:

```
typedef struct bioapi_error_info {
  int32_t ErrorCode;
  BioAPI_STRING ErrorMessage;
  BioAPI_STRING ErrorSourceName;
} BioAPI_ERROR_INFO;
```

Значением элемента *ErrorCode* является то же самое, что было возвращено последним вызовом функции БиоАПИ, возвратившим код ошибки. Значения остальных элементов не стандартизированы».

Подпункт 8.1.1.1. Второй абзац изложить в новой редакции:

«Любой вызов функции ***BioAPI_Init,*** если предыдущий вызов функции ***BioAPI_Init*** или ***BioAPI_InitEndpoint*** (только в БиоАПИ 2.1) не был закончен соответствующим вызовом функции ***BioAPI_Terminate,*** будет обработан следующим образом: инфраструктура БиоАПИ выдаст ответ BioAPI_OK (только в том случае, если помер версии инфраструктуры совместим с номером версии, который был предоставлен предыдущим вызовом функции ***BioAPI_Init*** или ***BioAPI_InitEndpoint)*** или

(Продолжение см. с. 68)

BioAPI_ERR_INCOMPATIBLE_VERSION, но не будет инициализирована повторно. Счетчик числа удачных вызовов ***BioAPI_Init*** или ***BioAPI_ InitEndpoint*** будет сохраняться инфраструктурой и не завершится до тех пор, пока число соответствующих вызовов ***BioAPI_Terminate*** не достигнет определенного значения».

Подпункт 8.1.2.1. Второй абзац изложить в новой редакции:

«Данная функция может быть вызвана только в том случае, если функция ***BioAPI_Init*** или ***BioAPI_InitEndpoint*** (только в БиоАПИ 2.1), для которой еще не была вызвана соответствующая функция, ранее была вызвана успешно, по крайней мере, один раз. Функция ***BioAPI_Terminate*** не должна вызываться до успешного выполнения функции ***BioAPI_Init*** или ***BioAPI_InitEndpoint*** (например, вызовы, которые не были закончены связанными вызовами ***BioAPI_Terminate***)».

Подпункт 8.1.4.1. Заменить абзац: «Данная функция может быть вызвана только в том случае, если был сделан, по крайней мере, один вызов функции ***BioAPI_Init,*** для которого еще не был сделан соответствующий вызов функции ***BioAPI_Terminate***» на «Данная функция может быть вызвана только в том случае, если был сделан, по крайней мере, один вызов функции ***BioAPI_Init*** или ***BioAPI_InitEndpoint*** (только в БиоАПИ 2.1), для которого еще не было произведено соответствующего вызова функции ***BioAPI_Terminate***».

Подпункт 8.1.5.1. Первый абзац после слов «разрешение всех событий» дополнить словами: «,кроме тех, которые были деактивированы последним вызовом ***BioAPI_EncibleEventNotifications*** (в БиоАПИ 2.1)»;

седьмой абзац. Заменить слова: «Если ПБУ загружен путем вызова функции ***BioAPI_BSPLoad***, он должен для каждого существующего модуля БиоАПИ немедленно вызвать событие «установка» на «Когда инфраструктура БиоАПИ вызывает функцию ***BioSPI_BSPLoad,*** она для каждого существующего модуля БиоАПИ от ПБУ получает уведомление о событии «установка». Если биометрическое приложение предоставило в вызове функции ***BioAPI_BSPLoad*** обработчик события и не деактивировало уведомления о событии «установка», то инфраструктура, в свою очередь, произведет обратный вызов обработчика событий приложения. Если используется БиоАПИ версии 2.1, то уведомления о событии установки могут быть деактивированы при помощи вызова функции ***BioAPI_EnableEventNotiflcations*** (см. примечание в 7.27)»;

предпоследний абзац изложить в новой редакции:

«Данная функция может быть вызвана только в том случае, если был выполнен, по крайней мере, один вызов функции ***BioAPI_Init*** или ***BioAPI_ InitEndpoint*** (только в БиоАПИ 2.1), для которого еще не было произведено соответствующего вызова функции ***BioAPI_Terminate.*** Функция

(Продолжение см. с. 69)

BioAPI_BSPAttach может быть вызвана несколько раз после выполнения функции ***BioAPI_BSPLoad***».

Подпункт 8.1.7.1 дополнить абзацами и примечанием:

«При использовании БиоАПИ версии 2.1, если модуль БиоАПИ, используемый в присоединенной сессии, становится недействительным, то и присоединенная сессия должна быть также признана недействительной. Когда присоединенная сессия признана недействительной, то единственными функциями, определяющими дескриптор данной присоединенной сессии, которые можно вызвать, являются функции ***BioAPI_ BSPDetach*** и ***BioAPI_GetBIRFromHandle***. Все остальные функции должны возвращать BioAPIERR_INVALID_ATTACH_SESSION.

Если ПБУ поддерживает события, то он может создать событие «отключения», когда модуль БиоАПИ признан недействительным. При использовании БиоАПИ версии 2.1, если приложение установило обработчик события и не деактивировало уведомления о событии «отключение», то оно получает уведомление о событии «отключение» и может сделать вывод, что все присоединенные сессии (если существуют), использующие отключенный модуль БиоАПИ, на текущий момент недействительны. В дальнейшем приложение может либо удалить эти присоединенные сессии с помощью вызова ***BioAPI_BSPDetach,*** либо быть готовым к получению ошибки BioAPIERR_INVALID_ATTACH_SESSION от последующего вызова функции, а потом вызвать ***BioAPI_BSPDetach.***

Примечание — Если модуль БиоАПИ был косвенно выбран для присоединенной сессии (либо посредством использования функции ***BioAPI_DONT_CARE,*** либо посредством невключения категории модуля в список, предоставленный в качестве входного параметра ***BioAPI_BSPAttach***), то приложение может не знать, какие присоединенные сессии использовали отключенный модуль БиоАПИ и, таким образом, не иметь возможности удалить эти присоединенные сессии без предварительного получения ошибок BioAPIERR_INVALID_ATTACH_SESSION при последующих вызовах функций».

Подпункт 8.1.7.2. Последний абзац. Заменить обозначение: «BioAPIERR_FRAMEWORK_INVALID_BSP_HANDLE» на «BioAPI_ INVALID_BSP_HANDLE».

Подпункт 8.1.10.1. Заменить абзац: «Данная функция может быть вызвана только в том случае, если был сделан, по крайней мере, один вызов функции ***BioAPI_Init,*** для которого еще не был сделан соответствующий вызов функции ***BioAPI_Terminate***» на «Данная функция может быть вызвана только в том случае, если был сделан, по крайней мере, один вызов функции ***BioAPI_Init*** или ***BioAPI_InitEndpoint*** (только в БиоАПИ 2.1), для которого еще не был сделан соответствующий вызов функции ***BioAPI_Terminate***».

(Продолжение см. с. 70)

Подраздел 8.1 дополнить пунктами — 8.1.13—8.1.18:

«8.1.13 Функция BioAPI_Control (БиоАПИ 2.1)

Данный пункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioAPI_Control
  (BioAPI_HANDLE BSPHandle,
  BioAPI_UNIT_ID UnitID,
  const BioAPI_UUID *ControlCode,
  const BioAPI_DATA *InputData,
  BioAPI_DATA *OutputData);
```

8.1.13.1 Описание

Данная функция посылает управляющие данные от приложения модулю БиоАПИ и получает обратно данные состояния или рабочие данные. Содержание параметра *ControlCode* посылаемых (входных) данных и получаемых (выходных) данных должно быть определено в спецификации на интерфейс для данного модуля БиоАПИ (или связанного ИПФ в том случае, если он присутствует).

Данная функция выделяет область памяти, достаточную для размещения выходных данных, которые должны быть возвращены приложению, заполняет блок данными и записывает в поля *Length* и *Data* структуры *OutputData* размер и адрес блока памяти (соответственно).

Область памяти, возвращенная при вызове функции БиоАПИ, должна быть освобождена приложением путем вызова функции ***BioAPI_Free*** (8.7.2).

8.1.13.2 Параметры

BSPHandle (входной) — дескриптор присоединенного ПБУ.

UnitId (входной) — ИД модуля БиоАПИ.

ControlCode (входной) — код функции в вызываемом модуле БиоАПИ.

InputData (входной) — адрес и длина буфера данных, которые должны быть посланы модулю БиоАПИ в соответствии с полученным *ControlCode*.

OutputData (выходной) — указатель на структуру BioAPI_DATA. На выходе она должна содержать адрес и длину буфера данных, содержащего данные, полученные от модуля БиоАПИ после обработки функции, указанной в *ControlCode*. Если функция не выделила область памяти, то адрес должен быть установлен на пустой указатель, а длина буфера задана равной нулю.

8.1.13.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на успешное выполнение функции или определяет тип ошибки. Значение BioAPI_OK указывает на отсутствие ошибки. Все остальные значения описывают тип ошибки.

(Продолжение см. с. 71)

8.1.13.4 Ошибки

BioAPIERR_BIOAPI_UNIT_NOT_INSERTED

BioAPIERR_INVALID_UNIT_ID

BioAPIERR_UNIT_IN_USE

BioAPIERR_INVALID_BSP_HANDLE

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

8.1.14 Функция BioAPI_Transform (БиоАПИ 2.1)

Данный пункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioAPI_Transform
  (BioAPI_HANDLE BSPHandle,
  const BioAPI_UUID *OperationUUID,
  const BioAPI_INPUT_BIR *InputBIRs,
  uint32_t NumberOfInputBIRs,
  BioAPI_BIR_HANDLE **OutputBIRs,
  uint32_t *NumberOfOutputBIRs);
```

8.1.14.1 Описание

Данная функция трансформирует одну или несколько ЗБИ, предоставленных в качестве входного параметра, в одну или несколько ЗБИ, предоставленных в качестве выходного параметра. Выполняемая трансформация определяется параметром *OperationUUID*.

Настоящий стандарт не устанавливает никаких стандартных значений для параметра *OperationUUID* и не устанавливает никаких определенных трансформаций. Предполагается, что значения *OperationUUID* и их семантика будут установлены либо поставщиком определенного ПБУ, либо в дополнительных спецификациях (таких как сведения о приложении).

Данная функция выполняет действия в следующем порядке:

a) выполняет трансформацию, определенную параметром *OperationUUID*, используя ЗБИ, предоставленные в качестве входного параметра, и создавая одну или несколько выходных ЗБИ, как требуется для определенной трансформации;

b) выделяет область памяти, достаточную для размещения массива элементов типа BioAPI_BIR_HANDLE с числом элементов, равным числу созданных выходных ЗБИ в перечислении a);

c) заполняет массив информацией о дескрипторах ЗБИ, созданных в качестве выходного параметра в перечислении a); и

d) возвращает адрес массива в параметре *OutputBIRs* и размер массива в параметре *NumberOfOutputBIRs*.

Область памяти, возвращаемая вызовом функции БиоАПИ, должна быть освобождена приложением с помощью функции ***BioAPI_Free***

(Продолжение см. с. 72)

(см. 8.7.2), а все существующие в массиве ***OutputBIRs*** дескрипторы ЗБИ должны быть удалены путем вызова функции ***BioAPI_FreeBIRHandle.***

8.1.14.2 Параметры

BSPHandle (входной) — дескриптор присоединенного ПБУ.

OperationUUID (входной) — УУИД, определяющий трансформацию, выполняемую ПБУ.

InputBIRs (входной) — массив ЗБИ (содержащий одну или несколько ЗБИ), предоставленных в качестве входного параметра трансформации.

NumberOfInputBIRs (входной) — число ЗБИ в массиве InputBIRs.

OutputBIRs (выходной) — указатель на адрес массива, состоящий из элементов типа BioAPI_BIR_HANDLE и содержащий дескрипторы выходных ЗБИ, созданные трансформацией.

NumberOfOutputBIRs (выходной) — указатель на число элементов массива OutputBIRs.

8.1.14.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на успешное выполнение функции или определяет тип ошибки. Значение BioAPI_OK указывает на отсутствие ошибки. Все остальные значения описывают тип ошибки.

8.1.14.4 Ошибки

BioAPIERR_BIOAPI_UNIT_NOT_INSERTED

BioAPIERR_UNIT_IN_USE

BioAPIERR_TRANSFORMATION_NOT_SUPPORTED

BioAPIERR_INVALID_BSP_HANDLE

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

8.1.15 Функция BioAPI_InitEndpoint (БиоАПИ 2.1)

Данный пункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioAPI_InitEndpoint
  (BioAPI_VERSION Version,
  const uint8_t *LocalEndpointIRI);
```

8.1.15.1 Описание

Данная функция выполняет те же действия, что и функция ***BioAPI_Init.***

Приложения, соответствующие настоящему стандарту, должны либо не вызывать функцию, либо (если вызывают ее) должны установить данный параметр *LocalEndpointIRI* на пустой указатель. Функция предоставлена для поддержки стандартов межсетевого обмена.

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ПБУ.

8.1.15.2 Параметры

Version (входной) — номер редакции и номер поправки спецификации БиоАПИ, с которой совместимо биометрическое приложение.

(Продолжение см. с. 73)

LocalEndpointIRI — данный параметр должен быть проигнорирован инфраструктурами, соответствующими этому разделу настоящего стандарта, и должен быть установлен на пустой указатель приложением. Он предоставлен для поддержки стандартов межсетевого обмена.

8.1.15.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на успешное выполнение функции или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

8.1.15.4 Ошибки

BioAPIERR_INCOMPATIBLE_VERSION

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

8.1.16 Функция BioAPI_LinkToEndpoint (БиоАПИ 2.1)

Данный пункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioAPI_LinkToEndpoint
  (const uint8_t *SlaveEndpointIRI);
```

8.1.16.1 Описание

Данная функция не выполняет никаких действий и предоставлена для измененных спецификаций, содержащихся в стандартах межсетевого обмена, которые определяют применение инфраструктур БиоАПИ, установленных на нескольких компьютерах, из приложения, работающего на том же или другом компьютере. В этих стандартах данная функция может быть использована для соединения приложения с инфраструктурой, работающей на другом компьютере, таким образом, предоставляя приложению доступ к ПБУ, управляемому этой же инфраструктурой.

Приложения, соответствующие настоящему стандарту, должны либо не вызывать функцию, либо (если вызывают ее) должны установить данный параметр *SlaveEndpointIRI* на пустой указатель. Функция предоставлена для поддержки стандартов межсетевого обмена.

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ПБУ.

8.1.16.2 Параметры

SlaveEndpointIRI — данный параметр должен быть проигнорирован инфраструктурами, соответствующими этому разделу настоящего стандарта, и должен быть установлен на пустой указатель приложением. Он предоставлен для поддержки стандартов межсетевого обмена.

8.1.16.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на успешное выполнение функции или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

(Продолжение см. с. 74)

8.1.16.4 Ошибки

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

8.1.17 Функция BioAPI_UnlinkFromEndpoint (БиоАПИ 2.1)

Данный пункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioAPI_UnlinkFromEndpoint
  (const uint8_t *SlaveEndpointIRI);
```

8.1.17.1 Описание

Данная функция не выполняет никаких действий и предоставлена для измененных спецификаций, содержащихся в стандартах межсетевого обмена, которые определяют применение инфраструктур БиоАПИ, установленных на нескольких компьютерах из приложения, работающего на том же или другом компьютере. В этих стандартах данная функция может быть использована для разрушения соединения между приложением и инфраструктурой.

Приложения должны либо не вызывать функцию, либо (если вызывают ее) должны установить данный параметр *SlaveEndpointIRI* на пустой указатель.

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ПБУ.

8.1.17.2 Параметры

SlaveEndpointIRI — данный параметр должен быть проигнорирован инфраструктурами, соответствующими этому разделу настоящего стандарта, и должен быть установлен на пустой указатель приложением. Он предоставлен для поддержки стандартов межсетевого обмена.

8.1.17.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на успешное выполнение функции или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

8.1.17.4 Ошибки

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

8.1.18 Функция BioAPI_EnumFrameworks (БиоАПИ 2.1)

Данный пункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioAPI_EnumFrameworks
  (BioAPI_FRAMEWORK_SCHEMA **FwSchemaArray,
   unit32_t *NumberOfElements);
```

8.1.18.1 Описание

Данная функция схожа с функцией ***BioAPI_GetFrameworkInfo,*** но позволяет получить информацию о нескольких инфраструктурах БиоАПИ

(Продолжение см. с. 75)

при использовании спецификации, предоставленной стандартом межсетевого обмена.

В спецификации, предоставленной в настоящем стандарте, данная функция возвращает информацию об одной инфраструктуре и, следовательно, аналогична ***BioAPI_GetFrameworkInfo*** (хотя и с разными параметрами).

Данная функция предоставляет информацию обо всех инфраструктурах БиоАПИ, которые на текущий момент видны приложению (только одна, за исключением использования стандарта межсетевого обмена). Данная функция выполняет действия в следующем порядке:

a) выделяет область памяти, достаточную для размещения массива элементов типа BioAPI_FRAMEWORK_SCHEMA с числом элементов, равным числу видимых инфраструктур;

b) заполняет массив схемами инфраструктур для всех видимых инфраструктур; и

c) возвращает адрес массива в параметре *FwSchemaArray* и число элементов массива в параметре *NumberOfElements.*

Данная функция должна вызываться только в том случае, если был произведен хотя бы один вызов ***BioAPI_Init*** или ***BioAPI_InitEndpoint,*** для которого еще не было произведено соответствующего вызова ***BioAPI_Terminate.***

Данная функция обрабатывается в Инфраструктуре БиоАПИ и не передается ни одному ПБУ.

Область памяти, содержащая массив, должна быть освобождена приложением с помощью вызова функции ***BioAPI_Free*** (см. 8.7.2), в том случае если она больше не используется приложением. Области памяти, указанные элементами *Path* и *HostingEndpoint* в рамках каждого элемента массива, также должны быть освобождены приложением с помощью вызова функции ***BioAPI_Free,*** в том случае, если они больше не используются приложением.

8.1.18.2 Параметры

BFPSchemaArray (выходной) — указатель на адрес массива элементов типа BioAPI_FRAMEWORK_SCHEMA (распределенного инфраструктурой), содержащего информацию о схемах инфраструктур.

NumberOfElements (выходной) — указатель на число элементов массива (число инфраструктур, на текущий момент видимых приложению — только одна, за исключением использования стандарта межсетевого обмена).

8.1.18.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на успешное выполнение функции или определяет тип ошибки. Значение BioAPI_OK указывает на ус-

(Продолжение см. с. 76)

пешное выполнение функции. Все остальные значения описывают тип ошибки.

8.1.18.4 Ошибки

Данные об обработке ошибок БиоАПИ приведены в разделе 11».

Подпункт 8.3.1.1. Первый абзац дополнить словами: «Применение данной функции не рекомендуется при использовании БиоАПИ версии 2.1. В этом случае должна использоваться функция ***BioAPI_EnableEvent-Notifications»***.

Пункт 8.3.2. Заголовок изложить в новой редакции:

«Функция BioAPI_SetGUICallbacks (БиоАПИ 2.0)»;

дополнить абзацем (перед первым):

«Данный пункт применяется только при использовании версии БиоАПИ 2.0».

Подраздел 8.3 дополнить пунктами — 8.3.3—8.3.11:

«8.3.3 Функция BioAPI_NotifyGUIProgressEvent (БиоАПИ 2.1)

Данный пункт применяется только при использовании версии БиоАПИ 2.1.

```
BioAPI_RETURN BioAPI BioAPI_NotifyGUIProgressEvent
  (const uint8_t *SubscriberEndpointIRI,
  const BioAPI_UUID *GUIEventSubscriptionUuid,
  const BioAPI_UUID *BSPUuid,
  BioAPI_UNIT_ID UnitID,
  BioAPI_GUI_OPERATION Operation,
  BioAPI_GUI_SUBOPERATION Suboperation,
  BioAPI_BIR_PURPOSE Purpose,
  BioAPI_GUI_MOMENT Moment,
  uint8_t SuboperationProgress,
  const BioAPI_GUI_BITMAP_ARRAY *Bitmaps,
  const uint8_t *Text,
  BioAPI_GUI_RESPONSE *Response);
```

8.3.3.1 Описание

Данная функция дает возможности приложению требовать того, чтобы событие выполнения ГИП, сгенерированное приложением, было направлено в подписку определенного именованного события ГИП. Событие ГИП, переданное данной функции, называется событием ГИП, сгенерированным приложением.

При вызове данной функции инфраструктура должна просматривать все существующие именованные подписки на события ГИП, созданные приложением, идентифицированным *SubscriberEndpointIRI* (пустой указатель — текущее приложение), в поисках подписки, в которой адрес обратного вызова для события выполнения ГИП установлен на непус-

(Продолжение см. с. 77)

той указатель, а УУИД подписки на событие ГИП тот же, что и *GUIEventSubscriptionUuid*. При наличии подписки на сопоставление инфраструктура должна произвести обратный вызов обработчику события выполнения ГИП, определенного в подписке, задавая входные параметры обратного вызова из входных параметров вызова ***BioAPI_NotifyGUIProgressEvent*** с теми же наименованиями. После возврата из функции обратного вызова инфраструктура должна скопировать выходные параметры обратного вызова в выходные параметры вызова ***BioAPI_NotifyGUIProgressEvent*** с теми же наименованиями.

Если совпадающих именованных подписок на событие ГИП не существует, то функцией возвращается BioAPI_NO_GUI_EVENT_HANDLER.

Данная функция может быть вызвана только в том случае (для установленного УУИД ПБУ), если был произведен хотя бы один вызов функции ***BioAPI_BSPLoad*** (для данного УУИД ПБУ), для которого еще не был произведен соответствующий вызов ***BioAPI_BSPUnload.***

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ни одному ПБУ.

8.3.3.2 Параметры

SubscriberEndpointIRI — интернационализированный идентификатор ресурсов, идентифицирующий приложение, создавшее одну или больше подписок на событие ГИП. Параметр должен быть установлен на пустой указатель, если данное приложение является текущим приложением. Приложение может узнать об интернационализированных идентификаторах ресурсов конечной точки подписчика других приложений, использующих инфраструктуру (и создавших одну или больше именованных подписок на событие ГИП для установленного ПБУ) путем вызова ***BioAPI_QueryGUIEventSubscriptions.*** Если других приложений, использующих инфраструктуру, не существует, то ИИР конечной точки подписчика установлены на пустые указатели во всех именованных подписках, возвращенных ***BioAPI_QueryGUIEventSubscriptions.***

GUIEventSubscriptionUuid — УУИД, идентифицирующий именованную подписку на событие ГИП. Данный параметр не должен быть установлен на пустой указатель. В некоторых случаях приложение получает данный УУИД из информации, возвращенной предшествующим вызовом ***BioAPI_QueryGUIEventSubscriptions;*** в других случаях приложение предоставляет известный УУИД.

Другие параметры определяют событие выполнения ГИП, сгенерированное приложением.

8.3.3.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на то, что функция была выполнена успешно, или определяет тип ошибки. Значение BioAPI_OK

(Продолжение см. с. 78)

указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

8.3.3.4 Ошибки

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

8.3.4 Функция BioAPI_NotifyGUISelectEvent (БиоАПИ 2.1)

Данный пункт применяется только при использовании версии БиоАПИ 2.1.

```
BioAPI_RETURN BioAPI_BioAPI_NotifyGUISelectEvent
  (const uint8_t *SubscriberEndpointIRI,
  const BioAPI_UUID *GUIEventSubscriptionUuid,
  const BioAPI_UUID *BSPUuid,
  BioAPI_UNIT_ID UnitID,
  BioAPI_GUI_OPERATION Operation,
  BioAPI_GUI_MOMENT Moment,
  BioAPI_RETURN ResultCode,
  int32_t MaxNumEnrollSamples,
  BioAPI_BIR_SUBTYPE_MASK SelectableInstances,
  BioAPI_BIR_SUBTYPE_MASK *SelectedInstances,
  BioAPI_BIR_SUBTYPE_MASK CapturedInstances,
  const uint8_t *Text,
  BioAPI_GUI_RESPONSE *Response);
```

8.3.4.1 Описание

Данная функция дает возможности приложению требовать того, чтобы событие выбора ГИП, сгенерированное приложением, было привязано в подписку определенного именованного события ГИП. Событие ГИП, переданное данной функции, называется событием ГИП, сгенерированным приложением.

При вызове данной функции инфраструктура должна просматривать все существующие именованные подписки на события ГИП, созданные приложением, идентифицированным *SubscriberEndpointIRI* (пустой указатель — текущее приложение), в поисках подписки, в которой адрес обратного вызова для события выполнения ГИП установлен на непустой указатель, а УУИД подписки на событие ГИП тот же, что и *GUIEventSubscriptionUuid*. При наличии подписки на сопоставление инфраструктура должна произвести обратный вызов обработчику события выбора ГИП, определенного в подписке, задавая входные параметры обратного вызова из входных параметров вызова ***BioAPI_NotifyGUISelectEvent*** с теми же наименованиями. После возврата из функции обратного вызова инфраструктура должна скопировать выходные параметры обратного вызова в выходные параметры вызова ***BioAPI_NotifyGUISelectEvent*** с теми же наименованиями.

(Продолжение см. с. 79)

Если совпадающих именованных подписок на событие ГИП не существует, то функцией возвращается BioAPI_NO_GUI_EVENT_HANDLER.

Данная функция может быть вызвана только в том случае (для установленного УУИД ПБУ), если был произведен хотя бы один вызов функции ***BioAPI_BSPLoad*** (для данного УУИД ПБУ), для которого еще не был произведен соответствующий вызов ***BioAPI_BSPUnload.***

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ни одному ПБУ.

8.3.4.2 Параметры

SubscriberEndpointIRI (входной, дополнительный) — интернационализированный идентификатор ресурсов, идентифицирующий приложение, создавшее одну или больше подписок на событие ГИП. Параметр должен быть установлен на пустой указатель, если данное приложение является текущим приложением. Приложение может узнать об интернационализированных идентификаторах ресурсов конечной точки подписчика других приложений, использующих инфраструктуру (и создавших одну или больше именованных подписок на событие ГИП для установленного ПБУ) путем вызова ***BioAPI_QueryGUIEventSubscriptions.*** Если других приложений, использующих инфраструктуру, не существует, то ИИР конечной точки подписчика установлены на пустые указатели во всех именованных подписках, возвращенных ***BioAPI_QueryGUIEventSubscriptions.***

GUIEventSubscriptionUuid (входной) — УУИД, идентифицирующий именованную подписку на событие ГИП. Данный параметр не должен быть установлен на пустой указатель. В некоторых случаях приложение получает данный УУИД из информации, возвращенной предшествующим вызовом ***BioAPI_QueryGUIEventSubscriptions;*** в других случаях приложение предоставляет известный УУИД.

Другие параметры определяют событие выбора ГИП, сгенерированное приложением.

8.3.4.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на то, что функция была выполнена успешно, или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

8.3.4.4 Ошибки

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

8.3.5 Функция BioAPI_NotifyGUISelectEvent (БиоАПИ 2.1)

Данный пункт применяется только при использовании версии БиоАПИ 2.1.

(Продолжение см. с. 80)

```
BioAPI_RETURN BioAPI BioAPI_NotifyGUIStateEvent
  (const uint8_t *SubsriberEndpointIRI,
  const BioAPI_UUID *GUIEventSubscriptionUuid,
  const BioAPI_UUID *BSPUuid,
  BioAPI_UNIT_ID UnitID,
  BioAPI_GUI_OPERATION Operation,
  BioAPI_GUI_SUBOPERATION Suboperation,
  BioAPI_BIR_PURPOSE Purpose,
  BioAPI_GUI_MOMENT Moment,
  BioAPI_RETURN ResultCode,
  int32_t EnrollSampleIndex,
  const BioAPI_GUI_BITMAP_ARRAY *Bitmaps,
  const uint8_t *Text,
  BioAPI_GUI_RESPONSE *Response,
  int32_t *EnrollSampleIndexToRecapture);
```

8.3.5.1 Описание

Данная функция дает возможности приложению требовать того, чтобы событие изменения состояния ГИП, сгенерированное приложением, было привязано к подписке определенного именованного события ГИП. Событие ГИП, переданное данной функции, называется событием ГИП, сгенерированным приложением.

При вызове данной функции инфраструктура должна просматривать все существующие именованные подписки на события ГИП, созданные приложением, идентифицированным *SubscriberEndpointIRI* (пустой указатель — текущее приложение), в поисках подписки, в которой адрес обратного вызова для события изменения состояния ГИП установлен на непустой указатель, а УУИД подписки на событие ГИП тот же, что и *GUIEventSubscriptionUuid*. При наличии подписки на сопоставление инфраструктура должна произвести обратный вызов обработчику события изменения состояния ГИП, определенного в подписке, задавая входные параметры обратного вызова из входных параметров вызова ***BioAPI_NotifyGUIStateEvent*** с теми же наименованиями. После возврата из функции обратного вызова инфраструктура должна скопировать выходные параметры обратного вызова в выходные параметры вызова ***BioAPI_NotifyGUIStateEvent*** с теми же наименованиями.

Если совпадающих именованных подписок на событие ГИП не существует, то функцией возвращается **BioAPI_NO_GUI_EVENT_HANDLER.**

Данная функция может быть вызвана только в том случае (для установленного УУИД ПБУ), если был произведен хотя бы один вызов функции ***BioAPI_BSPLoad*** (для данного УУИД ПБУ), для которого еще не был произведен соответствующий вызов ***BioAPI_BSPUnload.***

(Продолжение см. с. 81)

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ни одному ПБУ.

8.3.5.2 Параметры

SubscriberEndpointIRI (входной, дополнительный) — интернационализированный идентификатор ресурсов, идентифицирующий приложение, создавшее одну или больше подписок на событие ГИП. Параметр должен быть установлен на пустой указатель, если данное приложение является текущим приложением. Приложение может узнать об интернационализированных идентификаторах ресурсов конечной точки подписчика других приложений, использующих инфраструктуру (и создавших одну или больше именованных подписок на событие ГИП для установленного ПБУ) путем вызова ***BioAPI_QueryGUIEventSubscriptions.*** Если других приложений, использующих инфраструктуру, не существует, то ИИР конечной точки подписчика установлены на пустые указатели во всех именованных подписках, возвращенных ***BioAPI_QueryGUIEventSubscriptions.***

GUIEventSubscriptionUuid (входной) — УУИД, идентифицирующий именованную подписку на событие ГИП. Данный параметр не должен быть установлен на пустой указатель. В некоторых случаях приложение получает данный УУИД из информации, возвращенной предшествующим вызовом ***BioAPI_QueryGUIEventSubscriptions***; в других случаях приложение предоставляет известный УУИД.

Другие параметры определяют событие изменения состояния ГИП, сгенерированное приложением.

8.3.5.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на то, что функция была выполнена успешно, или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

8.3.5.4 Ошибки

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

8.3.6 Функция BioAPI_QueryGUIEventSubscriptions (БиоАПИ 2.1)

Данный пункт применяется только при использовании версии БиоАПИ 2.1.

```
BioAPI_RETURN BioAPI BioAPI_QueryGUIEventSubscriptions
  (const BioAPI_UUID *BSPUuid,
  BioAPI_GUI_EVENT_SUBSCRIPTION**GUIEventSubscriptionList,
  uint32_t *NumberOfElements);
```

8.3.6.1 Описание

Данная функция предоставляет информацию обо всех существующих именованных подписках на события ГИП для установленного ПБУ. Именованной подпиской на событие ГИП называется та подписка, которая

(Продолжение см. с. 82)

создана путем вызова функции ***BioAPI_SubscribeToGUIEvents,*** определяющего УУИД подписки на событие ГИП, установленный на непустой указатель. Инфраструктура вызывает обработчики событий, определенные в именованной подписке, для уведомления событий ГИП, сгенерированных ПБУ, и перенаправленных в данную именованную подписку (см. 8.3.7) и для уведомления событий ГИП, сгенерированных приложением (см. 8.3.3, 8.3.4 и 8.3.5), направленных в данную именованную подписку.

Путем вызова данной функции приложение может узнать об интернационализированных идентификаторах ресурсов конечной точки подписчика других приложений, использующих инфраструктуру (где эта возможность поддерживается архитектурой реализации инфраструктуры БиоАПИ) и создавших именованные подписки на событие ГИП для установленного ПБУ. Если других приложений, использующих инфраструктуру, не существует, то ИИР конечной точки подписчика установлены на пустой указатель во всех именованных подписках, возвращенных данной функцией.

Приложение может использовать информацию, возвращенную вызовом данной функции (ИИР конечной точки подписчика, УУИД подписки на событие ГИП и флаги), в последующих вызовах функции ***BioAPI_RedirectGUIEvent*** (для требования того, чтобы определенные последующие события ГИП, сгенерированные ПБУ, перенаправить определенным именованным подпискам) или в последующих вызовах ***BioAPI_NotifyGUISelectEvent, BioAPI_NotifyGUIStateEvent*** и т. д. (для требования того, чтобы событие ГИП, сгенерированное приложением, направить определенным именованным подпискам).

Данная функция выполняет действия в следующем порядке:

a) выделяет область памяти, достаточную для размещения массива элементов типа BioAPI_GUI_EVENT_SUBSCRIPTION с числом элементов, равным числу существующих именованных подписок на событие ГИП для установленного ПБУ;

b) заполняет массив информацией об именованных подписках; и

c) возвращает адрес массива в параметре *GUIEventSubscriptionList* и число элементов массива в параметре *NumberOfElements.*

Данная функция может быть вызвана только в том случае (для установленного УУИД ПБУ), если был произведен хотя бы один вызов функции ***BioAPI_BSPLoad*** (для данного УУИД ПБУ), для которого еще не был произведен соответствующий вызов ***BioAPI_BSPUnload.***

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ни одному ПБУ.

(Продолжение см. с. 83)

Область памяти, содержащая массив, должна быть освобождена приложением с помощью вызова функции ***BioAPI_Free*** (см. 8.7.2), в том случае если он больше не нужен приложению.

8.3.6.2 Параметры

GUIEventSubscriptionList (выходной) — указатель на адрес массива элементов типа BioAPI_GUI_EVENT_SUBSCRIPTION (выделенного инфраструктурой), содержащего информацию об именованной подписке на событие ГИП.

NumberOfElements (выходной) — указатель на число элементов массива.

8.3.6.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на то, что функция была выполнена успешно, или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

8.3.6.4 Ошибки

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

8.3.7 Функция BioAPI_RedirectGUIEvents (БиоАПИ 2.1)

Данный пункт применяется только при использовании номера версии БиоАПИ 2.1.

```
BioAPI_RETURN BioAPI BioAPI_RedirectGUIEvents
  (const uint8_t *SubscriberEndpointIRI,
  const BioAPI_UUID *GUIEventSubscriptionUuid,
  BioAPI_HANDLE BSPHandle,
  BioAPI_BOOL GUISelectEventRedirected,
  BioAPI_BOOL GUIStateEventRedirected,
  BioAPI_BOOL GUIProgressEventRedirected);
```

8.3.7.1 Описание

Данная функция создает «перенаправитель событий ГИП» для текущего приложения. Перенаправители событий ГИП поддерживаются инфраструктурой во время выполнения (обычно во внутренней таблице) и дают возможность приложению передавать обработку событиями ГИП, сгенерированными ПБУ, либо другому приложению, использующему инфраструктуру (где это поддерживается архитектурой реализации инфраструктуры), либо другому компоненту того же приложения.

Приложение, вызывающее данную функцию, должно предоставить ИИР конечной точки подписчика приложения, создавшего именованную подписку на событие ГИП (пустой указатель, если это то же приложение) и УУИД подписки на событие ГИП, как и информацию, определяющую область действия перенаправителя события ГИП (какая присоединенная сессия ПБУ и какой(ие) тип(ы) события(ий) ГИП подвер-

(Продолжение см. с. 84)

гаются воздействию). Все значения параметров, предоставленные в вызове функции, должны стать частью перенаправителя события ГИП, поддерживаемого инфраструктурой.

Каждый вызов данной функции создает новый перенаправитель событий ГИП и не изменяет, а также не заменяет существующий перенаправитель. Перенаправители событий ГИП, определяющие установленный дескриптор присоединенной сессии ПБУ, должны быть автоматически удалены инфраструктурой, когда присоединенная сессия удалена. В любое время приложение может запросить инфраструктуру об удалении существующего перенаправителя с помощью вызова ***BioAPI_UnredirectGUIEvents,*** предоставляя те же параметры, что были предоставлены в соответствующем вызове ***BioAPI_RedirectGUIEvents.***

Результатом многократного вызова данной функции, в любое время, не должны становиться многократные перенаправители событий выбора ГИП, многократные перенаправители событий изменения состояния ГИП или многократные перенаправители событий выполнения ГИП, созданные для одного и того же дескриптора ПБУ. Это гарантирует определенность в способе перенаправления события для любого входящего события ГИП.

Примечание — Предполагается недопустимость двух перенаправителей с одинаковыми параметрами.

Если инфраструктура получает вызов данной функции, который успешно создал перенаправитель события ГИП, то она в ответ должна вызвать функцию ***BioSPI_SubscribeToGUIEvents*** ПБУ только в том случае, если до данного вызова уже не были созданы подписки на событие ГИП и перенаправители события ГИП. Результатом последующих вызовов ***BioAPI_RedirectGUIEvents*** не будут дальнейшие вызовы ***BioSPI_SubscribeToGUIEvent,*** только если не были удалены все существующие перенаправители и подписки для ПБУ.

Если инфраструктура получает обратный вызов уведомления о событии выбора ГИП, то она должна сначала просмотреть все существующие перенаправители событий ГИП в поисках перенаправителя, где *GUISelectEventRedirected* имеет значение ВЕРНО и тот же дескриптор ПБУ, что и в событии ГИП. Применяется один из двух абзацев, указанных ниже.

Если совпадающих перенаправителей нет, то инфраструктура должна просмотреть все анонимные подписки на событие, созданные приложением, в поисках подписки с адресом обратного вызова, установленным на непустой указатель, для события выбора ГИП и либо дескриптором ПБУ, совпадающим с дескриптором ПБУ события, либо УУИД ПБУ, совпадающим с УУИД ПБУ события. Если совпадающая подписка есть,

(Продолжение см. с. 85)

то инфраструктура должна вызвать в данной подписке обработчик события выбора ГИП, в противном случае возвращать управление из ПБУ первоначальному обратному вызову с выходными параметрами по умолчанию.

Если совпадающий перенаправитель есть, то инфраструктура должна просмотреть все существующие именованные подписки на событие ГИП, созданные приложением, идентифицированным ИИР конечной точки подписчика в перенаправителе (пустой указатель — текущее приложение), в поисках подписки, в которой адрес обратного вызова для события выбора ГИП установлен на непустой указатель, УУИД подписки на событие выбора ГИП совпадает с УУИД подписки на событие выбора ГИП в перенаправителе, и УУИД ПБУ совпадает с УУИД ПБУ события. Если совпадающая подписка есть, то инфраструктура должна произвести обратный вызов обработчика события выбора ГИП, определенного в данной подписке, в противном случае возвращать управление из ПБУ первоначальному обратному вызову с выходными параметрами по умолчанию.

В обоих случаях после возвращения из обратного вызова приложения инфраструктура должна скопировать выходные параметры обратного вызова приложения в выходные параметры обратного вызова инфраструктуры с теми же наименованиями.

Случаи, перечисленные в пяти предыдущих абзацах, одинаково применимы как к событиям изменения состояния ГИП, так и к событиям выполнения ГИП.

Данная функция может быть вызвана только в том случае (для установленного УУИД ПБУ), если был произведен хотя бы один вызов функции ***BioAPI_BSPLoad*** (для данного УУИД ПБУ), для которого еще не был произведен соответствующий вызов ***BioAPI_BSPUnload.***

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ни одному ПБУ.

8.3.7.2 Параметры

SubscriberEndpointIRI (входной, дополнительный) — ИИР, идентифицирующий приложение, создавшее именованную подписку на событие ГИП. Параметр должен быть установлен на пустой указатель, если приложение-подписчик то же, что и текущее приложение.

GUIEventSubscriptionUuid (входной) — УУИД, идентифицирующий именованную подписку на событие ГИП. В некоторых случаях приложение получает данный УУИД из информации, возвращенной предшествующим вызовом ***BioAPI_QueryGUIEventSubscriptions.*** В других случаях приложение предоставляет известный УУИД.

(Продолжение см. с. 86)

BSPHandle (входной) — дескриптор присоединенной сессии ПБУ, связанный с событием ГИП. Данный вызов не оказывает влияния на события ГИП, относящиеся к другим присоединенным сессиям ПБУ.

GUISelectEventRedirected (входной) — флаг, определяющий то, находятся ли события выбора ГИП в области действий перенаправителя. Данный вызов оказывает влияние на события выбора ГИП тогда и только тогда, когда значение этого флага ВЕРНО.

GUIStateEventRedirected (входной) — флаг, определяющий то, находятся ли события изменения состояния ГИП в области действий перенаправителя. Данный вызов оказывает влияние на события выбора ГИП тогда и только тогда, когда значение этого флага ВЕРНО.

GUIProgressEventRedirected (входной) — флаг, определяющий то, находятся ли события выполнения ГИП в области действий перенаправителя. Данный вызов оказывает влияние на события выбора ГИП тогда и только тогда, когда значение этого флага ВЕРНО.

8.3.7.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на то, что функция была выполнена успешно, или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

8.3.7.4 Ошибки

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

8.3.8 Функция BioAPI_SubscribeGUIEvents (БиоАПИ 2.1)

Данный пункт применяется только при использовании версии БиоАПИ 2.1.

```
BioAPI_RETURN BioAPI BioAPI_SubscribeToGUIEvents
  (const BioAPI_UUID *GUIEventSubscriptionUuid,
  const BioAPI_UUID *BSPUuid,
  const BioAPI_HANDLE *BSPHandle,
  BioAPI_GUI_SELECT_EVENT_HANDLER GUISelectEventHandler,
  const void *GUISelectEventHandlerCtx,
  BioAPI_GUI_STATE_EVENT_HANDLER GUIStateEventHandler,
  const void *GUIStateEventHandlerCtx,
  BioAPI_GUI_PROGRESS_EVENT_HANDLER GUIProgres Event-
Handler,
  const void *GUIProgressEventHandlerCtx);
```

8.3.8.1 Описание

Данная функция создает «подписку на событие ГИП» для текущего приложения. Подписки на события ГИП поддерживаются инфраструктурой во время выполнения (обычно как записи внутренней таблицы) и дают возможность приложению получать уведомления о событиях ГИП по определенным установленным адресам обратных вызовов.

(Продолжение см. с. 87)

Приложение должно предоставить адреса обратных вызовов от одного до трех обработчиков событий ГИП (обработчик событий выбора ГИП, обработчик события изменения состояния ГИП, обработчик события выполнения ГИП), как и информацию, определяющую область действия подписки (либо загруженный ПБУ, либо присоединенная сессия ПБУ), таким образом, устанавливая предел на уведомления о событии ГИП, которые инфраструктура направит тем обработчикам событий ГИП. Все значения параметров, предоставленные в вызове данной функции, должны стать частью подписки на событие ГИП, поддерживаемого инфраструктурой.

В любом вызове данной функции должен быть предоставлен либо УУИД ПБУ, либо дескриптор ПБУ (но не оба). Если предоставлен дескриптор ПБУ, то подписка ограничена уведомлениями о событии ГИП, которые переносят этот дескриптор ПБУ. Если предоставлен УУИД ПБУ, то подписка ограничена уведомлениями о событии ГИП, которые переносят этот УУИД ПБУ, а также могут или не могут переносить дескриптор ПБУ. (Все уведомления о событии ГИП, которые переносят дескриптор ПБУ, также переносят УУИД ПБУ, но не наоборот).

Каждый вызов данной функции создает новую подписку на событие ГИП и не изменяет, а также не заменяет существующую подписку. Подписки на событие ГИП, определяющие дескриптор присоединенной сессии, должны быть автоматически удалены инфраструктурой, когда присоединенная сессия удалена. Подписки на события ГИП, определяющие УУИД ПБУ, должны быть автоматически удалены инфраструктурой, когда ПБУ выгружен. В любое время приложение может запросить инфраструктуру об удалении существующей подписки с помощью вызова ***BioAPI_UnredirectGUIEvents,*** предоставляя те же параметры, что были предоставлены в соответствующем вызове ***BioAPI_RedirectGUIEvents.***

Предоставленные в вызове контекстные адреса должны быть возвращены инфраструктурой приложению в последующих обратных вызовах обработчиков событий. В настоящем стандарте не рассматривается наделение смыслом конкретных адресов, но он определяет их для некоторых приложений.

Результатом многократного вызова данной функции, в любое время, не должны становиться многократные обработчики событий выбора ГИП, многократные обработчики событий изменения состояния ГИП или многократные обработчики событий выполнения ГИП, созданные для того же дескриптора ПБУ или для того же УУИД ПБУ. Сюда входят и случаи, когда одна подписка определяет УУИД ПБУ, а другая подписка определяет дескриптор присоединенной сессии того же ПБУ. Для любого входящего события ГИП это гарантирует определенность в том, вызов

(Продолжение см. с. 88)

какого обработчика события ГИП (если есть) должен производиться инфраструктурой.

Примечание — Предполагается недопустимость двух подписок с одинаковыми параметрами.

Если инфраструктура получает вызов данной функции, который успешно создал обработчик события ГИП, то она в ответ должна вызвать функцию ***BioSPI_SubscribeToGUIEvents*** ПБУ только в том случае, если до данного вызова уже не были созданы подписки на событие ГИП и перенаправители событий ГИП (см. 8.3.7). Результатом последующих вызовов ***BioAPI_SubscribeGUIEvents*** не будут дальнейшие вызовы ***BioSPI_SubscribeToGUIEvent,*** только если не были удалены все существующие подписки и перенаправители для данного ПБУ. ПБУ не нужно знать, сколько подписок на события ГИП требуется приложению, определяют ли эти подписки УУИД ПБУ или дескриптор ПБУ, не нужно знать типы, адреса обратных вызовов и контекстные адреса обработчиков событий ГИП, которые определены в данных подписках.

Данная функция может быть вызвана только в том случае (для установленного УУИД ПБУ), если был произведен хотя бы один вызов функции ***BioAPI_BSPLoad*** (для данного УУИД ПБУ), для которого еще не был произведен соответствующий вызов ***BioAPI_BSPUnload.***

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ни одному ПБУ.

Примечание — См. также С.7.

8.3.8.2 Параметры

GUIEventSubscriptionUuid (входной) — идентификатор подписки. Данный параметр должен быть установлен на пустой указатель (типичный случай) для анонимной подписки на событие ГИП (касающейся получения событий ГИП, созданных ПБУ и не перенаправленных). Данный параметр должен быть установлен на непустой указатель для названной подписки (касающейся получения уведомлений о перенаправленном событии ГИП и уведомления о событии ГИП, управляемом приложением).

BSPUuid (входной, дополнительный) — УУИД, определяющий ПБУ, которым ограничена подписка.

BSPHandle (входной, дополнительный) — дескриптор присоединенной сессии ПБУ, которой ограничена подписка. Данный параметр должен быть установлен на пустой указатель, если *GUIEventSubscriptionUuid* установлен на непустой указатель (названная подписка не может определять дескриптор присоединенной сессии).

(Продолжение см. с. 89)

GUISelectEventHandler (входной, дополнительный) — адрес обратного вызова функции приложения, которая должна получать от инфраструктуры уведомления о событии выбора ГИП.

GUISelectEventHandlerCtx (входной) — контекстный адрес, который должен возвращаться инфраструктурой приложению при каждом обратном вызове обработчика события выбора ГИП.

GUIStateEventHandler (входной, дополнительный) — адрес обратного вызова функции приложения, которая должна получать от инфраструктуры уведомления о событии изменения состояния ГИП.

GUIStateEventHandlerCtx (входной) — контекстный адрес, который должен возвращаться инфраструктурой приложению при каждом обратном вызове обработчика события изменения состояния ГИП.

GUIProgressEventHandler (входной, дополнительный) — адрес обратного вызова функции приложения, которая должна получать от инфраструктуры уведомления о событии выполнения ГИП.

GUIProgressEventHandlerCtx (входной) — контекстный адрес, который должен возвращаться инфраструктурой приложению при каждом обратном вызове обработчика события выполнения ГИП.

Должен быть определен только один из параметров *BSPUuid* и *BSPHandle*. Должен быть определен по крайней мере один из параметров *GUISelectEventHandler, GUIStateEventHandler* и *GUIProgressEventHandler*. Контекстный адрес должен быть определен только в том случае, если определен соответствующий адрес обратного вызова. Не определенные параметры должны быть установлены на пустой указатель.

8.3.8.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на то, что функция была выполнена успешно, или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

8.3.8.4 Ошибки:

BioAPIERR_INVALID POINTER

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

8.3.9 Функция BioAPI_UnredirectGUIEvents (БиоАПИ 2.1)

Данный пункт применяется только при использовании версии БиоАПИ 2.1.

```
BioAPI_RETURN BioAPI BioAPI_UnredirectGUIEvents
  (const uint8_t *SubscriberEndpointIRI,
  const BioAPI_UUID *GUIEventSubscriptionUuid,
  BioAPI_HANDLE BSPHandle,
  BioAPI_BOOL GUISelectEventRedirected,
  BioAPI_BOOL GUIStateEventRedirected,
  BioAPI_BOOL GUIProgressEventRedirected);
```

(Продолжение см. с. 90)

8.3.9.1 Описание

Данная функция удаляет перенаправитель события ГИП, созданный посредством вызова ***BioAPI_RedirectGUIEvents.*** Определение именно этого удаленного перенаправителя в точности совпадает с параметрами вызова.

Примечание — Вызов данной функции не удалит перенаправитель, даже если существует лишь один перенаправитель, если значения параметров вызова полностью не совпадают со значениями параметров вызова ***BioAPI_RedirectGUIEvents,*** создавшего перенаправитель. Приложение должно помнить значения данных параметров.

Если инфраструктуре поступает вызов данной функции, который успешно удаляет перенаправитель события ГИП, то инфраструктура должна в ответ вызвать функцию ***BioSPI_UnsubscribeFromGUIEvents*** ПБУ только в том случае, если больше не существует подписок на события ГИП и перенаправителей событий ГИП, созданных для данного ПБУ.

Обычно у приложений нет необходимости вызывать данную функцию, потому что все перенаправители событий ГИП для указанной присоединенной сессии ПБУ автоматически удаляются инфраструктурой, когда удаляется присоединенная сессия. Данная функция может быть использована в том случае, если приложению необходимо модифицировать перенаправителей текущего события ГИП без удаления присоединенной сессии.

Данная функция может быть вызвана только в том случае (для установленного УУИД ПБУ), если был произведен хотя бы один вызов функции ***BioAPI_BSPLoad*** (для данного УУИД ПБУ), для которого еще не был произведен соответствующий вызов ***BioAPI_BSPUnload.***

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ни одному ПБУ.

Примечание — См. также С.7.

8.3.9.2 Параметры

Параметры данной функции имеют тот же смысл, что и параметры функции ***BioAPI_RedirectGUIEvents*** с теми же наименованиями, и используются инфраструктурой для идентификации существующего перенаправителя события ГИП, который должен быть удален.

8.3.9.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на то, что функция была выполнена успешно, или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

(Продолжение см. с. 91)

8.3.9.4 Ошибки:

BioAPIERR_INVALID_POINTER

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

8.3.10 Функция BioAPI_UnsubscribeFromGUIEvents (БиоАПИ 2.1)

Данный пункт применяется только при использовании версии БиоАПИ 2.1.

```
BioAPI_RETURN BioAPI BioAPI_UnsubscribeFromGUIEvents
  (const BioAPI_UUID *GUIEventSubscriptionUuid,
  const BioAPI_UUID *BSPUuid,
  const BioAPI_HANDLE *BSPHandle,
  BioAPI_GUI_SELECT_EVENT_HANDLER GUISelectEventHandler,
  const void *GUISelectEventHandlerCtx,
  BioAPI_GUI_STATE_EVENT_HANDLER GUIStateEventHandler,
  const void *GUIStateEventHandlerCtx,
          BioAPI_GUI_PROGRESS_EVENT_HANDLER
      GUIProgressEventHandler
  const void *GUIProgressEventHandlerCtx);
```

8.3.10.1 Описание

Данная функция удаляет подписку на событие ГИП, созданную посредством вызова ***BioAPI_SubscribeToGUIEvents.*** Удаляется именно та подписка, определение которой точно соответствует параметрам вызова.

Примечание — Вызов данной функции не удалит подписку, даже если существует лишь одна подписка, если значения параметров вызова полностью не совпадают со значениями параметров вызова ***BioAPI_SubscribeToGUIEvents,*** создавшего подписку. Приложение должно помнить значения данных параметров.

Если инфраструктуре поступает вызов данной функции, который успешно удаляет подписку на событие ГИП, то инфраструктура должна в ответ вызвать функцию ***BioSPI_UnsubscribeFromGUIEvents*** ПБУ только в том случае, если больше не существует подписок на события ГИП и перенаправителей (см. 8.3.7) событий ГИП, созданных для данного ПБУ.

Обычно у приложений нет необходимости вызывать данную функцию, потому что все подписки на события ГИП для указанной присоединенной сессии ПБУ автоматически удаляются инфраструктурой, когда удаляется присоединенная сессия. Данная функция может быть использована в том случае, если приложению необходимо модифицировать подписку на текущее событие ГИП без удаления присоединенной сессии.

(Продолжение см. с. 92)

Пример — Данная функция может быть применена в следующих случаях: (1) приложение применяет управляемый приложением ГИП как при регистрации, так и при верификации/идентификации, но с разными обработчиками событий ГИП; (2) приложение использует ГИП, управляемый ПБУ, при регистрации, но управляемый приложением ГИП, при верификации/идентификации, или наоборот.

Данная функция может быть вызвана только в том случае (для установленного УУИД ПБУ), если был произведен хотя бы один вызов функции ***BioAPI_BSPLoad*** (для данного УУИД ПБУ), для которого еще не был произведен соответствующий вызов ***BioAPI_BSPUnload.***

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ни одному ПБУ.

Примечание — См. также С.7.

8.3.10.2 Параметры

Параметры данной функции имеют тот же смысл, что и параметры функции ***BioAPI_SubscribeGUIEvents*** с теми же наименованиями, и используются инфраструктурой для идентификации существующей подписки на событие ГИП, которая должна быть удалена.

8.3.10.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на то, что функция была выполнена успешно, или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

8.3.10.4 Ошибки:

BioAPIERR_INVALID_POINTER

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

8.3.11 Функция BioAPI_EbableEventNotifications (БиоАПИ 2.1)

Данный пункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioAPI_EnableEventNotifications
  (const BioAPI_UUID *BSPUuid,
  BioAPI_EVENT_MASK Events);
```

8.3.11.1 Описание

Функция разрешает события, определенные маской событий и поступающие от ПБУ, идентифицированного УУИД ПБУ, и деактивирует все остальные события, поступающие от этого же ПБУ.

События разрешены или деактивированы только для приложения, вызывающего данную функцию. Если существуют другие приложения, использующие инфраструктуру БиоАПИ или указанный ПБУ одновременно, то на такие приложения вызов функции ***BioAPI_EnableEvent-Notification*** влияния не окажет.

(Продолжение см. с. 93)

Данная функция может быть вызвана в любое время после ***BioAPI_Init*** или ***BioAPI_InitEndpoint,*** даже до загрузки указанного ПБУ. Маска событий, созданная вызовом данной функции, остается действующей до тех пор, пока эта функция не будет снова вызвана для того же ПБУ.

Данная функция должна вызываться только в том случае, если был произведен хотя бы один вызов ***BioAPI_Init*** или ***BioAPI_InitEndpoint,*** для которого еще не был произведен соответствующий вызов ***BioAPI_ Terminate.***

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ни одному ПБУ.

8.3.11.2 Параметры

BSPUuid (входной) — УУИД ПБУ.

Events (входной) — маска, обозначающая разрешенные события.

8.3.11.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на успешное выполнение функции или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

8.3.11.4 Ошибки

Данные об обработке ошибок БиоАПИ приведены в разделе 11».

Подпункт 9.2.1.1 Первый — четвертый абзацы (определения параметров) изложить в новой редакции:

«BSPUuid (входной) — УУИД ПБУ, вызывающего событие.

UnitID (входной) — ИД модуля БиоАПИ, связанного с данным событием.

UnitSchema (входной) — указатель на модуль схемы модуля БиоАПИ, связанного с данным событием.

EventType (входной) — произошедшее событие типа BioAPI_EVENT».

Подраздел 9.2 дополнить пунктами — 9.2.4—9.2.6.5:

«9.2.4 BioSPI_GUI_PROGRESS_EVENT_HANDLER (БиоАПИ 2.1)

Данный пункт применяется только при использовании номера версии БиоАПИ 2.1.

9.2.4.1 Функция обратного вызова

```
typedef BioAPI_RETURN (BioAPI
*BioSPI_GUI_PROGRESS_EVENT_HANDLER)
  (const BioAPI_UUID *BSPUuid,
  BioAPI_UNIT_ID UnitID,
  const BioAPI_HANDLE *BSPHandle,
  BioAPI_GUI_OPERATION Operation,
  BioAPI_GUI_SUBOPERATION Suboperation,
  BioAPI_BIR_PURPOSE Purpose,
```

(Продолжение см. с. 94)

BioAPI_GUI_MOMENT Moment,
uint8_t SuboperationProgress,
const BioAPI_GUI_BITMAP_ARRAY *Bitmaps,
const uint8_t *Text,
BioAPI_GUI_RESPONSE *Response);

9.2.4.2 Описание

Данный пункт описывает тип указателя на функцию для функции обработчика события ГИП инфраструктуры, который обрабатывает обратные вызовы уведомления о событии выполнения ГИП, поступающие от ПБУ. Для того чтобы получать уведомления о событии выполнения ГИП, инфраструктура должна зарегистрировать функцию обратного вызова типа BioSPI_GUI_PROGRESS_EVENT_HANDLER путем предоставления адреса обратного вызова функции в вызове ***BioSPI_SubscribeToGUIEvents*** (см. 9.3.3.3).

ПБУ производит обратный вызов функции инфраструктуры этого типа для того, чтобы уведомить инфраструктуру (в конечном счете, приложение) о том, что создано событие выполнения ГИП в процессе выполнения функции ПБУ.

ПБУ может создавать события выполнения ГИП только в процессе выполнения вызова ***BioSPI_Capture, BioSPI_Process, BioSPI_Cre-ateTemplate, BioSPI_VerifyMatch, BioSPI_IdentifyMatch, BioSPI_Verify, BioSPI_Identify*** или ***BioSPI_Enroll.*** Они могут быть созданы в любое время, даже многократно, в процессе любой подоперации.

9.2.4.3 Параметры

Параметры данного типа указателя на функцию совпадают с параметрами BioAPI_GUI_PROGRESS_EVENT_HANDLER, кроме того, что отсутствует параметр контекстного адреса.

Примечание — См. также раздел 7.

9.2.4.4 Возвращаемое значение

Значение BioAPI_RETURN указывает на то, что функция была выполнена успешно, или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

9.2.4.5 Ошибки

BioAPIERR_USER_CANCELLED
BioAPIERR_FUNCTION_FAILED

9.2.5 BioSPI_GUI_SELECT_EVENT_HANDLER (БиоАПИ 2.1)

Данный пункт применяется только при использовании номера версии БиоАПИ 2.1.

(Продолжение см. с. 95)

9.2.5.1 Функция обратного вызова

```
typedef BioAPI_RETURN (BioAPI *BioSPI_GUI_SELECT_EVENT_
HANDLER)
  (const BioAPI_UUID *BSPUuid,
  BioAPI_UNIT_ID UnitID,
  const BioAPI_HANDLE *BSPHandle,
  BioAPI_GUI_ENROLL_TYPE EnrollType,
  BioAPI_GUI_OPERATION Operation,
  BioAPI_GUI_MOMENT Moment,
  BioAPI_RETURN ResultCode,
  int32_t MaxNumEnrollSamples,
  BioAPI_BIR_SUBTYPE_MASK SelectableInstances,
  BioAPI_BIR_SUBTYPE_MASK *SelectedInstances,
  BioAPI_BIR_SUBTYPE_MASK CapturedInstances,
  const uint8_t *Text,
  BioAPI_GUI_RESPONSE *Response);
```

9.2.5.2 Описание

Данный пункт описывает тип указателя на функцию для функции обработчика события ГИП инфраструктуры, который обрабатывает обратные вызовы уведомления о событии выбора ГИП, поступающие от ПБУ. Для того чтобы получать уведомления о событии выбора ГИП, инфраструктура должна зарегистрировать функцию обратного вызова типа BioSPI_GUI_SELECT_EVENT_HANDLER путем предоставления адреса обратного вызова функции в вызове ***BioSPI_SubscribeToGUIEvents*** (см. 9.3.3.3).

ПБУ производит обратный вызов функции инфраструктуры этого типа для того, чтобы уведомить инфраструктуру (в конечном счете, приложение) о том, что создано событие выбора ГИП в процессе выполнения функции ПБУ.

ПБУ может создавать события выбора ГИП только в процессе выполнения вызова ***BioSPI_Capture, BioSPI_Verify, BioSPI_Identify*** или ***BioSPI_Enroll.*** Событие выбора ГИП со значением момента BioAPI_GUI_MOMENT_BEFORE_START генерируется перед запуском каждого цикла подопераций, а событие выбора ГИП со значением момента BioAPI_GUI_MOMENT_AFTER_END генерируется после окончания каждого цикла подопераций (см. подраздел 7.66).

9.2.5.3 Параметры

Параметры данного типа указателя на функцию совпадают с параметрами BioAPI_GUI_SELECT_EVENT_HANDLER, кроме того, что отсутствует параметр контекстного адреса.

Примечание — См. также раздел 7.

(Продолжение см. с. 96)

9.2.5.4 Возвращаемое значение

Значение BioAPI_RETURN указывает на то, что функция была выполнена успешно, или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

9.2.5.5 Ошибки

BioAPIERR_USER_CANCELLED

BioAPIERR_FUNCTION_FAILED

9.2.6 BioSPI_GUI_STATE_EVENT_HANDLER (БиоАПИ 2.1)

Данный пункт применяется только при использовании БиоАПИ версии 2.1.

9.2.6.1 Функция обратного вызова

```
typedef BioAPI_RETURN (BioAPI
*BioSPI_GUI_STATE_EVENT_HANDLER)
   (const BioAPI_UUID *BSPUuid,
   BioAPI_UNIT_ID UnitID,
   const BioAPI_HANDLE *BSPHandle,
   BioAPI_GUI_OPERATION Operation,
   BioAPI_GUI_SUBOPERATION Suboperation,
   BioAPI_BIR_PURPOSE Purpose,
   BioAPI_GUI_MOMENT Moment,
   BioAPI_RETURN ResultCode,
   int32_t EnrollSampleIndex,
   const BioAPI_GUI_BITMAP_ARRAY *Bitmaps,
   const uint8_t *Text,
   BioAPI_GUI_RESPONSE *Response,
   int32_t *EnrollSampleIndexToRecapture);
```

9.2.6.2 Описание

Данный пункт описывает тип указателя на функцию для функции обработчика события ГИП инфраструктуры, который обрабатывает обратные вызовы уведомления о событии изменения состояния ГИП, поступающие от ПБУ. Для того чтобы получать уведомления о событии изменения состояния ГИП, инфраструктура должна зарегистрировать функцию обратного вызова типа BioSPI_GUI_STATE_ EVENT_ HANDLER путем предоставления адреса обратного вызова функции при вызове ***BioSPI_SubscribeToGUIEvents*** (см. 9.3.3.3).

ПБУ производит обратный вызов функции инфраструктуры этого типа для того, чтобы уведомить инфраструктуру (и, в конечном счете, приложение) о том, что создано событие изменения состояния ГИП в процессе выполнения функции ПБУ.

(Продолжение см. с. 97)

ПБУ может создавать события изменения состояния ГИП только в процессе выполнения вызова ***BioSPI_Capture, BioSPI_Verify, BioSPI_Identify*** или ***BioSPI_Enroll.*** Событие изменения состояния ГИП со значением момента BioAPI_GUI_MOMENT_BEFORE_START генерируется перед запуском каждой подоперации, а событие выбора ГИП со значением момента BioAPI_GUI_MOMENT_AFTER_END генерируется после окончания каждой подоперации (см. подраздел 7.66).

9.2.6.3 Параметры

Параметры указателя на функцию данного типа совпадают с параметрами BioAPI_GUI_ STATE_EVENT_HANDLER, за исключением отсутствия параметра контекстного адреса.

Примечание — См. также раздел 7.

9.2.6.4 Возвращаемое значение

Значение BioAPI_RETURN указывает на то, что функция была выполнена успешно, или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

9.2.6.5 Ошибки

BioAPIERR_USER_CANCELLED

BioAPIERR_FUNCTION_FAILED».

Подпункт 9.3.1.1.1 дополнить абзацем (после первого):

«При использовании БиоАПИ версии 2.1 ПБУ, получающий вызов данной функции, должен для каждого существующего модуля БиоАПИ немедленно выслать одно уведомление о событии «установка» путем обратного вызова функции, предоставленной инфраструктурой, в адрес, который указывается в ***BioAPINotifyCallback.*** Если биометрическое приложение предоставило в вызове функции ***BioAPI_BSPLoad*** обработчик события, то инфраструктура, в свою очередь, произведет обратный вызов обработчика событий приложения. Если аппаратный компонент для данного модуля БиоАПИ не предоставлен, то событие «установки» не должно вызываться до тех пор, пока оно не будет подключено».

Подпункт 9.3.1.3.2 дополнить абзацем (перед последним примечанием):

«При использовании БиоАПИ версии 2.1 параметр не может иметь значение BioAPI_INVALID_BSP_HANDLE».

Пункт 9.3.1 дополнить подпунктами — 9.3.1.8, 9.3.1.9:

«9.3.1.8 BioSPI_Control (БиоАПИ 2.1)

Данный подпункт применяется только при использовании БиоАПИ версии 2.1.

(Продолжение см. с. 98)

```
BioAPI_RETURN BioAPI BioSPI_Control
  (BioAPI_HANDLE BSPHandle,
  BioAPI_UNIT_ID UnitID,
  const BioAPI_UUID *ControlCode,
  const BioAPI_DATA *InputData,
  BioAPI_DATA *OutputData);
```

Примечание — Подробности определения данной функции представлены в 8.1.13, BioAPI_Control.

9.3.1.9 BioSPI_Transform (БиоАПИ 2.1)

Данный подпункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioSPI_Transform
  (BioAPI_HANDLE BSPHandle,
  const BioAPI_UUID *OperationUUID,
  const BioAPI_INPUT_BIR *InputBIRs,
  uint32_t NumberOfInputBIRs,
  BioAPI_BIR_HANDLE **OutputBIRs,
  uint32_t *NumberOfOutputBIRs);
```

Примечание — Подробности определения данной функции представлены в 8.1.14, BioAPI_Transform».

Подпункт 9.3.3.2. Заголовок изложить в новой редакции:

«9.3.3.2 BioSPI_ SetGUICallbacks (БиоАПИ 2.0)»;

дополнить абзацем (перед первым):

«Данный подпункт применяется только при использовании БиоАПИ версии 2.0».

Пункт 9.3.3 дополнить подпунктами — 9.3.3.3—9.3.3.4.1:

«9.3.3.3 BioSPI_SubscribeToGUIEvents (БиоАПИ 2.1)

Данный подпункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioSPI_SubscribeToGUIEvents
  (const BioAPI_UUID *BSPUuid,
  BioSPI_GUI_SELECT_EVENT_HANDLER FwGUISelectEventHandler,
  BioSPI_GUI_STATE_EVENT_HANDLER FwGUIStateEventHandler,
  BioSPI_GUI_PROGRESS_EVENT_HANDLER FwGUIProgress-
EventHandler);
```

Данная функция предоставляет ПБУ адреса обратных вызовов обработчика событий выбора ГИП, обработчика событий изменения состояния ГИП и обработчика событий выполнения ГИП. Все три адреса не должны быть установлены на пустой указатель.

(Продолжение см. с. 99)

После вызова данной функции и до последующего вызова ***BioSPI_UnsubscribeFromGUIEvents*** или ***BioSPI_BSPUnload*** для каждого события ГИП, которое генерирует ПБУ в процессе выполнения вызовов функции ПБУ, ПБУ должен уведомить об этом событии инфраструктуру путем ответного вызова обработчика событий ГИП инфраструктуры, соответствующего типу события ГИП. Кроме того, любой ГИП, управляемый ПБУ, должен быть деактивирован в течение этого времени.

Параметр *BSPUuid* предоставляется только в качестве показателя надежности. ПБУ должен подтвердить, что значение данного параметра соответствует значению УУИД продукта ПБУ.

Эта функция, в отличие от функции ***BioAPI_SubscribeToGUIEvents,*** не должна создавать новую подписку на каждый вызов. На каждый вызов ПБУ должен просто заменить адреса старых обратных вызовов (изначально пустой указатель) адресами, предоставленными в вызове.

Единственный вызов ***BioAPI_UnsubscribeFromGUIEvents*** очистит все три адреса обратных вызовов. Необходим только один такой вызов, даже в том случае, если функция ***BioAPI_SubscribeToGUIEvents*** вызывалась многократно.

Примечание — После того как инфраструктура вызвала данную функцию, она уже не вызовет ее до тех пор, пока не будет произведен вызов ***BioSPI_UnsubscribeFromGUIEvents*** или ***BioSPI_BSPUnload.*** Инфраструктура передает все три адреса обратного вызова при каждом вызове данной функции.

9.3.3.3.1 Параметры

BSPUuid (входной) — УУИД ПБУ.

FwGUISelectEventHandler (входной) — адрес обратного вызова функции инфраструктуры, которая получает уведомления о событии выбора ГИП от ПБУ.

FwGUIStateEventHandler (входной) — адрес обратного вызова функции инфраструктуры, которая получает уведомления о событии изменения состояния ГИП от ПБУ.

FwGUIProgressEventHandler (входной, необязательный) — адрес обратного вызова функции инфраструктуры, которая получает уведомления о событии выполнения ГИП от ПБУ.

9.3.3.4 BioSPI_UnsubscribeFromGUIEvents (БиоАПИ 2.1)

Данный подпункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioSPI_UnsubscribeFromGUIEvents
  (const BioAPI_UUID *BSPUuid);
```

Данная функция очищает адреса обратных вызовов обработчика событий выбора ГИП, обработчика событий изменения состояния ГИП и обработчика событий выполнения ГИП.

(Продолжение см. с. 100)

После вызова данной функции ПБУ должен прекратить уведомлять инфраструктуру о событиях ГИП.

Параметр *BSPUuid* предоставляется только в качестве показателя надежности. ПБУ должен подтвердить, что значение данного параметра соответствует значению УУИД продукта ПБУ.

9.3.3.4.1 Параметры

BSPUuid (входной) — УУИД ПБУ.

Примечание — См. также раздел 7».

Пункт 10.1.2. Таблицу 3 для поля BSPUUID изложить в новой редакции:

Имя поля	Тип данных поля	Описание
BSPUUID	UINT8_T (16)	УУИД, однозначно идентифицирующий ПБУ. Данное поле должно присутствовать только при использовании БиоАПИ версии 2.0
BSPProductUUID	UINT8_T (16)	УУИД, однозначно идентифицирующий ПБУ как продукт программного обеспечения. Данное поле должно присутствовать только при использовании БиоАПИ версии 2.1

таблицу 3 дополнить элементами схемы ПБУ (перед сноской):

Имя поля	Тип данных поля	Описание
MaxNumEnrollInstances	UINT32_T	Максимальное число различных экземпляров, для которых ПБУ может создать контрольные шаблоны за одну операцию регистрации. Данное поле должно присутствовать только при использовании БиоАПИ версии 2.1

(Продолжение см. с. 101)

Окончание

Имя поля	Тип данных поля	Описание
HostingEndpointIRI	UINT8_T	ИИР, идентифицирующий инфраструктуру, чей регистр компонентов содержит регистрацию ПБУ. Данное поле должно присутствовать только при использовании БиоАПИ версии 2.1
BSPAccessUUID	BioAPI_UUID	УУИД, уникальный в области применения приложения, который приложение может использовать для обращения к ПБУ вместо УУИД продукта ПБУ. Данное поле должно присутствовать только при использовании БиоАПИ версии 2.1

Подраздел 10.2 дополнить пунктами — 10.2.3—10.2.7:

«10.2.3 Функция BioAPI_RegisterBSP (БиоАПИ 2.1)

Данный пункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioAPI_RegisterBSP
  (const uint8_t *HostingEndpointIRI,
  const BioAPI_BSP_SCHEMA *BSPSchema, BioAPI_BOOL Update);
```

10.2.3.1 Описание

В спецификации, предоставленной настоящим стандартом, данная функция выполняет те же действия, что и функция ***BioAPI_Util_InstallBSP*** с параметром *Action*, установленным на BioAPI_INSTALL_ACTION_INSTALL или BioAPI_INSTALL_ACTION_REFRESH. Значение дополнительного параметра *HostingEndpointIRI* должно быть проигнорировано инфраструктурами, соответствующими этому разделу настоящего стандарта, и установлено на пустой указатель приложением. Данный параметр предоставлен для поддержки стандартов межсетевого обмена.

В отличие от функции ***BioAPI_Util_InstallBSP*** данная функция не содержит параметр *Error*. Информация об ошибке предоставлена в возвращаемом значении.

(Продолжение см. с. 102)

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ПБУ.

10.2.3.2 Параметры

HostingEndpointEndpointIRI — данный параметр должен быть проигнорирован инфраструктурами, соответствующими этому разделу настоящего стандарта, и установлен на пустой указатель приложением. Данный параметр предоставлен для поддержки стандартов межсетевого обмена.

BSPSchema (входной) — указатель на элементы схемы ПБУ, определенные в 7.16, описывающие свойства и характеристики устанавливаемого ПБУ.

Update (входной) — булево значение, указывающее на то, должна ли функция обновить существующую регистрацию схемы ПБУ или должна создать новую регистрацию. Если значение данного параметра равно нулю и схема ПБУ находится в реестре компонентов, то функция должна возвратить значение BioAPIERR_COMPONENT_ALREADY_REGISTERED. Если значение данного параметра не равно нулю и схема ПБУ не находится в реестре компонентов, то функция должна возвратить значение BioAPIERR_COMPONENT_NOT_REGISTERED.

10.2.3.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на успешное выполнение функции или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

10.2.3.4 Ошибки

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

10.2.4 Функция BioAPI_UnregisterBSP (БиоАПИ 2.1)

Данный пункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioAPI_UnregisterBSP
  (const uint8_t *HostingEndpointIRI,
  const BioAPI_UUID *BSPUuid);
```

10.2.4.1 Описание

В спецификации, предоставленной настоящим стандартом, данная функция выполняет те же действия, что и функция ***BioAPI_Util_InstallBSP*** с параметром *Action*, установленным на BioAPI_INSTALL_ACTION_UNINSTALL. Значение дополнительного параметра *HostingEndpointIRI* должно быть проигнорировано инфраструктурами, соответствующими этому разделу настоящего стандарта, и установлено на пустой указатель приложением. Данный параметр предоставлен для поддержки стандартов межсетевого обмена.

(Продолжение см. с. 103)

В отличие от функции ***BioAPI_Util_UninstallBSP*** данная функция не содержит параметр *Error*. Информация об ошибке предоставлена в возвращаемом значении.

Приложения, соответствующие настоящему стандарту, должны либо не вызывать функцию, либо (если вызывают ее) должны установить данный параметр *HostingEndpointIRI* на пустой указатель. Функция предоставлена для поддержки стандартов межсетевого обмена.

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ПБУ.

10.2.4.2 Параметры

HostingEndpointEndpointIRI — данный параметр должен быть проигнорирован инфраструктурами, соответствующими этому разделу настоящего стандарта, и установлен на пустой указатель приложением. Данный параметр предоставлен для поддержки стандартов межсетевого обмена.

BSPSchema (входной) — указатель на УУИД ПБУ, чья регистрация схемы ПБУ должна быть удалена из реестра компонентов.

10.2.4.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на успешное выполнение функции или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

10.2.4.4 Ошибки

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

10.2.5 Функция BioAPI_RegisterBFP (БиоАПИ 2.1)

Данный пункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioAPI_RegisterBFP
  (const uint8_t *HostingEndpointIRI,
  const BioAPI_BFP_SCHEMA *BFPSchema,
  BioAPI_BOOL Update);
```

10.2.5.1 Описание

Данная функция выполняет те же действия, что и функция ***BioAPI_Util_InstallBFP*** с параметром *Action,* установленным на BioAPI_INSTALL_ACTION_INSTALL или BioAPI_INSTALL_ACTION_REFRESH. Значение дополнительного параметра *HostingEndrpointIRI* должно быть проигнорировано инфраструктурами, соответствующими этому разделу настоящего стандарта, и установлено на пустой указатель приложением. Данный параметр предоставлен для поддержки стандартов межсетевого обмена.

(Продолжение см. с. 104)

В отличие от функции ***BioAPI_Util_UninstallBFP*** данная функция не содержит параметр *Error*. Информация об ошибке предоставлена в возвращаемом значении.

Приложения, соответствующие настоящему стандарту, должны либо не вызывать функцию, либо (если вызывают ее) должны установить данный параметр *HostingEndpointIRI* на пустой указатель. Функция предоставлена для поддержки стандартов межсетевого обмена.

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ПБФ.

10.2.5.2 Параметры

HostingEndpointEndpointIRI — данный параметр должен быть проигнорирован инфраструктурами, соответствующими этому разделу настоящего стандарта, и установлен на пустой указатель приложением. Данный параметр предоставлен для поддержки стандартов межсетевого обмена.

BFPSchema (входной) — указатель на элементы схемы ПБФ, определенные в 7.16, описывающие свойства и характеристики устанавливаемого ПБФ.

Update (входной) — булево значение, определяющее должна ли функция обновить существующую регистрацию схемы ПБФ или должна создать новую регистрацию. Если значение данного параметра равно нулю и схема ПБФ находится в реестре компонентов, то функция должна возвратить значение BioAPIERR_COMPONENT_ALREADY_REGISTERED. Если значение данного параметра не равно нулю и схема ПБФ не находится в реестре компонентов, то функция должна возвратить значение BioAPIERR_COMPONENT_NOT_REGISTERED.

10.2.5.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на успешное выполнение функции или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

10.2.5.4 Ошибки

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

10.2.6 Функция BioAPI_UnregisterBFP (БиоАПИ 2.1)

Данный пункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioAPI_UnregisterBFP
  (const uint8_t *HostingEndpointIRI,
  const BioAPI_UUID *BFPUuid);
```

10.2.6.1 Описание

В спецификации, предоставленной настоящим стандартом, данная функция выполняет те же действия, что и функция ***BioAPI_Util_InstallBFP***

(Продолжение см. с. 105)

с параметром *Action,* установленным на BioAPI_INSTALL_ACTION_ UNINSTALL. Значение дополнительного параметра *HostingEndpointIRI* должно быть проигнорировано инфраструктурами, соответствующими этому разделу настоящего стандарта, и установлено на пустой указатель приложением. Данный параметр предоставлен для поддержки стандартов межсетевого обмена.

В отличие от функции ***BioAPI_Util_UninstallBFP*** данная функция не содержит параметр *Error.* Информация об ошибке предоставлена в возвращаемом значении.

Приложения, соответствующие настоящему стандарту, должны либо не вызывать функцию, либо (если вызывают ее) должны установить данный параметр *HostingEndpointIRI* на пустой указатель. Функция предоставлена для поддержки стандартов межсетевого обмена.

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ПБФ.

10.2.6.2 Параметры

HostingEndpointEndpointIRI — данный параметр должен быть проигнорирован инфраструктурами, соответствующими этому разделу настоящего стандарта, и установлен на пустой указатель приложением. Данный параметр предоставлен для поддержки стандартов межсетевого обмена.

BFPSchema (входной) — указатель на УУИД ПБФ, чья регистрация схемы ПБФ должна быть удалена из реестра компонентов.

10.2.6.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на успешное выполнение функции или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

10.2.6.4 Ошибки

Данные об обработке ошибок БиоАПИ приведены в разделе 11.

10.2.7 Функция BioAPI_GetLastErrorInfo (БиоАПИ 2.1)

Данный пункт применяется только при использовании БиоАПИ версии 2.1.

```
BioAPI_RETURN BioAPI BioAPI_GetLastErrorInfo
  (BioAPI_ERROR_INFO *ErrorInfo);
```

10.2.7.1 Описание

Данная функция записывает в элементы выходного параметра *ErrorInfo* (см. 7.72) информацию о последнем типе ошибки, инициировавшем функцию БиоАПИ возвратить код ошибки.

Данная функция в основном предназначена для предоставления приложению диагностической информации. Так как возможные значения элементов *ErrorInfo* не стандартизированы и могут меняться в зависимо-

(Продолжение см. с. 106)

сти от инфраструктуры и реализации, нормальные приложения БиоАПИ не должны основываться на этих значениях при принятии каких-либо важных решений по обработке данных.

Данная функция обрабатывается в инфраструктуре БиоАПИ и не передается ПБУ.

10.2.7.2 Параметры

ErrorInfo (выходной) — структура данных типа BioAPI_ERROR_INFO, содержащая информацию о последней произошедшей ошибке.

10.2.7.3 Возвращаемое значение

Значение BioAPI_RETURN указывает на успешное выполнение функции или определяет тип ошибки. Значение BioAPI_OK указывает на успешное выполнение функции. Все остальные значения описывают тип ошибки.

10.2.7.4 Ошибки

Данные об обработке ошибок БиоАПИ приведены в разделе 11».

Пункт 11.2.3 дополнить абзацами:

«#define BioAPIERR_IDENTIFY_IN_PROGRESS (0x000120)

Идентификация находится в процессе выполнения. Данное определение применяется только при использовании БиоАПИ версии 2.1.

#define BioAPIERR_LOW_QUALITY_REFERENCE_TEMPLATE (0x000121)

Качество контрольного образца низкое. Данное определение применяется только при использовании БиоАПИ версии 2.1.

#define BioAPIERR_NO_GUI_EVENT_HANDLER (0x000122)

Обработчик событий ГИП не установлен. Данное определение применяется только при использовании БиоАПИ версии 2.1.

#define BioAPIERR_TRANSFORMATION_NOT_SUPPORTED (0x000121)

ПБУ не поддерживает трансформацию ЗБИ. Данное определение применяется только при использовании БиоАПИ версии 2.1.

#define BioAPIERR_INVALID_ATTACH_SESSION (0x000122)

Присоединенная сессия недействительна. Данное определение применяется только при использовании БиоАПИ версии 2.1.

#define BioAPIERR_COMPONENT_ALREADY_REGISTERED (0x000123)

Схема ПБУ или ПБФ уже присутствует в реестре компонентов. Данное определение применяется только при использовании БиоАПИ версии 2.1.

#define BioAPIERR_COMPONENT_NOT_REGISTERED (0x000124)

(Продолжение см. с. 107)

Схема ПБУ или ПБФ не присутствует в реестре компонентов. Данное определение применяется только при использовании БиоАПИ версии 2.1».

Пункт 11.2.7 дополнить абзацами (после примечания):

«#define BioAPIERR_TOO_EARLY (0x000502)

Действия пользователя произошли слишком рано. Данное определение применяется только при использовании БиоАПИ версии 2.1.

#define BioAPIERR_TOO_LATE (0x000503)

Действия пользователя произошли слишком поздно. Данное определение применяется только при использовании БиоАПИ версии 2.1.

#define BioAPIERR_TOO_FAST (0x000504)

Действия пользователя произошли слишком быстро. Данное определение применяется только при использовании БиоАПИ версии 2.1.

#define BioAPIERR_TOO_SLOW (0x000505)

Действия пользователя произошли слишком медленно. Данное определение применяется только при использовании БиоАПИ версии 2.1.

#define BioAPIERR_TOO_LONG (0x000506)

Объем полученных данных слишком велик. Данное определение применяется только при использовании БиоАПИ версии 2.1.

#define BioAPIERR_TOO_SHORT (0x000507)

Объем полученных данных слишком мал. Данное определение применяется только при использовании БиоАПИ версии 2.1.

#define BioAPIERR_TOO_LARGE (0x000508)

Объем полученных данных слишком велик. Данное определение применяется только при использовании БиоАПИ версии 2.1.

#define BioAPIERR_TOO_SMALL (0x000509)

Объем полученных данных слишком мал. Данное определение применяется только при использовании БиоАПИ версии 2.1».

Приложение А. Пункт А.1.1 изложить в новой редакции:

«А.1.1 Соответствие требованиям настоящего стандарта состоит из следующих пяти классов:

- БиоАПИ совместимое биометрическое приложение;
- БиоАПИ 2.0 совместимая инфраструктура БиоАПИ;
- БиоАПИ 2.1 совместимая инфраструктура БиоАПИ;
- БиоАПИ 2.0 совместимый ПБУ, включающий в себя один из следующих подклассов:
  - БиоАПИ 2.0 совместимый ПБУ для верификации;
  - БиоАПИ 2.0 совместимый ПБУ для идентификации;
  - БиоАПИ 2.0 совместимый ПБУ для получения данных;

(Продолжение см. с. 108)

- БиоАПИ 2.0 совместимый механизм верификации;
- БиоАПИ 2.0 совместимый механизм идентификации.

- БиоАПИ 2.1 совместимый ПБУ, включающий в себя один из следующих подклассов:
  - БиоАПИ 2.1 совместимый ПБУ для верификации;
  - БиоАПИ 2.1 совместимый ПБУ для идентификации;
  - БиоАПИ 2.1 совместимый ПБУ для получения данных;
  - БиоАПИ 2.1 совместимый механизм верификации;
  - БиоАПИ 2.1 совместимый механизм идентификации».

Пункт А.3.2 после слов «Для соответствия БиоАПИ» дополнить словами: «(или версия 2.0 или версия 2.1)».

Пункт А.3.3 изложить в новой редакции:

«А.3.3 Соответствующая инфраструктура БиоАПИ 2.0 должна поддерживать все обязательные и дополнительные опции, установленные настоящим стандартом, кроме подпунктов и определений, которые указаны как применимые только при использовании версии 2.1».

Подраздел А.3 дополнить пунктами — А.3.4, А.3.5:

«А.3.4 Соответствующая инфраструктура БиоАПИ 2.1 должна поддерживать все обязательные и дополнительные опции, установленные настоящим стандартом, кроме подпунктов и определений, которые указаны как применимые только при использовании версии 2.0.

А.3.5 Не определено никаких подгрупп функций, соответствующих инфраструктуре».

Подраздел А.4. Первый абзац. Заменить слова: «Для соответствия настоящему стандарту» на «Для соответствия БиоАПИ (или версия 2.0, или версия 2.1)»;

дополнить абзацами (после первого):

«ПБУ, совместимые с БиоАПИ 2.0 (любого совместимого подкласса), должны поддерживать все обязательные функции, установленные настоящим стандартом для соответствующего подкласса, кроме подпунктов и определений, которые указаны как применимые только при использовании версии 2.1.

ПБУ, совместимые с БиоАПИ 2.1 (любого совместимого подкласса), должны поддерживать все обязательные функции, установленные настоящим стандартом для соответствующего подкласса, кроме подпунктов и определений, которые указаны как применимые только при использовании версии 2.0»;

таблицу А.1 для области «Функции управления обратными вызовами и событиями» и функции ***«BioSPI_SetGUICallbacks»*** изложить в новой редакции и дополнить следующими функциями:

(Продолжение см. с. 109)

Функция	ПБУ для			Механизм верификации	Механизм идентификации
	верификации	идентификации	получения данных		
BioSPI_SetGUICallbacks (БиоАПИ 2.0)					
BioSPI_SubscribeToGUIEvents (БиоАПИ 2.1)					
BioSPI_UnsubscribeFromGUIEvents (БиоАПИ 2.1)					

таблицу А.1 дополнить функциями — «События ГИП» (в конце таблицы):

Функция	ПБУ для			Механизм верификации	Механизм идентификации
	верификации	идентификации	получения данных		
События ГИП					
События выбора ГИП (БиоАПИ 2.1)					
События изменения состояния ГИП (БиоАПИ 2.1)					
События выполнения ГИП (БиоАПИ 2.1)					

Подпункт А.4.6.2. Таблица А.2. Подфункцию «Обратные вызовы потокового ГИП» изложить в новой редакции и дополнить следующими подфункциями:

Возможность	Поддерживается	Не поддерживается
Обратные вызовы потокового ГИП (БиоАПИ 2.1)		

(Продолжение см. с. 110)

Окончание

Возможность	Поддерживается	Не поддерживается
Генерирование событий выполнения ГИП в течение подопераций захвата (БиоАПИ 2.1)		
Генерирование событий выполнения ГИП в течение подопераций идентификации сопоставления (БиоАПИ 2.1)		

Приложение В. Пункт В.5. Заменить слова: «{iso registration-authority cbeff(19785) organization(0) 257 bir(1) patron-format (6)} на «{iso registration-authority cbeff(19785) biometric-organization(0) jtc1-sc37(257) patron-format(1) bioAPI(8)}».

Пункт В.10. Таблицу В.1 для наименований «BioAPI_BIR_BIOMETRIC_TYPE (БиоАПИ, 7.8)» и «BioAPI_BIR_SUBTYPE (БиоАПИ, 7.14)» изложить в новой редакции:

Наименование поля БиоАПИ, ссылки БиоАПИ и поля формата постоянного клиента	Наименование элемента данных ЕСФОБД	Длина, байт	Абстрактное значение	Кодировка[a]	Ссылка ЕСФОБД
BioAPI_BIR_BIOMETRIC_TYPE (БиоАПИ, 7.8)	CBEFF_BDB_biometric_type	4	NO BIOTYPE AVAILABLE (доступных биотипов нет) MULTIPLE_BIOMETRIC_TYPES FACE (изображение лица) VOICE (голос) FINGER (изображение отпечатка пальца) IRIS (изображение радужной оболочки глаза) RETINA (сетчатка) HAND GEOMETRY (геометрия контура кисти руки)	'00 00 00 00' '00 00 00 01' '00 00 00 02' '00 00 00 04' '00 00 00 08' '00 00 00 10' '00 00 00 20' '00 00 00 40'	

(Продолжение см. с. 111)

Окончание

Наименование поля БиоАПИ, ссылки БиоАПИ и поля формата постоянного клиента	Наименование элемента данных ЕСФОБД	Длина, байт	Абстрактное значение	Кодировка[a]	Ссылка ЕСФОБД
BioAPI_BIR_BIOMETRIC_TYPE (БиоАПИ, 7.8)	CBEFF_BDB_biometric_type	4	SIGNATURE SIGN (динамика подписи) KEYSTROKE (нажатие клавиши) LIP MOVEMENT (движения губ) RESERVED (зарезервировано) RESERVED (зарезервировано) GAIT (походка) VEIN (вена) DNA (ДНК) EAR (ухо) FOOT (ступня) SCENT (запах) OTHER (другое) PASSWORD (пароль)	00 00 00 80' '00 00 01 00' '00 00 02 00' '00 00 04 00' '00 00 08 00' '00 00 10 00' '00 00 20 00' '00 00 40 00' '00 00 80 00' '00 01 00 00' '00 02 00 00' '40 00 00 00' '80 00 00 00'	
BioAPI_BIR_SUBTYPE (БиоАПИ, 7.14)	CBEFF_BDB_biometric_subtype	1	NO_SUBTYPE_AVAILABLE VEIN_ONLY_MASK LEFT_MASK RIGHT_MASK THUMB (большой палец) POINTERFINGER (указательный палец) MIDDLEFINGER (средний палец) RINGFINGER (безымянный палец) LITTLEFINGER (мизинец) VEIN_PALM (ладонь) VEIN_BACKOFHAND (тыльная сторона ладони) VEIN_WRIST (запястье)	'00' '80' '01' '02' '04' '08' '10' '20' '40' '04' '08' '10'	

(Продолжение см. с. 112)

Приложение С. Подраздел С.7 изложить в новой редакции:

«С.7 Анализ интерфейса пользователя (БиоАПИ 2.1)

Анализ и примеры в данном подразделе применяются только при использовании БиоАПИ версии 2.1.

С.7.1 Общие положения

Интерфейс пользователя для паролей и личных идентификационных номеров является достаточно простым, но для биометрической технологии он может быть весьма сложным и в значительной степени зависеть от применяемой технологии, требуя многочисленные взаимодействия с пользователем. Некоторые биометрические технологии демонстрируют поток данных пользователю (например, лицо и голос), а другие требуют от пользователя подтверждения каждого взятого образца (например, лицо, голос, подпись). В процессе регистрации некоторые технологии верифицируют каждый взятый образец на основе предыдущих образцов (тест-верификация), используя определенный алгоритм сопоставления. Число образцов, взятых для конкретного назначения (регистрация, верификация или идентификация), может меняться в зависимости от используемой технологии и метода реализации, и интерфейс пользователя для регистрации обычно отличается от интерфейсов для верификации и идентификации. В первом случае интерфейс пользователя обычно включает в себя больше взаимодействий и является наиболее трудоемким, а во втором случае является наиболее простым и быстрым в использовании. Таким образом, стандартизация интерфейсов пользователя для биометрической регистрации, верификации или идентификации представляет собой сложную задачу в том случае, если необходимо предоставить стандартизированные механизмы или достаточную гибкость. Интерфейс ГИП БиоАПИ предоставляет этот стандарт с соответствующей гибкостью.

Большинство текущих реализаций ПБУ содержат встроенный интерфейс пользователя (на самом деле это требование БиоАПИ для соответствующей реализации ПБУ), таким образом, приложению не требуется знать все подробности о взаимодействии между ПБУ и пользователем, предоставившим биометрический образец. Однако БиоАПИ также дает возможность приложению контролировать «вид и поведение» интерфейса пользователя, разрешая приложению предоставить обработчик событий ГИП, который вызывает инфраструктура БиоАПИ в течение определенных операций в ответ на обратные вызовы, поступающие от ПБУ. Ниже приведены некоторые преимущества управляемого приложением ГИП над ГИП управляемым ПБУ:

а) приложение может использовать графические изображения и текст (возможно в отдельном окне) для предоставления пользователю обратной связи, задач или руководств относительно текущей биометрической операции;

(Продолжение см. с. 113)

b) изображения могут демонстрироваться таким образом, чтобы соответствовать виду и поведению отдельного приложения и находиться в главном окне приложения или в отдельном окне, по усмотрению приложения; и

c) приложение может применять язык человека, который может не поддерживаться ПБУ, так как (если используется управляемый приложением ГИП) большинство взаимодействий посредством обратных вызовов между ПБУ и приложением (через инфраструктуру БиоАПИ) просто идентифицируют семантику, которая должна быть предоставлена пользователю; язык человека, применяемый для предоставления сообщения, является проблемой для приложения (несмотря на то, что событие выполнения ГИП также позволяет ПБУ предоставлять полутоновые и цветные изображения).

Уведомления о событиях выбора и изменения состояния ГИП применяются для управления сбором образцов и для указания приложению на изменение в состоянии (в процессе операций регистрации, верификации или идентификации). Приложение вызывает функцию ***BioAPI_SubscribeToGUIEvents*** для управления ГИП самостоятельно вместо того, чтобы позволять ПБУ управлять ГИП.

События выбора и изменения состояния ГИП генерируются ПБУ только в процессе выполнения операции, инициализированной вызовом приложением любой из следующих функций БиоАПИ:

- BioAPI_Capture;
- BioAPI_Enroll;
- BioAPI_Verify;
- BioAPI_Identify.

События выполнения ГИП могут быть сгенерированы ПБУ в процессе того, как ПБУ выполняет операцию, инициализированную вызовом приложения любой из следующих функций БиоАПИ:

- BioAPI_Capture;
- BioAPI_Process;
- BioAPI_CreateTemplate;
- BioAPI_VerifyMatch;
- BioAPI-IdentifyMatch;
- BioAPI_Enroll;
- BioAPI_Verify;
- BioAPI_Identify.

Приложение получит уведомления о событиях ГИП, которые предоставляют информацию, связанную с ГИП, в течение выполнения операций получения данных, регистрации, верификации или идентификации. В таблице С.1 представлен список событий ГИП, которые могут произойти.

(Продолжение см. с. 114)

Таблица С.1 — События ГИП

Событие ГИП	Описание
Событие выбора	Данное событие ГИП генерируется ПБУ перед запуском и после окончания цикла подопераций в процессе операций получения данных, верификации, идентификации и регистрации. В начале цикла подопераций событие выбора ГИП позволяет приложению определить (в его ответе на обратный вызов уведомления) подтип (см. ИСО/МЭК 19785-1, 6.5.7) или подтипы образцов, которые должны быть захвачены (например, для пальца: левый/ правый, указательный/ средний/ безымянный/ мизинец) при помощи подоперации(й) захвата в рамках цикла подопераций, который скоро будет запущен. Некоторые ПБУ способны захватывать одновременно несколько подтипов или экземпляров биометрической модальности (например, все пять пальцев или обе радужные оболочки глаза) за одну подоперацию захвата, поэтому в ответ на многоэкземплярный захват предоставляется битовая маска. В конце цикла подопераций событие выбора ГИП сообщает приложению о завершении цикла подопераций (возможно предоставляя дополнительную информацию) и позволяет приложению указать на то, должен ли ПБУ считать всю операцию завершенной (с возвратом приложению BioAPI_OK или кодом ошибки), прервать операцию (с возвратом приложению кода ошибки) или сбросить все текущие результаты и запустить новый цикл подопераций (без выхода из первоначального вызова функции ПБУ)
Событие изменения состояния	Данное событие генерируется ПБУ перед запуском и после завершения каждой подоперации операции. Определены следующие пять подопераций: - захват данных: захватывает один или несколько экземпляров модальности при помощи модуля сканера ПБУ и создает образец промежуточного уровня обработки с целью его последующей верификации, идентификации или регистрации; - обработка: обрабатывает промежуточный образец с целью его последующей верификации или идентификации и преобразовывает в обработанный образец;

(Продолжение см. с. 115)

Продолжение таблицы С.1

Событие ГИП	Описание
Событие изменения состояния	- создание шаблона: обрабатывает промежуточный образец с целью его последующей регистрации и преобразовывает его в контрольный шаблон, возможно после выбора «лучшего» промежуточного образца из набора; - верификация-сопоставление: выполняет верификацию-сопоставление образца, обработанного с целью его последующей верификации, и контрольного шаблона; в контексте операции регистрации, это также называется тест-верификация; - идентификация-сопоставление: выполняет идентификацию-сопоставление образца, обработанного с целью его последующей идентификации, и множества контрольных шаблонов. Пять подопераций имеют те же наименования, что и пять функций БиоАПИ, потому что они выполняют, в сущности, те же задания, хотя на другом абстрактном уровне. К примеру, для BioSPI_Process существует единственная подоперация обработки, также для BioSPI_CreateTemplate (единственная подоперация создания шаблона), BioSPI_VerifyMatch (единственная подоперация верификации-сопоставления) и BioSPI_IdentifyMatch (единственная подоперация идентификации-Агсопоставления). Однако для более сложных операций (BioSPI_Capture, BioSPI_Verify, BioSPI_Identify и BioSPI_Enroll) существует цикл подопераций, включающий в себя одну или более подопераций захвата данных, за которыми, возможно, следуют другие подоперации. Полный цикл может быть повторен один раз или более без выхода из операции. Если ПБУ предоставляет тип регистрации Многократный-Захват функции BioAPI_GUI_SELECT_EVENT_HANDLER, то цикл подопераций для BioSPI_Enroll состоит из подопераций многократного захвата, за которыми следует подоперация создания шаблона. Если ПБУ предоставляет тип регистрации Тестировать-Верифицировать, то цикл подопераций для BioSPI_ENROLL состоит из захвата с целью его дальнейшей регистрации, захвата с целью его дальнейшей верификации, создания шаблона, обработки и верификации-сопоставления (тест верификации)

(Продолжение см. с. 116)

Окончание таблицы С.1

Событие ГИП	Описание
Событие выполнения	ПБУ может сгенерировать данное событие в процессе любой подоперации в рамках операции захвата данных, обработки, создания шаблона, верификации-сопоставления, идентификации-сопоставления, верификации, идентификации или регистрации для сообщения приложению о выполнении (степени выполнения, завершенности) подоперации. Уведомление о событии выполнения ГИП для подоперации получения данных (захвата) может включать в себя потоковые данные (например, поток видео в реальном времени). Многократные уведомления о событии выполнения ГИП для длительных подопераций идентификации-сопоставления могут использоваться для сообщения пользователю о проценте выполнения подоперации идентификации-сопоставления

ПБУ управляется конечным механизмом, связанным с операцией, и предоставляет информацию об изменении состоянии, текстовые сообщения и (в соответствующих случаях) битовые изображения путем вызова дескриптора инфраструктуры уведомления о событии ГИП, который в свою очередь отвечает вызовом дескриптора уведомления о событии ГИП приложения. После демонстрации изображения пользователю (или оператору, управляющему биометрическим процессом), приложение определяет, продолжить ли вызывать первоначальную функцию, отменить ее или изменить путь, ответив обратному вызову уведомления о событии ГИП.

Событие выполнения ГИП может быть применено для предоставления потоковых данных приложению (предназначенных для демонстрации пользователю или оператору) в форме последовательности битовых изображений. В течение подоперации идентификации-сопоставления событие выполнения ГИП может быть применено для предоставления пользователю степени выполнения идентификации. Генерация события выполнения ГИП является прерогативой ПБУ, который указывает в своем реестре компонентов информацию о поддержке реализации данной возможности.

Функция ***BioAPI_SubscribeToGUIEvents*** подписывает приложение на события выбора, изменения состояния и выполнения ГИП, в то вре-

(Продолжение см. с. 117)

мя как функция ***BioAPI_Unsubscribe FromGUIEvents*** удаляет существующую подписку на событие ГИП. Все ПБУ, которые реализуют возможность управления ГИП приложением, необходимы для поддержки соответствующих функций ПБУ ***BioSPI_SubscribeToGUIEvents*** и ***BioSPI_UnsubscribeFromGUIEvents.***

Взаимодействия между биометрическим приложением и ПБУ, поддерживающим регистрацию, определяется типом регистрации, предоставленным ПБУ. Далее определены следующие два типа регистрации:

a) Тестировать-Верифицировать: В процессе операции регистрации (инициированной вызовом ***BioAPI_Enroll***) ПБУ, предоставляющий данный тип регистрации, выполняет:

1) подоперацию захвата данных для создания кандидата на контрольный шаблон;

2) вторую подоперацию захвата данных для получения образца для тестирования;

3) подоперацию по созданию шаблона (по первому образцу);

4) подоперацию обработки (по второму образцу); и

5) подоперацию верификации-сопоставления для определения того, совпадает ли образец для тестирования и кандидат на контрольный шаблон.

Кандидат на контрольный шаблон принимается только в том случае, когда операция верификации проведена успешно. В противном случае приложение может решить, должен ли цикл быть повторен, либо операция регистрации должна завершиться с ошибкой;

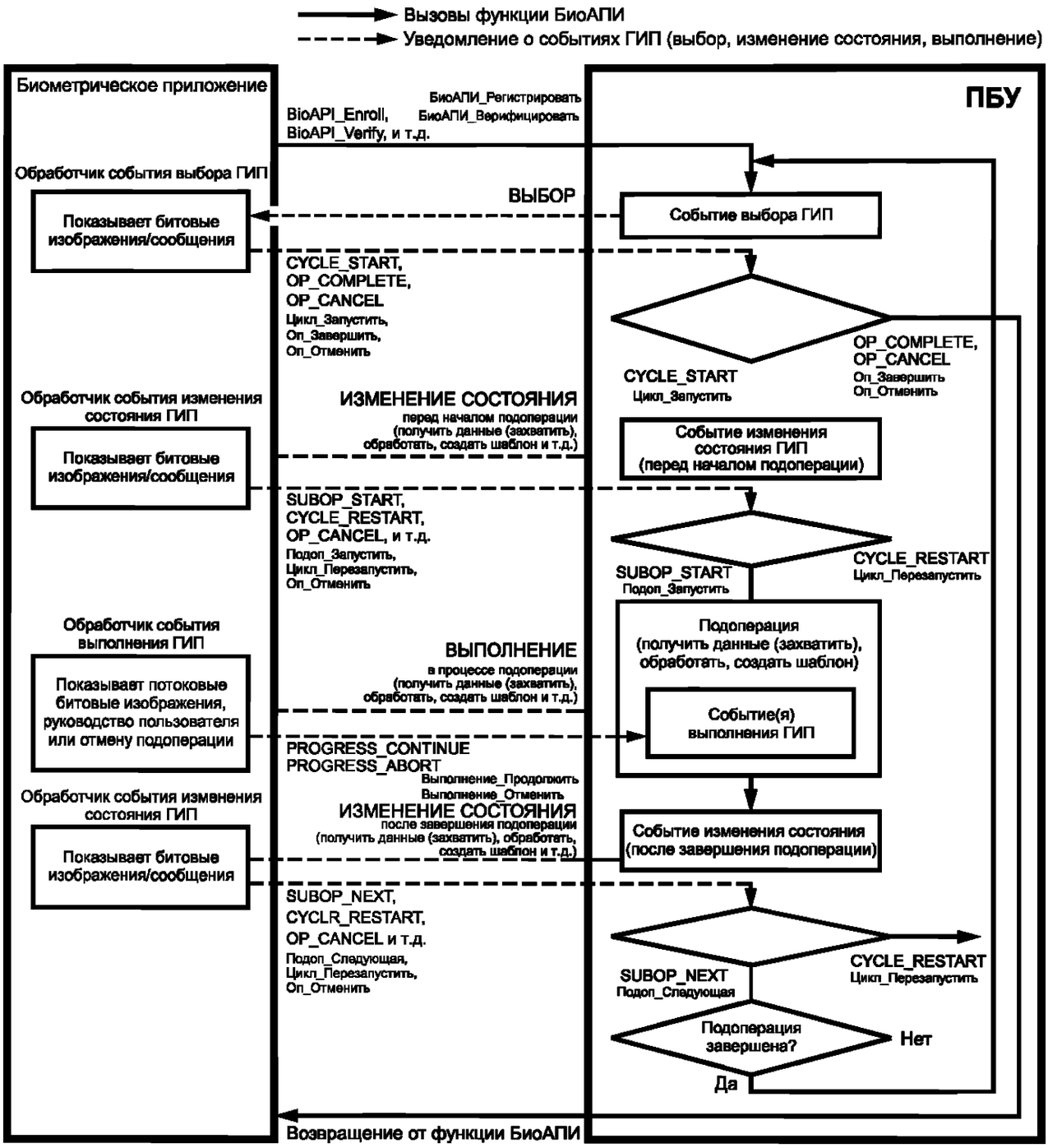
b) Многократный-Захват: ПБУ, предоставляющий данный тип регистрации, точно установил критерии соответствия образца для создания контрольного шаблона. В процессе операции регистрации ПБУ выполняет одну или более подопераций захвата данных в рамках определенного заранее ограничения (*MaxNumEnrollInstances,* см. 7.60) с целью получения образца, удовлетворяющего всем критериям.

Примечание — ПБУ, выполняющий (вероятно) несколько захватов с целью получения образца, используемого в качестве кандидата на регистрацию, и далее захватывающий следующий образец для сопоставления с кандидатом (сочетание Тестировать-Верифицировать и Многократного-Захвата) не поддерживается в версиях БиоАПИ 2.0 или 2.1.

ПБУ предоставляет свой тип регистрации в качестве входного параметра функции BioAPI_GUI_SELECT_EVENT_HANDLER, таким образом делая его доступным для приложения в начале каждой операции регистрации.

На рисунке С.2а показана обобщенная схема процесса реализации для ГИП, управляемого приложением.

(Продолжение см. с. 118)



П р и м е ч а н и е — В действительной реализации между приложением и ПБУ находится инфраструктура БиоАПИ. Инфраструктура БиоАПИ управляет всеми вызовами и уведомлением о событиях БиоАПИ.

Рисунок С.2а — Пример обобщенного потока процесса реализации ГИП

(Продолжение см. с. 119)

С.7.2 Пример последовательности уведомлений о событиях ГИП для операции Регистрации (тип регистрации — Тестировать-Верифицировать, без ошибок и перезапусков)

Пример последовательности уведомлений о событиях ГИП представлен в таблице С.2.

Таблица С.2

Событие ГИП	Входные параметры	Выходные параметры
выбор	Операция = **BioAPI_GUI_ OPERATION_ ENROLL** Момент = **BioAPI_GUI_ MOMENT_BEFORE_START** Код Результата = н/д МаксЧислоОбразцовРегистрации = 1 ВыбираемыеЭкземпляры = левый указательный палец, правый указательный палец ЗахваченныеЭкземпляры = н/д	Ответ = **BioAPI_GUI_ RESPONSE_CYCLE_ START** ВыбранныеЭкземпляры = левый указательный палец
изменение состояния	Операция = **BioAPI_GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_ GUI_SUBOPERATION_ CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_ BEFORE_ START** Код Результата = н/д ИндексОбразцаРегистрации = 1	Ответ = **BioAPI_GUI_ RESPONSE_SUBOP_ START** ИндексОбразцаРегистрацииНаПерезахват = н/д
выполнение	Операция = **BioAPI_GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_GUI_ SUBOPERATION_ CAPTURE** Назначение = **BioAPI_PURPOSE ENROLL** Момент = **BioAPI_GUI_ MOMENT_DURING** СтепеньВыполненияПодоперации = 20	Ответ = **BioAPI_ GUI_ RESPONSE_PROGRESS_ CONTINUE**

(Продолжение см. с. 120)

Продолжение таблицы С.2

Событие ГИП	Входные параметры	Выходные параметры
выполнение	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_DURING** СтепеньВыполненияПодоперации = 50	Ответ = **BioAPI_GUI_RESPONSE_PROGRESS_CONTINUE**
изменение состояния	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_AFTER_END** Код Результата = **BioAPI_OK** ИндексОбразцаРегистрации = 1	Ответ = **BioAPI_GUI_RESPONSE_SUBOP_NEXT** ИндексОбразцаРегистрацииНаПерезахват = н/д
изменение состояния	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CREATETEMPLATE** Назначение = н/д Момент = **BioAPI_GUI_MOMENT_BEFORE_START** Код Результата = н/д ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_RESPONSE_SUBOP_START** ИндексОбразцаРегистрацииНаПерезахват = н/д

(Продолжение см. с. 121)

Продолжение таблицы С.2

Событие ГИП	Входные параметры	Выходные параметры
изменение состояния	Операция = **BioAPI_GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_ CREATETEMPLATE** Назначение = н/д Момент = **BioAPI_GUI_ MOMENT_AFTER_END** Код Результата = **BioAPI_OK** ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_RESPONSE_ SUBOP_NEXT** ИндексОбразцаРегистрацииНаПерезахват = н/д
изменение состояния	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ VERIFY** Момент = **BioAPI_GUI_ MOMENT_BEFORE_START** Код Результата = н/д ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_RESPONSE_ SUBOP_START** ИндексОбразцаРегистрацииНаПерезахват = н/д
выполнение	Операция = **BioAPI_GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_ CAPTURE** Назначение = **BioAPI_PURPOSE_ VERIFY** Момент = **BioAPI_GUI_ MOMENT_DURING** СтепеньВыполненияПодоперации =10	Ответ = **BioAPI_GUI_RESPONSE_ PROGRESS_CONTINUE**

(Продолжение см. с. 122)

Продолжение таблицы С.2

Событие ГИП	Входные параметры	Выходные параметры
выполнение	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_VERIFY** Момент = **BioAPI_GUI_MOMENT_DURING** СтепеньВыполненияПодоперации = 40	Ответ = **BioAPI_GUI_RESPONSE_PROGRESS_CONTINUE**
выполнение	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_ SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_VERIFY** Момент = **BioAPI_GUI_MOMENT_DURING** СтепеньВыполненияПодоперации = 90	Ответ = **BioAPI_GUI_RESPONSE_PROGRESS_CONTINUE**
изменение состояния	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_VERIFY** Момент = **BioAPI_GUI_MOMENT_AFTER_END** Код Результата = **BioAPI_OK** ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_RESPONSE_SUBOP_NEXT** ИндексОбразцаРегистрацииНаПерезахват = н/д

(Продолжение см. с. 123)

Продолжение таблицы С.2

Событие ГИП	Входные параметры	Выходные параметры
изменение состояния	Операция = **BioAPI_GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_GUI_ SUBOPERATION_ PROCESS** Назначение = н/д Момент = **BioAPI_GUI_ MOMENT_BEFORE_START** Код Результата = н/д ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_RESPONSE_ SUBOP_START** ИндексОбразцаРегистрацииНаПерезахват = н/д
изменение состояния	Операция = **BioAPI_GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_GUI_ SUBOPERATION_ PROCESS** Назначение = н/д Момент = **BioAPI_GUI_ MOMENT_AFTER_END** Код Результата = **BioAPI_OK** ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_RESPONSE_ SUBOP_NEXT** ИндексОбразцаРегистрацииНаПерезахват = н/д
изменение состояния	Операция = **BioAPI_GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_GUI_ SUBOPERATION_ VERIFYMATCH** Назначение = н/д Момент = **BioAPI_GUI_ MOMENT_ BEFORE_START** Код Результата = н/д ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_RESPONSE_ SUBOP_START** ИндексОбразцаРегистрацииНаПерезахват = н/д

(Продолжение см. с. 124)

Окончание таблицы С.2

Событие ГИП	Входные параметры	Выходные параметры
изменение состояния	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_VERIFYMATCH** Назначение = н/д Момент = **BioAPI_GUI_MOMENT_AFTER_END** Код Результата = **BioAPI_OK** ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_RESPONSE_SUBOP_NEXT** ИндексОбразцаРегистрацииНаПерезахват = н/д
выбор	Операция = **BioAPI_GUI_OPERATION_ENROLL** Момент = **BioAPI_GUI_MOMENT_AFTER_END** Код Результата = **BioAPI_OK** МаксЧислоОбразцовРегистрации = 1 ВыбираемыеЭкземпляры = левый указательный палец, правый указательный палец ЗахваченныеЭкземпляры = левый указательный палец	Ответ = **BioAPI_GUI_RESPONSE_OP_COMPLETE** ВыбранныеЭкземпляры = н/д

С.7.3 Пример последовательности уведомлений о событиях ГИП для операции Регистрации (тип регистрации — Многократный-Захват, без ошибок и перезапусков, один перезахват)

Пример последовательности уведомлений о событиях ГИП представлен в таблице С.3.

(Продолжение см. с. 125)

Таблица С.3

Событие ГИП	Входные параметры	Выходные параметры
выбор	Операция = **BioAPI_GUI_OPERATION_ENROLL** Момент = **BioAPI_GUI_ MOMENT_BEFORE_START** Код Результата = н/д МаксЧислоОбразцовРегистрации = 3 ВыбираемыеЭкземпляры = левый указательный палец, правый указательный палец ЗахваченныеЭкземпляры = н/д	Ответ = **BioAPI GUI_ RESPONSE_CYCLE_ START** ВыбранныеЭкземпляры = левый указательный палец
изменение состояния	Операция = **BioAPI_GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_ GUI_SUBOPERATION_ CAPTURE** Назначение = **BioAPI_PURPOSE_ ENROLL** Момент = **BioAPI_GUI_MOMENT_ BEFORE_ START** Код Результата = н/д ИндексОбразцаРегистрации = 1	Ответ = **BioAPI_GUI_ RESPONSE_SUBOP_ START** ИндексОбразцаРегистрацииНаПерезахват = н/д
выполнение	Операция = **BioAPI_GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_GUI_ SUBOPERATION_ CAPTURE** Назначение = **BioAPI_PURPOSE ENROLL** Момент = **BioAPI_GUI_ MOMENT_ DURING** СтепеньВыполненияПодоперации = 18	Ответ = **BioAPI_ GUI_ RESPONSE_PROGRESS_ CONTINUE**

(Продолжение см. с. 126)

Продолжение таблицы С.3

Событие ГИП	Входные параметры	Выходные параметры
выполнение	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_DURING** СтепеньВыполненияПодоперации = 65	Ответ = **BioAPI_GUI_RESPONSE_PROGRESS_CONTINUE**
изменение состояния	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_AFTER_END** Код Результата = **BioAPI_OK** ИндексОбразцаРегистрации = 1	Ответ = **BioAPI_GUI_RESPONSE_SUBOP_NEXT** ИндексОбразцаРегистрацииНаПерезахват = н/д
изменение состояния	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_BEFORE_START** Код Результата = н/д ИндексОбразцаРегистрации = 2	Ответ = **BioAPI_GUI_RESPONSE_SUBOP_START** ИндексОбразцаРегистрацииНаПерезахват = н/д

(Продолжение см. с. 127)

Продолжение таблицы С.3

Событие ГИП	Входные параметры	Выходные параметры
выполнение	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_DURING** СтепеньВыполненияПодоперации = 30	Ответ = **BioAPI_GUI_RESPONSE_PROGRESS_CONTINUE**
выполнение	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_DURING** СтепеньВыполненияПодоперации = 80	Ответ = **BioAPI_GUI_RESPONSE_PROGRESS_CONTINUE**
выполнение	Операция = **BioAPI_GUI_OPERATION ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_DURING** СтепеньВыполненияПодоперации = 95	Ответ = **BioAPI_GUI_RESPONSE_PROGRESS_CONTINUE**

(Продолжение см. с. 128)

Продолжение таблицы С.3

Событие ГИП	Входные параметры	Выходные параметры
изменение состояния	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_AFTER_END** Код Результата = **BioAPI_OK** ИндексОбразцаРегистрации = 2	Ответ = **BioAPI_GUI_RESPONSE_SUBOP_NEXT** ИндексОбразцаРегистрацииНаПерезахват = н/д
изменение состояния	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_BEFORE_START** Код Результата = н/д ИндексОбразцаРегистрации = 3	Ответ = **BioAPI_GUI_RESPONSE_SUBOP_START** ИндексОбразцаРегистрацииНаПерезахват = н/д
выполнение	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_DURING** СтепеньВыполненияПодоперации = 15	Ответ = **BioAPI_GUI_RESPONSE_PROGRESS_CONTINUE**

(Продолжение см. с. 129)

Продолжение таблицы С.3

Событие ГИП	Входные параметры	Выходные параметры
выполнение	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_DURING** СтепеньВыполненияПодоперации = 70	Ответ = **BioAPI_GUI_RESPONSE_PROGRESS_CONTINUE**
выполнение	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_DURING** СтепеньВыполненияПодоперации = 90	Ответ = **BioAPI_GUI_RESPONSE_PROGRESS_CONTINUE**
изменение состояния	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_ENROLL** Момент = **BioAPI_GUI_MOMENT_AFTER_END** Код Результата = **BioAPI_OK** ИндексОбразцаРегистрации = 3	Ответ = **BioAPI_GUI_RESPONSE_SUBOP_NEXT** ИндексОбразцаРегистрацииНаПерезахват = н/д

(Продолжение см. с. 130)

Продолжение таблицы С.3

Событие ГИП	Входные параметры	Выходные параметры
изменение состояния	Операция = **BioAPI_ GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_ CREATETEMPLATE** Назначение = н/д Момент = **BioAPI_GUI_MOMENT_ BEFORE_START** Код Результата = н/д ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_RESPONSE_ SUBOP_START** ИндексОбразцаРегистрацииНаПерезахват = н/д
изменение состояния	Операция = **BioAPI_GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_ CREATETEMPLATE** Назначение = н/д Момент = **BioAPI_GUI_ MOMENT_ AFTER_END** Код Результата = **BioAPI-ERR_LOW_QUALITY_ REFERENCE_TEMPLATE** ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_RESPONSE_ RECAPTURE** ИндексОбразцаРегистрацииНаПерезахват = 2
изменение состояния	Операция = **BioAPI_GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION CAPTURE** Назначение = **BioAPI_PURPOSE_ ENROLL** Момент = **BioAPI_GUI_MOMENT_ BEFORE_START** Код Результата = н/д ИндексОбразцаРегистрации = 2	Ответ = **BioAPI_GUI_RESPONSE_ SUBOP_START** ИндексОбразцаРегистрацииНаПерезахват = н/д

(Продолжение см. с. 131)

Продолжение таблицы С.3

Событие ГИП	Входные параметры	Выходные параметры
выполнение	Операция = **BioAPI_GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_ CAPTURE** Назначение = **BioAPI_PURPOSE_ ENROLL** Момент = **BioAPI_GUI_ MOMENT_DURING** СтепеньВылолненияПодоперации = 45	Ответ = **BioAPI_GUI_RESPONSE_ PROGRESS_CONTINUE**
изменение состояния	Операция = **BioAPI_GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_GUI SUBOPERATION_ CAPTURE** Назначение = **BioAPI_PURPOSE_ ENROLL** Момент = **BioAPI_GUI_ MOMENT_AFTER_END** Код Результата = **BioAPI_OK** ИндексОбразцаРегистрации = 2	Ответ = **BioAPI_GUI_RESPONSE_ NEXT_SUBOP** ИндексОбразцаРегистрацииНаПерезахват = н/д
изменение состояния	Операция = **BioAPI_ GUI_ OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_ CREATETEMPLATE** Назначение = н/д Момент = **BioAPI_GUI_MOMENT_ BEFORE_START** Код Результата = н/д ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_RESPONSE_ START_SUBOP** ИндексОбразцаРегистрацииНаПерезахват = н/д

(Продолжение см. с. 132)

Окончание таблицы С.3

Событие ГИП	Входные параметры	Выходные параметры
изменение состояния	Операция = **BioAPI_GUI_OPERATION_ENROLL** Подоперация = **BioAPI_GUI_SUBOPERATION_CREATETEMPLATE** Назначение = н/д Момент = **BioAPI_GUI_MOMENT_AFTER_END** Код Результата = **BioAPI_OK** ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_RESPONSE_NEXT_SUBOP** ИндексОбразцаРегистрацииНаПерезахват = н/д
выбор	Операция = **BioAPI_ GUI_OPERATION_ENROLL** Момент = **BioAPI_GUI_MOMENT_AFTER_END** Код Результата = **BioAPI_OK** МаксЧислоОбразцовРегистрации = 1 ВыбираемыеЭкземпляры = левый указательный палец, правый указательный палец ЗахваченныеЭкземпляры = левый указательный палец	Ответ = **BioAPI_GUI_RESPONSE_OP_COMPLETE** ВыбранныеЭкземпляры = н/д
выбор	Операция = **BioAPI_ GUI_OPERATION_INDENTIFY** Момент = **BioAPI_GUI_MOMENT_BEFORE_START** Код Результата = н/д МаксЧислоОбразцовРегистрации = н/д ВыбираемыеЭкземпляры = правый ЗахваченныеЭкземпляры = н/д	Ответ = **BioAPI_GUI_RESPONSE_START_CYCLE_** ВыбранныеЭкземпляры = правый

(Продолжение см. с. 133)

С.7.4 Пример последовательности уведомлений о событиях ГИП для операции Идентификации (без ошибок и перезапусков)

Пример последовательности уведомлений о событиях ГИП представлен в таблице С.4.

Таблица С.4

Событие ГИП	Входные параметры	Выходные параметры
изменение состояния	Операция = **BioAPI_GUI_ OPERATION_INDENTIFY** Подоперация = **BioAPI_ GUI_SUBOPERATION_ CAPTURE** Назначение = **BioAPI_PURPOSE_INDENTIFY** Момент = **BioAPI_GUI_MOMENT_ BEFORE_ START** Код Результата = н/д ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_ RESPONSE_START_ SUBOP** ИндексОбразцаРегистрацииНаПерезахват = н/д
выполнение	Операция = **BioAPI_GUI_ OPERATION_INDENTIFY** Подоперация = **BioAPI_GUI_ SUBOPERATION_ CAPTURE** Назначение = **BioAPI_PURPOSE_ INDENTIFY** Момент = **BioAPI_GUI_ MOMENT_DURING** СтепеньВыполненияПодоперации = 32	Ответ = **BioAPI_ GUI_ RESPONSE_ CONTINUE_SUBOP**
выполнение	Операция = **BioAPI_GUI_ OPERATION_INDENTIFY** Подоперация = **BioAPI_GUI_ SUBOPERATION_ CAPTURE** Назначение = **BioAPI_PURPOSE_ INDENTIFY** Момент = **BioAPI_GUI_ MOMENT_DURING** СтепеньВыполненияПодоперации = 70	Ответ = **BioAPI_ GUI_ RESPONSE_ CONTINUE_SUBOP**

(Продолжение см. с. 134)

Продолжение таблицы С.4

Событие ГИП	Входные параметры	Выходные параметры
изменение состояния	Операция = **BioAPI_GUI_OPERATION_IDENTIFY** Подоперация = **BioAPI_GUI_SUBOPERATION_CAPTURE** Назначение = **BioAPI_PURPOSE_IDENTIFY** Момент = **BioAPI_GUI_MOMENT_AFTER_END** Код Результата = **BioAPI_OK** ИндексОбразцаРегистрации = 1	Ответ = **BioAPI_GUI_RESPONSE_NEXT_SUBOP** ИндексОбразцаРегистрацииНаПерезахват = н/д
изменение состояния	Операция = **BioAPI_ GUI_OPERATION_IDENTIFY** Подоперация = **BioAPI_GUI_SUBOPERATION_IDENTIFYMATCH** Назначение = н/д Момент = **BioAPI_GUI_MOMENT_BEFORE_START** Код Результата = н/д ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_RESPONSE_STAR_SUBOP** ИндексОбразцаРегистрацииНаПерезахват = н/д
выполнение	Операция = **BioAPI_ GUI_OPERATION_IDENTIFY** Подоперация = **BioAPI_GUI SUBOPERATION_IDENTIFYMATCH** Назначение = н/д Момент = **BioAPI_GUI_MOMENT_DURING** СтепеньВыполненияПодоперации = 9	Ответ = **BioAPI_GUI_RESPONSE_PROGRESS_CONTINUE**

(Продолжение см. с. 135)

Продолжение таблицы С.4

Событие ГИП	Входные параметры	Выходные параметры
выполнение	Операция = **BioAPI_ GUI_ OPERATION_IDENTIFY** Подоперация = **BioAPI_GUI SUBOPERATION_IDENTIFYMATCH** Назначение = н/д Момент = **BioAPI_GUI_ MOMENT_DURING** СтепеньВыполненияПодоперации = 30	Ответ = **BioAPI_GUI_RESPONSE_ PROGRESS_CONTINUE**
выполнение	Операция = **BioAPI_ GUI_ OPERATION_IDENTIFY** Подоперация = **BioAPI_GUI_ SUBOPERATION_IDENTIFYMATCH** Назначение = н/д Момент = **BioAPI_GUI_ MOMENT_DURING** СтепеньВыполненияПодоперации = 64	Ответ = **BioAPI_GUI_RESPONSE_ PROGRESS_CONTINUE**
выполнение	Операция = **BioAPI_ GUI_ OPERATION_IDENTIFY** Подоперация = **BioAPI_GUI SUBOPERATION_IDENTIFYMATCH** Назначение = н/д Момент = **BioAPI_GUI_ MOMENT_DURING** СтепеньВыполненияПодоперации = 83	Ответ = **BioAPI_GUI_RESPONSE_ PROGRESS_CONTINUE**

(Продолжение см. с. 136)

Окончание таблицы С.4

Событие ГИП	Входные параметры	Выходные параметры
выполнение	Операция = **BioAPI_ GUI_ OPERATION_IDENTIFY** Подоперация = **BioAPI_GUI_ SUBOPERATION_IDENTIFYMATCH** Назначение = н/д Момент = **BioAPI_GUI_ MOMENT_DURING** СтепеньВыполненияПодоперации = 98	Ответ = **BioAPI_GUI_RESPONSE_ PROGRESS_CONTINUE**
изменения состояния	Операция = **BioAPI_ GUI_ OPERATION_IDENTIFY** Подоперация = **BioAPI_GUI_ SUBOPERATION_IDENTIFYMATCH** Назначение = н/д Момент = **BioAPI_GUI_ MOMENT_AFTER_END** Код Результата = **BioAPI_OK** ИндексОбразцаРегистрации = н/д	Ответ = **BioAPI_GUI_RESPONSE_ SUBOR_NEXT** ИндексОбразцаРегистрацииНаПерезахват = н/д
выбор	Операция = **BioAPI_ GUI_ OPERATION_IDENTIFY** Момент = **BioAPI_GUI_ MOMENT_AFTER_END** Код Результата = **BioAPI_OK** МаксЧислоОбразцовРегистрации = н/д ВыбираемыеЭкземпляры = правый ЗахваченныеЭкземпляры = правый	Ответ = **BioAPI_GUI_RESPONSE_ OP_COMPLETE** ВыбранныеЭкземпляры = н/д

(Продолжение см. с. 137)

Пункт С.8.4. Первый абзац изложить в новой редакции:

«Существует множество способов взаимодействия биометрических компонентов и компонентов смарт-карты (программные и аппаратные средства) для выполнения операции СНК. Один из потоков процесса высокого уровня для такой операции изображен на рисунке С.3».

Приложение D дополнить подразделом — D.3:

«D.3 Последовательность вызовов с ГИП (БиоАПИ 2.1)

Примеры в пунктах D.3.1, D.3.2 применимы только при использовании БиоАПИ версии 2.1.

D.3.1 Регистрация с ГИП

Если ПБУ поддерживает функцию BioAPI_SunscribeToGUIEvents, и тип регистрации, предоставленный ПБУ, является Тестировать-Верифицировать, то последовательность вызовов при регистрации может быть следующая:

```
int ApplicationControlledEnrollFunction()
    {
        BioAPI_RETURN bioReturn;
        BioAPI_BIR_HANDLE EnrolledTemplate;
        BioAPI_HANDLE BSPHandle;
        BioAPI_GUI_SELECT_EVENT_HANDLER   EnrollGUI-
            SelectEventHandler;
        BioAPI_GUI_STATE_EVENT_HANDLER EnrollGUIState-
            EventHandler;
        BioAPI_GUI_PROGRESS_EVENT_HANDLER EnrollGUI-
            ProgressEventHandler;
        // Задание обработчиков событий ГИП.
        bioReturn = BioAPI_SubscribeToGUIEvents(
            NULL,
            NULL,
            &BSPHandle,
            EnrollGUISelectEventHandler,
            NULL,
            EnrollGUIStateEventHandler,
            NULL,
            EnrollGUIProgressEventHandler,
            NULL);
        if(bioReturn != BioAPI_OK)
        {
          printf("Код Ошибки БиоАПИ: %d\n", bioReturn);
          return 0;
        }
```

(Продолжение см. с. 138)

```
        // При вызове функции BioAPI_Enroll события ГИП
        // будут вызваны перед возращением значения этой функции.
        bioReturn = BioAPI_Enroll(
                BSPHandle,
                BioAPI_PURPOSE_ENROLL
                NULL,
                &EnrolledTemplate,
                NULL,-1, NULL, NULL);
        // После того, как функция BioAPI_Enroll возвратила значе-
        ние,
        // приложение может удалить записи обратного вызова путем
        // вызова функции BioAPI_UnsubscribeFromGUIEvents.
        BioAPI_UnsubscribeFromGUIEvents (
                NULL,
                NULL,
                &BSPHandle,
                EnrollGUISelectEventHandler,
                NULL,
                EnrollGUIStateEventHandler,
                NULL,
                EnrollGUIProgressEventHandler,
                NULL);
        if(bioReturn != BioAPI_OK)
        {
                printf("Код Ошибки БиоАПИ: %d\n", bioReturn);
                return 0;
        }
        return 1;
}
BIOAPI_RETURN CALLBACK EnrollGUISelectEventHandler
        (const BioAPI_UUID *BSPUuid,
        BioAPI_UNIT_ID UnitID,
        const BioAPI_HANDLE *BSPHandle,
        uint32_t EnrollType,
        const void *GUISelectEventHandlerCtx,
        BioAPI_GUI_OPERATION Operation,
        BioAPI_GUI_MOMENT Moment,
        BioAPI_RETURN ResultCode,
        int32_t MaxNumEnrollSamples,
        BioAPI_BIR_SUBTYPE_MASK SelectableInstances,
        BioAPI_BIR_SUBTYPE_MASK*SeIectedInstances,
```

(Продолжение см. с. 139)

```
        BioAPI_BIR_SUBTYPE_MASK CapturedInstances,
        const uint8_t *Text,
        BioAPI_GUI_RESPONSE *Response)
{
// Показать ГИП пользователю для выбора конкретного
// биометрического подтипа с целью его последующей регистрации.
// Задать параметр SelectedInstances, основанный на подтипе(ах),
// выбранном(ых) пользователем.
// Если событие выбора ГИП сгенерировано перед циклом
подопераций
// (Moment = BioAPI_GUI_MOMENT_BEFORE_START), то
необходимо задать
// параметру Response значение BioAPI_GUI_RESPONSE_CYCLE_
START,
// если пользователь хочет приступить к регистрации, или
// значение BioAPI_GUI_RESPONSE_OP_CANCEL, если
// пользователь хочет прервать регистрацию.
// Если событие выбора ГИП сгенерировано после цикла подопераций
// (Moment = BioAPI_GUI_MOMENT_AFTER END), то необходимо
задать
// параметру Response значение BioAPI_GUI_RESPONSE OP_
COMPLETE,
// если пользователь удовлетворен регистрацией, или
// значение BioAPI_GUI_RESPONSE_CYCLE_RESTART, если
// пользователь хочет повторить цикл, или
// значение BioAPI_GUI_RESPONSE_OP_CANCEL, если
// пользователь хочет прервать регистрацию.
        return BioAPI_OK;
}
BioAPI_RETURN CALLBACK EnrollGUIStateEventHandler
        (const BioAPI_UUID *BSPUuid,
        BioAPI_UNIT_ID UnitID,
        const BioAPI_HANDLE *BSPHandle,
        const void *GUIStateEventHandlerCtx,
        BioAPI_GUI_OPERATION Operation,
        BioAPI_GUI_SUBOPERATION Suboperation,
        BioAPI_BIR_PURPOSE Purpose,
        BioAPI_GUI_MOMENT Moment,
        BioAPI_RETURN ResultCode,
        int32_t EnrollSampleIndex,
        const BioAPI_GUI_BITMAP_ARRAY *Bitmaps,
```

(Продолжение см. с. 140)

```
        const uint8_t *Text,
        BioAPI_GUI_RESPONSE *Response,
        int32_t *EnrollSampleIndexToRecapture)
{
// Показать ГИП, основанный на значениях параметров
// Operation, Suboperation, Purpose, Moment, и
// ResultCode, предоставленных ПБУ. Если Текст (Text)
// был предоставлен, он может быть отображен.
        switch(Suboperation) {
        case BioAPI_GUI_SUBOPERATION_CAPTURE:
              switch(Purpose) {
              case BioAPI_PURPOSE_ENROLL:
              case BioAPI_PURPOSE_ENROLL_FOR_VERIFICATION_ONLY:
              case BioAPI_PURPOSE_ENROLL_FOR_
              IDENTICATION_ONLY:
                  switch(Moment) {
                  caseBioAPI_GUI_MOMENT_BEFORE_START:
                      // Пример сообщения: “Нажмите кнопку [Захват]
                      // для начала захвата изображения”.
                      break;
                  case BioAPI_GUI_MOMENT_AFTER_END:
                      // Пример сообщения 1: “Захват прошел
                      успешно”.
                      // Пример сообщения 2: “Захват не удался”.
                      break;
                  }
                  break;
              case BioAPI_PURPOSE_VERIFY:
                 switch(Moment) {
                 case BioAPI_GUI_MOMENT_BEFORE_START:
                      // Пример сообщения: “Нажмите кнопку [Захват],
                      чтобы
                      //начать захват изображения для операции
                      верификации”.
                      break;
                 case BioAPI_GUI_MOMENT_AFTER_END:
                      // Пример сообщения 1: “Захват прошел
                      успешно”.
                      // Пример сообщения 2: “Захват не удался”.
                      break;
```

(Продолжение см. с. 141)

```
            }
            break;
        }
        break;
    // другие подоперации
    }
    return BioAPI_OK;
}
BioAPI_RETURN CALLBACK EnrollGUIProgressEventHandler
    (const BioAPI_UUID *BSPUuid,
    BioAPI_UNIT_ID UnitID,
    const BioAPI_HANDLE *BSPHandle,
    const void *GUIProgressEventHandlerCtx,
    BioAPI_GUI_OPERATION Operation,
    BioAPI_GUI_SUBOPERATION Suboperation,
    BioAPI_BIR_PURPOSE Purpose,
    BioAPI_GUI_MOMENT Moment,
    uint8_t SuboperationProgress,
    const BioAPI_GUI_BITMAP_ARRAY *Bitmaps,
    const uint8_t *Text,
    BioAPI_GUI_RESPONSE *Response)
{
    // Вывести потоковые данные на экран.
}
```

Как только шаблон возвращен, он может быть помещен в хранилище данных для извлечения в дальнейшем с целью верификации (см. D.2).

D.3.2 Выполнение верификации с ГИП

Если ПБУ поддерживает функцию BioAPI_SubscribeToGUIEvents, то последовательность вызовов при верификации с помощью BioAPI_Verify может быть следующей:

```
int ApplicationControlledVerifyFunction ()
{
    BioAPI_INPUT_BIR birEnroll;
    BioAPI_BIR_HEADER birHeader;
    int MaxFMR, AchievedFMR;
    BioAPI_BOOL bPrecedence, bResponse;
    BioAPI_HANDLE BSPHandle;
    BioAPI_GUI_SELECT_EVENT_HANDLER VerifyGUISelectEventHandler;
    BioAPI_GUI_STATE_EVENT_HANDLER VerifyGUIStateEventHandler;
    BioAPI_GUI_PROGRESS_EVENT_HANDLER VerifyGUIProgressEvent-
        Handler;
```

(Продолжение см. с. 142)

```
// Задание событий ГИП.
bioReturn = BioAPI_SubscribeToGUIEvents(
        NULL,
        NULL,
        &BSPHandle,
        VerifyGUISelectEventHandler,
        NULL,
        VerifyGUIStateEventHandler,
        NULL,
        VerifyGUIProgressEventHandler,
        NULL);
// При вызове функции BioAPI_Verify обратный вызов изменения
// состояния ГИП будет вызван перед возращением значения этой
функции.
MaxFMR=1;
bioReturn = BioAPI_Verify( BSPHandle,
        &MaxFMR,
        NULL,
        &bPrecedence,
        &birEnroll,
        NULL,
        &bResponse,
        &AchievedFMR,
        NULL,
        NULL,
        -1,
        NULL);
free(birEnroll.lnputBIR.BIR);
// После того как функция BioAPI_Verify возвратила значение,
// приложение может удалить записи обратного вызова путем
// вызова функции BioAPI_UnsubscribeFromGUIEvents.
BioAPI_UnsubscribeFromGUIEvents (
        NULL,
        NULL,
        &BSPHandle,
        VerifyGUISelectEventHandler,
        NULL,
        VerifyGUIStateEventHandler,
        NULL,
        VerifyGUIProgressEventHandler,
        NULL);
```

(Продолжение см. с. 143)

```
if(bioReturn != BioAPI_OK)
{
        printf("Код Ошибки БиоАПИ: %d\n", bioReturn);
        return 0;
}
else if(bResponse != BioAPI_TRUE)
{
        printf("Нет Совпадений\n");
        return 0;
}
else
       {
        printf("Совпадение найдено\n");
        return 1;
       }
}
BIOAPI_RETURN CALLBACK VerifyGUISelectEventHandler
        (const BioAPI_UUID *BSPUuid,
        BioAPI_UNIT_ID UnitID,
        const BioAPI_HANDLE *BSPHandle,
        uint32_t EnrollType,
        const void *GUISelectEventHandlerCtx,
        BioAPI_GUI_OPERATION Operation,
        BioAPI_GUI_MOMENT Moment,
        BioAPI_RETURN ResuItCode,
        int32_t MaxNumEnrollSamples,
        BioAPI_BIR_SUBTYPE_MASK SelectableInstances,
        BioAPI_BIR_SUBTYPE_MASK *SelectedInstances,
        BioAPI_BIR_SUBTYPE_MASK CapturedInstances,
        const uint8_t *Text,
        BioAPI_GUI_RESPONSE *Response)
{
// Показать ГИП пользователю для выбора конкретного
// биометрического подтипа с целью его последующей регистрации.
// Задать параметр SelectedInstances, основанный на подтипе(ах),
// выбранном(ых) пользователем.
// Если событие выбора ГИП сгенерировано перед циклом
подопераций
// (Moment = BioAPI_GUI_MOMENT_BEFORE_START), то
необходимо задать
```

(Продолжение см. с. 144)

```
// параметру Response значение BioAPI_GUI_RESPONSE_ CYCLE_
START, если
// пользователь хочет приступить к регистрации, или
// значение BioAPI_GUI_RESPONSE_OP_CANCEL, если
// пользователь хочет прервать регистрацию.
// Если событие выбора ГИП сгенерировано после цикла подопераций
// (Moment = BioAPI_GUI_MOMENT_AFTER_END), то необходимо
задать
// параметру Response значение BioAPI_GUI_RESPONSE_OP_
COMPLETE, если
// пользователь удовлетворен регистрацией, или
// значение BioAPI_GUI_RESPONSE_CYCLE_RESTART, если
// пользователь хочет повторить цикл, или
// значение BioAPI_GUI_RESPONSE_OP_CANCEL, если
// пользователь хочет прервать регистрацию.
        return BioAPI_OK;
}
BioAPI_RETURN CALLBACK VerifyGUIStateEventHandler
        (const BioAPI_UUID *BSPUuid,
        BioAPI_UNIT_ID UnitID,
        const BioAPI_HANDLE *BSPHandle,
        const void *GUIStateEventHandlerCtx,
        BioAPI_GUI_OPERATION Operation,
        BioAPI_GUI_SUBOPERATION Suboperation,
        BioAPI_BIR_PURPOSE Purpose,
        BioAPI_GUI_MOMENT Moment,
        BioAPI_RETURN ResultCode,
        int32_t EnrollSampleIndex,
        const BioAPI_GUI_BITMAP_ARRAY *Bitmaps,
        const uint8_t *Text,
        BioAPI_GUI_RESPONSE *Response,
        int32_t *EnrollSampleIndexToRecapture)
{
// Показать ГИП, основанный на значениях параметров
// Operation, Suboperation, Purpose, Moment, и
// ResultCode, предоставленных ПБУ. Если Текст (Text)
// был предоставлен, он может быть отображен.
        Switch (Suboperation){
        caseBioAPI_GUI_SUBOPERATION_CAPTURE:
```

(Продолжение см. с. 145)

```
            switch(Moment){
            case BioAPI_GUI MOMENT_BEFORE_START:
                    // Пример сообщения: “Нажмите кнопку [Захват]
                    // для начала захвата изображения”.
                    break;
            caseBioAPI_GUI_MOMENT_AFTER_END:
                    // Пример сообщения 1: “Захват прошел успешно”.
                    // Пример сообщения 2: “Захват не удался”.
                    break;
            }
            break;
        case BioAPI_GUI_SUBOPERATION_VERIFYMATCH:
            switch(Moment) {
            caseBioAPI_GUI_MOMENT_BEFORE_START:
                    // Сообщения не требуются.
                    break;
            case BioAPI GUI_MOMENT AFTER_END:
                    // Пример сообщения 1: “Проверка прошла
                    успешно”.
                    // Пример сообщения 2: “Проверка не удалась”.
                    break;
            }
            break;
        // другие подоперации
        }
        return BioAPI_OK;
}
BioAPI_RETURN CALLBACK VerifyGUIProgressEventHandler
        (const BioAPI_UUID *BSPUuid,
        BioAPI_UNIT_ID UnitID,
        const BioAPI_HANDLE *BSPHandle,
        const void *GUIProgressEventHandlerCtx,
        BioAPI_GUI_OPERATION Operation,
        BioAPI_GUI_SUBOPERATION Suboperation,
        BioAPI_BIR_PURPOSE Purpose,
        BioAPI_GUI_MOMENT Moment,
        uint8_t SuboperationProgress,
        const BioAPI_GUI_BITMAP_ARRAY *Bitmaps,
```

(Продолжение см. с. 146)

```
        const uint8_t * Text,
        BioAPI_GUI_RESPONSE *Response)
{
        // Вывести потоковые данные на экран.
        return BioAPI_OK;
}  »
```

(ИУС № 9 2012 г.)