ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р ИСО/МЭК 24709-1 — 2009

Автоматическая идентификация

ИДЕНТИФИКАЦИЯ БИОМЕТРИЧЕСКАЯ

Испытания на соответствие биометрическому программному интерфейсу (БиоАПИ)

Часть 1

Методы и процедуры

ISO/IEC 24709-1:2007 Information technology — Conformance testing for the biometric application programming interface (BioAPI) — Part 1: Methods and procedures (IDT)

Издание официальное

БЗ 4—2009/148

Москва
Стандартинформ
2011

gost.e-ecolog.ru

Предисловие

Цели и принципы стандартизации в Российской Федерации установлены Федеральным законом от 27 декабря 2002 г. № 184-ФЗ «О техническом регулировании», а правила применения национальных стандартов Российской Федерации — ГОСТ Р 1.0—2004 «Стандартизация в Российской Федерации. Основные положения»

Сведения о стандарте

1 ПОДГОТОВЛЕН Научно-исследовательским и испытательным центром биометрической техники Московского государственного технического университета имени Н. Э. Баумана (НИИЦ БТ МГТУ им. Н. Э. Баумана) на основе собственного аутентичного перевода на русский язык стандарта, указанного в пункте 4

2 ВНЕСЕН Управлением технического регулирования и стандартизации Федерального агентства по техническому регулированию и метрологии

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 8 декабря 2009 года № 590-ст

4 Настоящий стандарт идентичен международному стандарту ИСО/МЭК 24709-1—2007 «Информационные технологии. Испытания на соответствие биометрическому программному интерфейсу (БиоАПИ). Часть 1. Методы и процедуры» (ISO/IEC 24709-1—2007 «Information technology — Conformance testing for the biometric application programming interface (BioAPI). Part 1: Methods and procedures»).

Наименование настоящего стандарта изменено относительно наименования указанного международного стандарта для приведения в соответствие с ГОСТ Р 1.5 (подраздел 3.5) и учета его принадлежности к группе стандартов «Автоматическая идентификация».

При применении настоящего стандарта рекомендуется использовать вместо ссылочных международных (региональных) стандартов соответствующие им национальные стандарты, сведения о которых приведены в дополнительном приложении ДА

5 ВВЕДЕН ВПЕРВЫЕ

Информация об изменениях к настоящему стандарту публикуется в ежегодно издаваемом информационном указателе «Национальные стандарты», а текст изменений и поправок — в ежемесячно издаваемых информационных указателях «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ежемесячно издаваемом информационном указателе «Национальные стандарты». Соответствующая информация, уведомление и тексты размещаются также в информационной системе общего пользования — на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет

© Стандартинформ, 2011

Настоящий стандарт не может быть полностью или частично воспроизведен, тиражирован и распространен в качестве официального издания без разрешения Федерального агентства по техническому регулированию и метрологии

Содержание

1 Область применения . . . 1
2 Соответствие . . . 1
3 Нормативные ссылки . . . 2
4 Термины и определения . . . 2
5 Сокращения . . . 3
6 Методика испытаний на соответствие . . . 4
6.1 Общие положения . . . 4
6.2 Модели испытаний на соответствие . . . 7
6.3 Абстрактная испытательная машина . . . 10
7 Общие свойства языка утверждений . . . 12
7.1 Общие положения . . . 12
7.2 Переменные . . . 14
7.3 Встроенные переменные . . . 15
7.4 Представление целых чисел . . . 15
7.5 Представление логических значений . . . 16
7.6 Представление универсальных уникальных идентификаторов . . . 16
7.7 Представление набора байтов . . . 16
7.8 XML-документы . . . 16
8 Элементы языка утверждений . . . 16
8.1 Элемент <package> . . . 16
8.2 Элемент <assertion> (дочерний для элемента <package>) . . . 17
8.3 Элемент <input> (дочерний для элемента <assertion>) . . . 18
8.4 Элемент <invoke> (дочерний для элемента <assertion>) . . . 19
8.5 Элемент <input> (дочерний для элемента <invoke>) . . . 20
8.6 Элемент <output> (дочерний для элемента <invoke>) . . . 21
8.7 Элемент <return> (дочерний для элемента <invoke>) . . . 21
8.8 Элемент <bind> (дочерний для элемента <assertion>) . . . 22
8.9 Элемент <activity> (дочерний для элемента <package>) . . . 22
8.10 Элемент <input> (дочерний для элемента <activity>) . . . 28
8.11 Элемент <output> (дочерний для элемента <activity>) . . . 28
8.12 Элемент <set> . . . 29
8.13 Элемент <add> . . . 29
8.14 Элемент <subtract> . . . 30
8.15 Элемент <invoke> (дочерний для элемента <activity>) . . . 31
8.16 Элемент <only_if> . . . 33
8.17 Элемент <wait_until> . . . 34
8.18 Элемент <assert_condition> . . . 35
8.19 Элемент <and> . . . 36
8.20 Элемент <or> . . . 37
8.21 Элемент <xor> . . . 37
8.22 Элемент <not> . . . 38
8.23 Элемент <equal_to> . . . 38
8.24 Элемент <not_equal_to> . . . 39
8.25 Элемент <greater_than> . . . 39
8.26 Элемент <greater_than_or_equal_to> . . . 40
8.27 Элемент <less_than> . . . 40
8.28 Элемент <less_than_or_equal_to> . . . 40
8.29 Элемент <same_as> . . . 40
8.30 Элемент <not_same_as> . . . 41
8.31 Элемент <existing> . . . 41
8.32 Элемент <not_existing> . . . 41
9 Функции стандартного интерфейса . . . 41
9.1 Общие положения . . . 41
9.2 Группы параметров . . . 45

9.3 Функция BioAPI_Init . . . 66
9.4 Функция BioAPI_Terminate . . . 67
9.5 Функция BioAPI_GetFrameworkInfo . . . 67
9.6 Функция BioAPI_EnumBSPs . . . 68
9.7 Функция BioAPI_BSPLoad . . . 69
9.8 Функция BioAPI_BSPUnload . . . 71
9.9 Функция BioAPI_BSPAttach . . . 72
9.10 Функция BioAPI_BSPDetach . . . 74
9.11 Функция BioAPI_QueryUnits . . . 74
9.12 Функция BioAPI_EnumBFPs . . . 76
9.13 Функция BioAPI_QueryBFPs . . . 78
9.14 Функция BioAPI_ControlUnit . . . 79
9.15 Функция BioAPI_FreeBIRHandle . . . 81
9.16 Функция BioAPI_GetBIRFromHandle . . . 82
9.17 Функция BioAPI_GetHeaderFromHandle . . . 83
9.18 Функция BioAPI_EnableEvents . . . 84
9.19 Функция BioAPI_SetGUICallbacks . . . 85
9.20 Функция BioAPI_Capture . . . 86
9.21 Функция BioAPI_CreateTemplate . . . 88
9.22 Функция BioAPI_Process . . . 91
9.23 Функция BioAPI_ProcessWithAuxBIR . . . 92
9.24 Функция BioAPI_VerifyMatch . . . 94
9.25 Функция BioAPI_IdentifyMatch . . . 97
9.26 Функция BioAPI_Enroll . . . 100
9.27 Функция BioAPI_Verify . . . 102
9.28 Функция BioAPI_Identify . . . 105
9.29 Функция BioAPI_Import . . . 108
9.30 Функция BioAPI_PresetIdentifyPopulation . . . 110
9.31 Функция BioAPI_DbOpen . . . 111
9.32 Функция BioAPI_DbClose . . . 113
9.33 Функция BioAPI_DbCreate . . . 114
9.34 Функция BioAPI_DbDelete . . . 115
9.35 Функция BioAPI_DbSetMarker . . . 116
9.36 Функция BioAPI_DbFreeMarker . . . 117
9.37 Функция BioAPI_DbStoreBIR . . . 118
9.38 Функция BioAPI_DbGetBIR . . . 119
9.39 Функция BioAPI_DbGetNextBIR . . . 121
9.40 Функция BioAPI_DbDeleteBIR . . . 122
9.41 Функция BioAPI_SetPowerMode . . . 123
9.42 Функция BioAPI_SetIndicatorStatus . . . 124
9.43 Функция BioAPI_GetIndicatorStatus . . . 125
9.44 Функция BioAPI_CalibrateSensor . . . 126
9.45 Функция BioAPI_Cancel . . . 127
9.46 Функция BioAPI_Free . . . 127
9.47 Функция BioAPI_Util_InstallBSP . . . 128
9.48 Функция BioAPI_Util_InstallBFP . . . 129
9.49 Функция BioSPI_BSPLoad . . . 131
9.50 Функция BioSPI_BSPUnload . . . 132
9.51 Функция BioSPI_BSPAttach . . . 133
9.52 Функция BioSPI_BSPDetach . . . 135
9.53 Функция BioSPI_QueryUnits . . . 135
9.54 Функция BioSPI_QueryBFPs . . . 137
9.55 Функция BioSPI_ControlUnit . . . 138
9.56 Функция BioSPI_FreeBIRHandle . . . 139
9.57 Функция BioSPI_GetBIRFromHandle . . . 140

9.58 Функция BioSPI_GetHeaderFromHandle . . . 140
9.59 Функция BioSPI_EnableEvents . . . 141
9.60 Функция BioSPI_SetGUICallback . . . 142
9.61 Функция BioSPI_Capture . . . 142
9.62 Функция BioSPI_CreateTemplate . . . 144
9.63 Функция BioSPI_Process . . . 145
9.64 Функция BioSPI_ProcessWithAuxBIR . . . 146
9.65 Функция BioSPI_VerifyMatch . . . 147
9.66 Функция BioSPI_IdentifyMatch . . . 149
9.67 Функция BioSPI_Enroll . . . 150
9.68 Функция BioSPI_Verify . . . 152
9.69 Функция BioSPI_Identify . . . 154
9.70 Функция BioSPI_Import . . . 156
9.71 Функция BioSPI_PresentIdentifyPopulation . . . 157
9.72 ФункцияBioSPI_DbOpen . . . 158
9.73 Функция BioSPI_DbClose . . . 159
9.74 Функция BioSPI_DbCreate . . . 159
9.75 Функция BioSPI_DbDelete . . . 160
9.76 Функция BioSPI_DbSetMarker . . . 161
9.77 Функция BioSPI_DbFreeMarker . . . 161
9.78 Функция BioSPI_DbStoreBIR . . . 162
9.79 Функция BioSPI_DbGetBIR . . . 163
9.80 Функция BioSPI_DbGetNextBIR . . . 164
9.81 Функция BioSPI_DbDeleteBIR . . . 165
9.82 Функция BioSPI_SetPowerMode . . . 166
9.83 Функция BioSPI_SetIndicatorStatus . . . 166
9.84 Функция BioSPI_GetIndicatorStatus . . . 167
9.85 Функция BioSPI_CalibrateSensor . . . 168
9.86 Функция BioSPI_Cancel . . . 168
9.87 Функция BioSPI_Free . . . 169
9.88 Функция BioAPI_EventHandler . . . 169
9.89 Функция BioAPI_GUI_STATE_CALLBACK . . . 170
9.90 Функция BioAPI_GUI_STREAMING_CALLBACK . . . 172
9.91 Функция BioSPI_EventHandler . . . 173
9.92 Функция BioSPI_GUI_STATE_CALLBACK . . . 175
9.93 Функция BioSPI_GUI_STREAMING_CALLBACK . . . 176
9.94 Функция BioSPI_BFP_ENUMERATION_HANDLER . . . 177
9.95 Функция BioSPI_MEMORY_FREE_HANDLER . . . 179
10 Встроенные переменные . . . 180
10.1 Переменные, имеющие постоянное значение . . . 180
10.2 Переменные, имеющие значения, которые могут изменяться . . . 185
11 Протокол испытания . . . 185
12 Отчет о результатах испытаний . . . 187
13 Комплект тестов на соответствие BioAPI . . . 188
13.1 Общие положения . . . 188
13.2 Структура комплекта тестов на соответствие биометрическому программному интерфейсу . . . 188
Приложение А(обязательное) XML-схема языка утверждения . . . 190
Приложение В(обязательное) Схема языка утверждения ASN.1 . . . 194
Приложение С(обязательное) XML-схема протокола теста . . . 197
Приложение D(справочное) Руководство по проведению испытаний . . . 199
Приложение Е(справочное) Программа оценки соответствия BioAPI . . . 203
Приложение F(справочное) XML-диаграммы языка утверждений . . . 207
Приложение ДА (справочное) Сведения о соответствии ссылочных международных стандартов ссылочным национальным стандартам Российской Федерации . . . 226
Библиография . . . 227

Введение

Международные организации ИСО и МЭК образуют специализированную систему стандартизации во всем мире. Национальные органы по стандартизации, являющиеся членами ИСО и МЭК, принимают участие в разработке международных стандартов через технические комитеты (ТК), учрежденные соответствующей организацией для решения вопросов в отдельных областях технической деятельности. ТК ИСО и МЭК сотрудничают в областях взаимного интереса. Другие международные организации, правительственные и неправительственные, взаимодействуя с ИСО и МЭК, также принимают участие в работе. В области информационных технологий ИСО и МЭК организовали соединенный технический комитет (СТК) — ИСО/МЭК СТК 1.

Проекты международных стандартов составляются в соответствии с правилами, определенными во второй части Руководящих указаний ИСО/МЭК.

Главной задачей СТК является подготовка международных стандартов. Проекты международных стандартов, официально принятые СТК, распространяются по национальным органам по стандартизации для голосования. Для публикации международного стандарта необходимо, чтобы он был одобрен по меньшей мере 75 % национальных органов по стандартизации, участвующих в голосовании.

Следует обратить внимание на то, что некоторые части настоящего стандарта могут являться объектом патентных прав. ИСО и МЭК не несут ответственность за определение некоторых или всех подобных патентных прав.

Стандарт ИСО/МЭК 24709-1 подготовлен соединенным техническим комитетом ИСО/МЭК СТК1(Информационные технологии), подкомитетом ПК 37 (Биометрия).

Комплекс стандартов ИСО/МЭК 24709 состоит из следующих частей, объединенных общим заголовком «Информационные технологии. Испытания на соответствие биометрического программного интерфейса»:

- часть 1 — методы и процедуры;
- часть 2 — формальные тестовые утверждения для поставщиков биометрических услуг (ПБУ);
- часть 3 — формальные тестовые утверждения для инфраструктур BioAPI;
- часть 4 — формальные тестовые утверждения для биометрических приложений.

Настоящий стандарт устанавливает метод испытаний биометрических программных интерфейсов на соответствие требованиям ИСО/МЭК 19784-1. В настоящем стандарте приведены три модели испытаний на соответствие, позволяющие проводить испытания следующих компонентов BioAPI: приложения, инфраструктуры и поставщика биометрической услуги (ПБУ). Помимо этого, стандарт устанавливает язык утверждений, который используется для определения формальных тестовых утверждений. Фактические тестовые утверждения для каждого из трех компонентов биометрического программного интерфейса приведены в последующих частях комплекса стандартов ИСО/МЭК 24709.

Помимо определения моделей испытаний на соответствие, в стандарте ИСО/МЭК 19784-1 приведены руководства, в которых определены общие концепции создания программы оценки и сертификации соответствия спецификации BioAPI, а также управления этой программой. Данные руководства определяют виды предпринимаемых мер, ответственность, услуги и документы, рекомендуемые для проведения оценки и сертификации приложений на соответствие спецификации BioAPI. В ИСО/МЭК 19784-1 также приведено руководство по созданию методики полной оценки соответствия приложения спецификации BioAPI.

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Автоматическая идентификация

ИДЕНТИФИКАЦИЯ БИОМЕТРИЧЕСКАЯ

Испытания на соответствие биометрическому программному интерфейсу (БиоАПИ)

Часть 1

Методы и процедуры

Automatic identification. Biometric identification. Conformance testing for the biometric application programing interface (BioAPI). Part 1: Methods and procedures

Дата введения — 2011 — 01 — 01

1 Область применения

1.1 Настоящий стандарт устанавливает понятия, структуру, методы испытаний и критерии, необходимые для проведения испытаний биометрической продукции на соответствие спецификации BioAPI (см. ИСО/МЭК 19784-1). Помимо этого, настоящий стандарт содержит руководящие указания по определению комплекта тестов для соответствия спецификации BioAPI, написанию обобщенных тестовых утверждений и определению процедур проверки на соответствие.

1.2 Настоящий стандарт распространяется только на испытания на соответствие биометрической продукции спецификации BioAPI. Стандарт не устанавливает требований к методам испытаний других характеристик биометрической продукции (таких как приемлемость и удобство для конечного пользователя, устойчивость, безопасность и так далее). Испытания, проводимые с использованием методов, предназначенных для конкретного биометрического продукта, также не рассматриваются в настоящем стандарте.

1.3 Настоящий стандарт следует использовать для разработки описаний методов испытаний, наборов тестов и тестирующих программ для испытаний продуктов, претендующих на соответствие спецификации BioAPI. Настоящий стандарт в основном предназначен для использования в испытательных лабораториях, однако он может быть использован разработчиками и пользователями обобщенных тестовых утверждений и реализаций тестовых методов.

2 Соответствие

2.1 Набор тестов для испытания на соответствие спецификации BioAPI, соответствующий требованиям настоящего стандарта, должен поддерживать одну или более моделей испытания на соответствие (см. 6.2) и обладать способностью выполнять корректные тестовые утверждения, относящиеся к поддерживаемой(ым) модели(ям), и написанные на языке обобщенных тестовых утверждений, определенном в разделах 7 — 10.

Примечание — Настоящий стандарт не ограничивает форму или структуру набора тестов для испытаний на соответствие BioAPI в отношении количества компонентов программного обеспечения, задач каждого компонента, содержания и формы обмена информацией между компонентами.

2.2 С помощью набора тестов для проведения испытаний на соответствие BioAPI должна быть обеспечена возможность проведения синтаксической проверки любого пакета данных (см. 7.1.6), содержащего утверждения или процессы (или и утверждения, и процессы одновременно) для любой модели испытания на соответствие, включая модели, которые не поддерживаются приложением (в случае, если такие модели существуют).

2.3 Для каждой модели испытания на соответствие комплект тестов на соответствие BioAPI должен обладать способностью выполнять процессы (характерные для данного вида аппаратного обеспечения),

Издание официальное

необходимые для взаимодействия с тестируемой реализацией (TP), путем вызова стандартных функций биометрического программного интерфейса, содержащихся в TP, и приема обратного вызова от этих функций.

Примечания

1 При испытании приложения на соответствие BioAPI не требуется способность набора тестов запускать или останавливать выполнение тестируемой реализации, однако необходим механизм, позволяющий определить начало и окончание выполнения тестируемой реализации.

2 Также не требуется способность набора тестов испытывать абсолютно все реализации, претендующие на соответствие спецификации BioAPI. К этому утверждению, в частности (но не ограничиваясь указанным), относится случай, когда тестируемая реализация была создана для вычислительной платформы, отличающейся от платформы, для которой был создан комплект тестов, а также случай, когда тестируемая реализация зависит от аппаратного обеспечения, недоступного в вычислительной системе, используемой для проведения испытания.

2.4 Комплект тестов на соответствие BioAPI должен формировать протокол испытания и отчет о результатах испытания для каждой тестируемой реализации.

2.5 Если комплект тестов не способен обеспечить проведение испытаний представленной реализации, то это должно быть указано в отчете. Неспособность комплекта тестов обеспечить проведение испытаний на соответствие не означает несоответствия представленной реализации спецификации BioAPI.

2.6 Комплект тестов должен предоставить пользователю возможность ввода всех входных данных, необходимых для проведения испытания.

Примечание — К входным данным относятся утверждение, подлежащее обработке (имя пакета и имя утверждения), список всех входных параметров утверждения и остальная информация, которая должна быть включена в отчет о результатах испытания (см. раздел 12).

3 Нормативные ссылки

При практическом использовании данного стандарта настоятельно рекомендуется применять приведенный ниже нормативный документ. В случае ссылок на документ, у которого указана дата утверждения, необходимо использовать только указанную редакцию. В случае, когда дата утверждения не указана, следует использовать последнюю редакцию ссылочных стандартов, включая любые поправки и изменения к ним:

ИСО/МЭК 19784-1 BioAPI — Биометрический программный интерфейс — Часть 1: Спецификация биометрического программного интерфейса.

4 Термины и определения

В настоящем стандарте используются термины и определения, приведенные в ИСО/МЭК 19784-1, а также следующие термины с соответствующими определениями.

4.1 **абстрактная испытательная машина**: Абстрактная вычислительная машина, обеспечивающая возможность проведения испытаний на соответствие стандартного компонента BioAPI.

4.2 **исходный стандарт**: Стандарт, устанавливающий метод испытаний.

4.3 **комплект тестов на соответствие BioAPI**: Программное обеспечение для проведения испытаний на соответствие ИСО/МЭК 19784-1.

4.4 **утверждение соответствия BioAPI**: Утверждение, описывающее соответствие тестируемой реализации всем необходимым требованиям BioAPI.

4.5 **сертификация**: Признание завершения испытаний на соответствие и соблюдения всех критериев, установленных организацией по сертификации.

4.6 **соответствие**: Выполнение продукцией, процессом или услугой всех предъявленных требований [ИСО/МЭК 13210:1999].

4.7 **требование к соответствию**: Требование, установленное в исходном стандарте, устанавливающее необходимое условие в конечной, краткой и точно выраженной форме.

Примечания

1 Одно или совокупность требований к соответствию образуют утверждение.

2 Приведенное выше определение является адаптированной выдержкой из определения, приведенного в ИСО/МЭК 13210:1999.

4.8 **оценка соответствия**: Любое действие, прямо или косвенно связанное с определением соблюдения соответствующих требований.

4.9 **тестируемая реализация**: Программное и аппаратное обеспечение, установленное в одной или нескольких системах, удовлетворяющее требованиям исходного стандарта.

4.10 **стандартный компонент BioAPI**: Приложение BioAPI, инфраструктура BioAPI или поставщик биометрической услуги, установленные в ИСО/МЭК 19784-1.

Примечания

1 См. 6.1.3.8.

2 Несмотря на то что, в соответствии с ИСО/МЭК 19784-1, поставщик биометрической функции (ПБФ) также является компонентом BioAPI, он не включен в определение 4.10, поскольку ИСО/МЭК 19784-1 не определяет ни один интерфейс ПБФ.

4.11 **стандартный интерфейс BioAPI**: Любой интерфейс, установленный в ИСО/МЭК 19784-1, с помощью которого один стандартный компонент BioAPI обращается к другому стандартному компоненту BioAPI.

Примечание — См. 6.1.3.8.

4.12 **тестовое утверждение**: Подробное описание испытания на соответствие требованию в тестируемой реализации (ТР) в форме программного обеспечения или процедурных методов, которые формируют результаты испытания (также называемые исходом испытания или заключением испытания), используемые для оценки соответствия требованию.

4.13 **тестовый пример**: Описание действий, необходимых для достижения определенной цели испытания или совокупности целей испытания.

Примечание — Приведенное выше определение является адаптированным определением термина «абстрактный тестовый пример», приведенного в ИСО/МЭК 9646-1:1994.

4.14 **реализация метода испытания**: Программное обеспечение, процедуры или другие средства, использующиеся для оценки соответствия.

4.15 **спецификация метода испытания**: Документ, устанавливающий требования к функциональным возможностям и поведению, указанным в исходном стандарте в виде утверждений, и обеспечивающий полный набор кодов результатов испытания на соответствие [ИСО/МЭК 13210:1999].

4.16 **цель испытания**: Формальное описание четко определенной цели испытания, сосредоточенной на одном требовании к соответствию, которое определено в соответствующем описании биометрического продукта.

4.17 **отчет об испытании**: Документ, содержащий результаты испытания и иную информацию, связанную с реализацией методов испытания над тестируемой реализацией.

4.18 **код результата испытания (заключение испытания)**: Значение, описывающее результат испытания.

Примечание — Приведенное выше определение является адаптированной выдержкой из определения, приведенного в ИСО/МЭК 13210:1999.

4.19 **сертификация** — Процесс испытания программного обеспечения для оценки его соответствия определенной спецификации.

5 Сокращения

В настоящем стандарте использованы следующие сокращения:

ПИП (API) — программный интерфейс приложений (Application Programming Interface);

ПСБПИ (BCS) — подтверждение соответствия биометрическому программному интерфейсу (BioAPI Conformity Statement);

ЗБИ (BIR) — запись биометрической информации (Biometric Information Record);

ПБУ (BSP) — поставщик биометрической услуги (Biometric Service Provider);

ЕСФОБД (CBEFF) — единая структура форматов обмена биометрическими данными (Common Biometric Exchange Formats Framework);

КТСБПИ (CTS) — комплект тестов на соответствие биометрическому программному интерфейсу (BioAPI Conformance Test Suite);

ТР (IUT) — тестируемая реализация (Implementation Under Test);
ИПУ (SPI) — интерфейс поставщика услуги (Service Provider Interface);
УУИД (UUID) — универсальный уникальный идентификатор (Universally Unique Identifier);
ПБУ (BSP) — поставщик биометрической услуги (Biometric Service Provider);
ПБФ (BFP) — поставщик биометрической функции (Biometric Function Provider).

6 Методика испытаний на соответствие

6.1 Общие положения

6.1.1 Тестируемая реализация

6.1.1.1 Тестируемая реализация — это объект, который подвергается испытанию на соответствие. Для спецификаций BioAPI тестируемой реализацией является программное обеспечение, которое считается реализующим данную спецификацию. Программное обеспечение и поддерживающее его аппаратное обеспечение представляют собой тестируемую реализацию и должны быть указаны как в отчете об испытании, так и в сертификате соответствия.

6.1.1.2 Биометрическая продукция, претендующая на соответствие спецификации BioAPI, должна соответствовать всем применимым для данной продукции требованиям, изложенным в разделе «Соответствие» спецификации BioAPI. Как правило, в данном разделе установлены следующие виды требований:

a) обязательные требования, которые необходимо соблюдать во всех случаях;

b) условные требования, которые необходимо соблюдать, если они изложены в представленной спецификации;

c) дополнительные требования, которые могут быть выбраны для соответствия реализации и в этом случае должны быть соблюдены.

6.1.1.3 Для оценки соответствия биометрической продукции необходимо утверждение, содержащее список характеристик, реализованных в соответствии со спецификацией BioAPI, для того чтобы тестируемая реализация могла быть испытана на соответствие только определенным требованиям. Такое утверждение называется ПСБПИ и должно быть подготовлено поставщиком ТР до начала испытаний на соответствие. В минимальном варианте ПСБПИ должно содержать перечень всех обязательных, условных и дополнительных требований к соответствию спецификации BioAPI, включенных в ТР.

6.1.2 Метод испытания

6.1.2.1 Для того чтобы испытание имело смысл, все реализации BioAPI должны быть испытаны с помощью одного и того же метода. Испытание на соответствие отражает смысл технических требований спецификации BioAPI и оценивает степень полноты, с которой биометрическая продукция реализует спецификацию.

6.1.2.2 В настоящем стандарте испытания на соответствие представляют собой испытание функциональных возможностей реализации BioAPI по принципу «черного ящика».

6.1.2.3 Учитывая сложность спецификации BioAPI и возможные способы реализации продукции, удовлетворяющей спецификации BioAPI, наиболее подходящим является метод фальсификационного испытания. Этот метод состоит из следующих этапов:

a) анализ спецификации BioAPI и разработка документально подтвержденных тестовых примеров в форме обобщенных тестовых утверждений. В дальнейшем тестовые примеры могут быть сгруппированы для формирования тестовых сценариев. Тестовые утверждения описаны в других частях комплекса стандартов ИСО/МЭК 24709;

b) реализация тестовых утверждений в виде выполняемых тестовых сценариев, которые вместе с используемыми файлами данных формируют КТСБПИ;

c) проверка реакции тестируемой реализации на различные комбинации корректных и некорректных входных данных и сопоставление выходных данных, полученных в результате, с набором соответствующих ожидаемых результатов.

d) оценка результатов испытания с помощью критерия успешного (или неуспешного) прохождения теста.

6.1.2.4 Фальсификационное испытание позволяет определить несоответствие, то есть наличие ошибок означает несоответствие, а их отсутствие не подразумевает соответствие. Этот метод испытания предназначен для предоставления обоснованной степени уверенности и фактической гарантии того, что тестируемая реализация соответствует спецификации. Применение данного метода не гарантирует полного соответствия реализации требованиям спецификации; гарантия полного соответствия требует проведения полного и всестороннего испытания, которое невозможно по техническим и экономическим причинам.

6.1.2.5 Реализация метода испытания должна содержать документальное подтверждение соответствия требованиям настоящего стандарта и иным спецификациям метода тестирования, на соответствие которым она претендует. Каждая реализация метода испытания должна включать в себя:

a) КТСБПИ, содержащий документацию комплекта тестов, в которой должны быть описаны категории тестов, цели испытания для каждого отдельного теста, инструкция по выполнению комплекта тестов и ожидаемые результаты испытания для каждого отдельного теста. КТСБПИ должен обеспечивать возможность выполнения наборов тестовых сценариев, сбора возвращаемых значений, оценки этих значений и записи результатов этой оценки в удобочитаемом виде;

b) документально подтвержденные тестовые примеры, в достаточной степени гарантирующие соответствие спецификации, которые должны быть представлены в форме тестовых утверждений с использованием языка обобщенных тестовых утверждений, которые входят в состав КТСБПИ;

c) процедуру испытания на соответствие, которая должна определять все действия, необходимые для подготовки к испытанию на соответствие, проведения испытания и формирования отчета о результатах испытания. Процедура проведения испытания на соответствие должна быть подробно описана, чтобы испытание данной реализации можно было провести повторно.

6.1.2.6 Реализация метода испытания должна использовать необходимые определения, типы, синтаксис и конструкции тестовых утверждений, определенные в комплексе стандартов ИСО/МЭК 24709. Для результатов испытаний, установленных в настоящем стандарте, необходимо использовать коды результата испытания, приведенные в стандарте.

6.1.2.7 Процесс испытания на соответствие представляет собой законченный процесс выполнения всех действий испытания на соответствие, которые необходимы для оценки соответствия тестируемой реализации спецификации BioAPI. Процесс испытания на соответствие состоит из трех этапов:

a) подготовка к испытанию, включающая в себя анализ ПСБПИ, подготовку плана испытания на соответствие, выбор и настройку КТСБПИ, подготовку ТР и соответствующих условий испытания (средств, необходимых для проведения испытания);

b) проведение испытания, включающее в себя выполнение КТСБПИ и запись наблюдаемых результатов в протоколе испытания на соответствие. Результаты испытания на соответствие применимы только к ТР и среде, в которой выполнялся КТСБПИ;

c) подготовка отчета об испытании, который включает в себя запись всех событий, происходивших во время выполнения каждого тестового примера, включая все исходящие значения теста и заключение испытания, а также описание тестовой среды (операционная система, конфигурация аппаратного обеспечения и так далее).

6.1.2.8 Для достижения цели правдоподобного испытания на соответствие результаты выполнения тестовых примеров в тестируемой реализации должны быть одинаковыми. Поскольку не всегда возможно провести полный комплект тестов на соответствие биометрическому программному интерфейсу и убедиться в полном соответствии результатов полученным после проведения другого комплекта тестов, необходимо сделать все возможное, чтобы обеспечить повторяемость результатов тестов и свести к минимуму вероятность получения разных результатов тестов в разных случаях.

6.1.2.9 Необходимо просмотреть наблюдаемые результаты испытания, чтобы убедиться в том, что все процедуры выполнены верно. Также необходимо, чтобы все входы, выходы и иные события тестирования, условия тестирования и конфигурация ТР были внесены в протокол для каждого выполненного тестового примера. Этой информации должно быть достаточно, чтобы можно было создать протокол соответствия для каждого случая проведения испытания для его использования в дальнейшем.

6.1.3 Стандартные компоненты BioAPI и стандартные интерфейсы BioAPI

6.1.3.1 В настоящем стандарте установлены требования к методу проведения оценки соответствия реализаций ИСО/МЭК 19784 указанному стандарту (ИСО/МЭК 19784-1). Различают три типа реализации ИСО/МЭК 19784-1 (см. рисунок 1):

a) приложение BioAPI;

b) инфраструктура BioAPI;

c) ПБУ BioAPI.

6.1.3.2 Приложение BioAPI является компонентом программного обеспечения (или набором компонентов программного обеспечения), который использует интерфейс BioAPI, определенный в ИСО/МЭК 19784-1, производя один или несколько вызовов функции BioAPI в процессе его выполнения.

6.1.3.3 Инфраструктура BioAPI является компонентом программного обеспечения (или набором компонентов программного обеспечения), который реализует BioAPI, определенный в ИСО/МЭК 19784-1, и

его заданное поведение, производя один или несколько вызовов функции BioSPI, определенной в ИСО/МЭК 19784-1, в процессе его выполнения.

6.1.3.4 ПБУ BioAPI — это компонент программного обеспечения (или набор компонентов программного обеспечения), который реализует BioSPI, определенный в ИСО/МЭК 19784-1, и его заданное поведение.

6.1.3.5 ПБФ BioAPI является компонентом структуры BioAPI, но его интерфейс не определен в ИСО/МЭК 19784-1. Поэтому настоящий стандарт не содержит описание данного компонента.

6.1.3.6 Помимо двух основных указанных выше интерфейсов BioAPI и BioSPI, ИСО/МЭК 19784-1 утверждает два других интерфейса, описанные в 6.1.3.6.1 — 6.1.3.6.5.

6.1.3.6.1 Инфраструктуры BioAPI обеспечивают вспомогательный интерфейс, который поддерживает получение следующих данных от ПБУ:

a) уведомления о событиях, относящихся к элементам BioAPI;

b) потоковые данные (полутоновые битовые образы), которые посылаются во время действий, выполняемых ПБУ или ПБФ (от имени ПБУ);

c) информация о состоянии графического интерфейса пользователя (ГИП), которая посылается во время действий, выполняемых ПБУ или ПБФ от имени ПБУ.

6.1.3.6.2 Интерфейс, описанный в 6.1.3.6.1 и реализованный с помощью инфраструктур BioAPI, называется «интерфейсом обратного вызова инфраструктуры».

6.1.3.6.3 Приложения BioAPI могут реализовывать вспомогательный интерфейс, который поддерживает получение некоторой или всей представленной ниже информации от инфраструктуры BioAPI:

a) уведомления о событиях, связанных с элементами BioAPI, происходящих в ПБУ или ПБФ и переданных инфраструктурой приложению;

b) потоковые данные (полутоновые битовые образы), которые посылаются во время действий, выполняемых ПБУ или ПБФ (от имени ПБУ) и переданные инфраструктурой приложению;

c) информация о состоянии ГИП, которая посылается во время действий, выполняемых ПБУ или ПБФ (от имени ПБУ), переданная инфраструктурой приложению.

Передача инфраструктурой BioAPI приложению информации, полученной от поставщика(ов) биометрической услуги зависит от функций, поддерживаемых каждым приложением, и от использования в настоящий момент приложением ПБУ, который передал информацию в инфраструктуру BioAPI.

6.1.3.6.4 Интерфейс, описанный в 6.1.3.6.3, дополнительно реализуемый приложениями BioAPI, называется «интерфейсом обратного вызова приложения».

6.1.3.6.5 Кроме указанных выше, инфраструктуры BioAPI содержат функции, которые поддерживают установку и удаление ПБУ и ПБФ. В настоящем стандарте эти функции включены в интерфейс BioAPI.

6.1.3.7 ПБУ BioAPI может реализовывать вспомогательный интерфейс, который поддерживает получение информации от ПБФ, но этот интерфейс не определен в ИСО/МЭК 19784-1. Поэтому настоящий стандарт не содержит описание данного вспомогательного интерфейса.

6.1.3.8 Три типа реализаций интерфейса BioAPI (приложение BioAPI, инфраструктура BioAPI и ПБУ BioAPI), описанные выше в настоящем стандарте, называют стандартными компонентами BioAPI. Интерфейсы BioAPI, BioSPI, обратного вызова приложения и обратного вызова инфраструктуры, указанные выше, называют стандартными интерфейсами BioAPI. Настоящий стандарт предполагает, что каждый из стандартных компонентов BioAPI взаимодействует с другим компонентом только путем вызова функций стандартных интерфейсов BioAPI.

6.1.3.9 В настоящем стандарте установлен метод испытания на соответствие на основе трех стандартных компонентов BioAPI и четырех стандартных интерфейсов BioAPI.

6.1.4 Физические архитектуры

6.1.4.1 ИСО/МЭК 19784-1 не устанавливает способ загрузки каких-либо стандартных компонентов BioAPI в память, не утверждает физических связей между стандартными компонентами BioAPI и устройствами, подключенными к сети, а также не ограничивает количество экземпляров каждого стандартного компонента BioAPI в устройстве. Эти характеристики зависят от особенностей платформы и реализации.

6.1.4.2 Обычно ПБУ реализуются в виде динамически подключаемых библиотек на платформах, поддерживающих эту возможность.

6.1.4.3 В одной из типовых физических архитектур BioAPI инфраструктура BioAPI также является динамически подключаемой библиотекой, а приложения BioAPI являются исполняемыми программами, которые загружают динамически подключаемую библиотеку инфраструктуры BioAPI в память. В данной физической архитектуре может быть только одно приложение BioAPI, использующее экземпляр инфраструктуры BioAPI в данный момент времени. Если в устройстве одновременно функционируют несколько

приложений BioAPI, каждое из этих приложений располагает отдельным экземпляром инфраструктуры BioAPI и отдельным экземпляром ПБУ, который загружен более чем одним приложением.

6.1.4.4 В другой типовой физической архитектуре BioAPI инфраструктура BioAPI является независимой исполняемой программой (например, сервисом операционной системы) и любое количество приложений BioAPI может быть запущено и завершено в любой момент работы инфраструктуры BioAPI, при этом все приложения BioAPI одновременно используют одни и те же экземпляры инфраструктуры BioAPI и ПБУ BioAPI, которые загружены более чем одним приложением.

6.1.4.5 Учитывая абстрактность архитектуры BioAPI, описанной в ИСО/МЭК 19784-1, метод испытания на соответствие, установленный в настоящем стандарте, не зависит от особенностей физической реализации архитектуры BioAPI. Метод испытания на соответствие BioAPI относится к стандартным компонентам BioAPI как к обобщенным компонентам, обладающим интерфейсом и набором функций, описанными в ИСО/МЭК 19784-1, без учета особенностей их физической реализации.

6.1.4.6 Однако особые реализации метода испытания на соответствие (комплекты тестов на соответствие BioAPI) проводят на специальных платформах и созданы для определенных физических архитектур BioAPI. Поэтому каждый комплект тестов на соответствие BioAPI создается для тестирования лишь ограниченного множества возможных реализаций стандартных компонентов BioAPI. В приложении Е настоящего стандарта приведены рекомендации по повышению уровня соответствия разных реализаций методов испытания на соответствие, которые созданы для различных платформ или для различных физических архитектур в пределах одной платформы.

6.2 Модели испытаний на соответствие

6.2.1 Настоящий стандарт предусматривает отдельный метод испытаний на соответствие каждого стандартного компонента BioAPI. Поскольку существует три стандартных компонента BioAPI (приложение BioAPI, инфраструктура BioAPI и поставщик биометрической услуги BioAPI), настоящий стандарт устанавливает три модели испытаний на соответствие.

6.2.2. Три модели испытаний на соответствие являются независимыми друг от друга, несмотря на то что имеют много общих свойств.

6.2.3 Реализации метода испытания на соответствие отражают различия между тремя указанными моделями испытаний и приводят к различию процедур испытаний, предназначенных для единомоментного испытания на соответствие только одной из следующих реализаций:

a) реализация приложения BioAPI;

b) реализация инфраструктуры BioAPI;

c) реализация ПБУ BioAPI.

6.2.4 Каждую модель испытания создают на основе базовой архитектуры BioAPI путем внесения определенных изменений.

6.2.5 Базовая архитектура BioAPI содержит обычное приложение BioAPI, обычную инфраструктуру BioAPI и один или более ПБУ BioAPI. Каждая модель испытания дополняет или заменяет элементы базовой архитектуры в соответствии с приведенным ниже описанием.

6.2.5.1 В модели испытания на соответствие приложения BioAPI особый компонент тестирования (называемый инфраструктурой для тестирования приложений) должен находиться между тестируемым приложением и обычной инфраструктурой BioAPI (см. рисунок 2). Этот компонент должен с одной стороны реализовывать стандартный интерфейс BioAPI, а с другой — интерфейс обратного вызова приложения. В результате он должен представлять собой для тестируемого приложения инфраструктуру, а для инфраструктуры — приложение. Компонент тестирования должен обеспечивать возможность передачи вызовов приложения обычной инфраструктуре, а вызовов инфраструктуры — тестируемому приложению, а также следить за потоком входящих вызовов, анализировать и регистрировать эти вызовы, а также формировать дополнительные вызовы для проведения тестирования.

6.2.5.2 В модели испытания на соответствие инфраструктуры BioAPI особый компонент тестирования (называемый приложением для тестирования инфраструктуры) должен заменять обычное приложение, а другой компонент тестирования (называемый ПБУ для тестирования инфраструктуры) должен заменять обычный ПБУ (см. рисунок 3). Между этими двумя компонентами тестирования должна находиться тестируемая инфраструктура. Приложение для тестирования инфраструктуры должно реализовывать интерфейс обратного вызова приложения, а ПБУ для тестирования инфраструктуры должен реализовывать интерфейс BioAPI. Поэтому тестируемая инфраструктура не может отличить обычно используемое приложение от приложения, предназначенного для проведения испытания, а обычного поставщика биометрической услуги — от ПБУ, используемого для испытания. Кроме того, приложение для тестирования инфраструктуры и ПБУ

для тестирования инфраструктуры должны обладать специальным интерфейсом тестирования, который позволяет этим компонентам связываться для проведения испытания.

6.2.5.3 В модели испытаний на соответствие ПБУ BioAPI особый компонент тестирования (называемый приложением для тестирования ПБУ) должен заменять обычное приложение и обычную инфраструктуру (см. рисунок 4). Этот компонент тестирования должен выполнять функции как приложения BioAPI, так и инфраструктуры BioAPI, а также реализовывать интерфейс обратного вызова инфраструктуры. В результате он должен представлять собой для тестируемого ПБУ обычную инфраструктуру. Компонент тестирования должен обеспечивать возможность вызова интерфейса BioSPI тестируемого ПБУ.

6.2.6 Если испытанию на соответствие подвергается реализация BioAPI, включающая в себя два и более стандартных компонентов BioAPI, процедуры испытания каждого компонента следует проводить отдельно с формированием отдельных отчетов о результатах испытания для каждого тестируемого компонента (см. раздел 12).

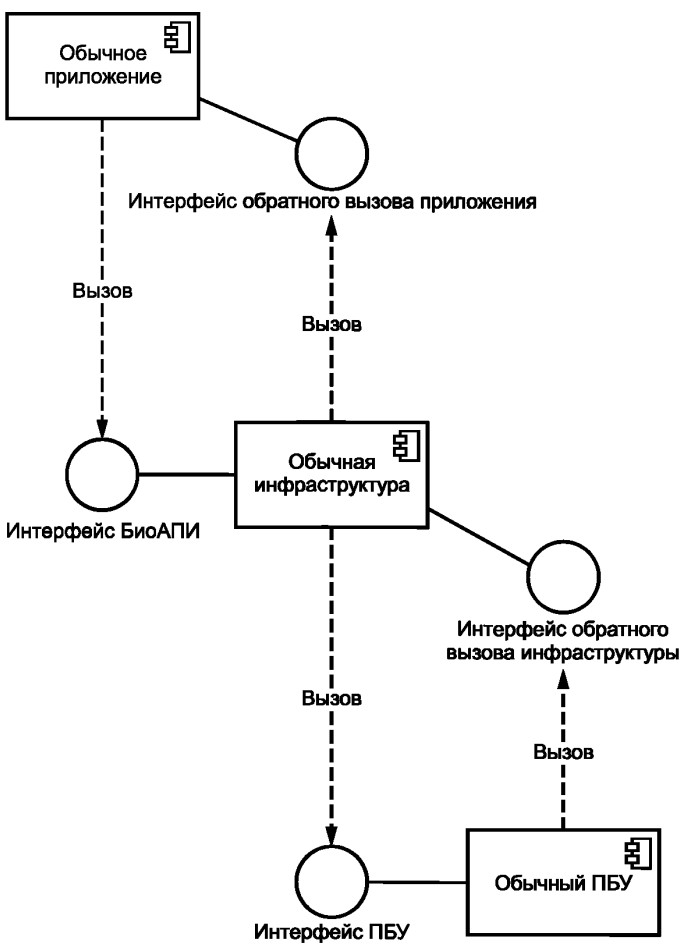


Рисунок 1 — Базовая архитектура компонентов BioAPI

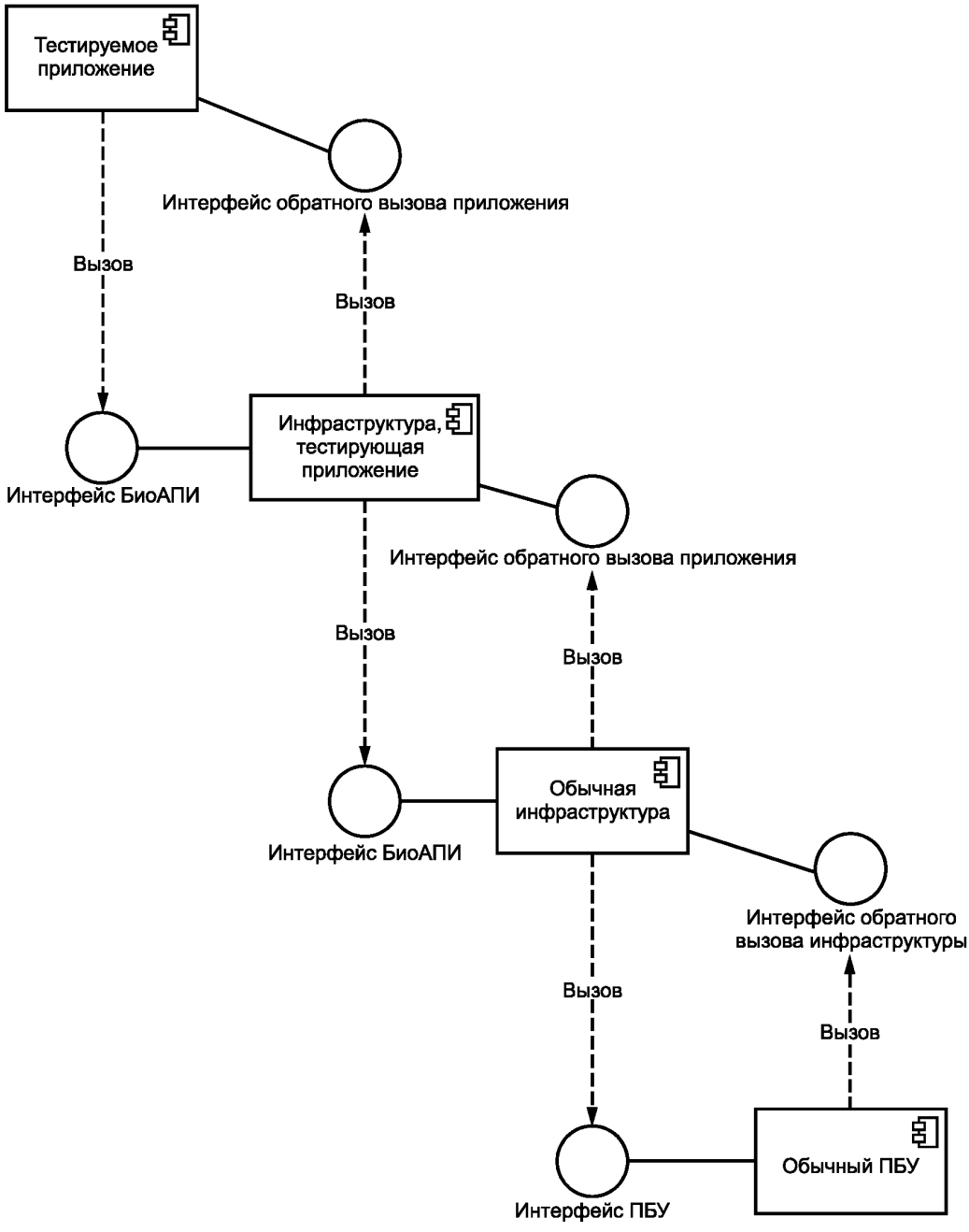


Рисунок 2 — Модель испытания на соответствие для приложений BioAPI

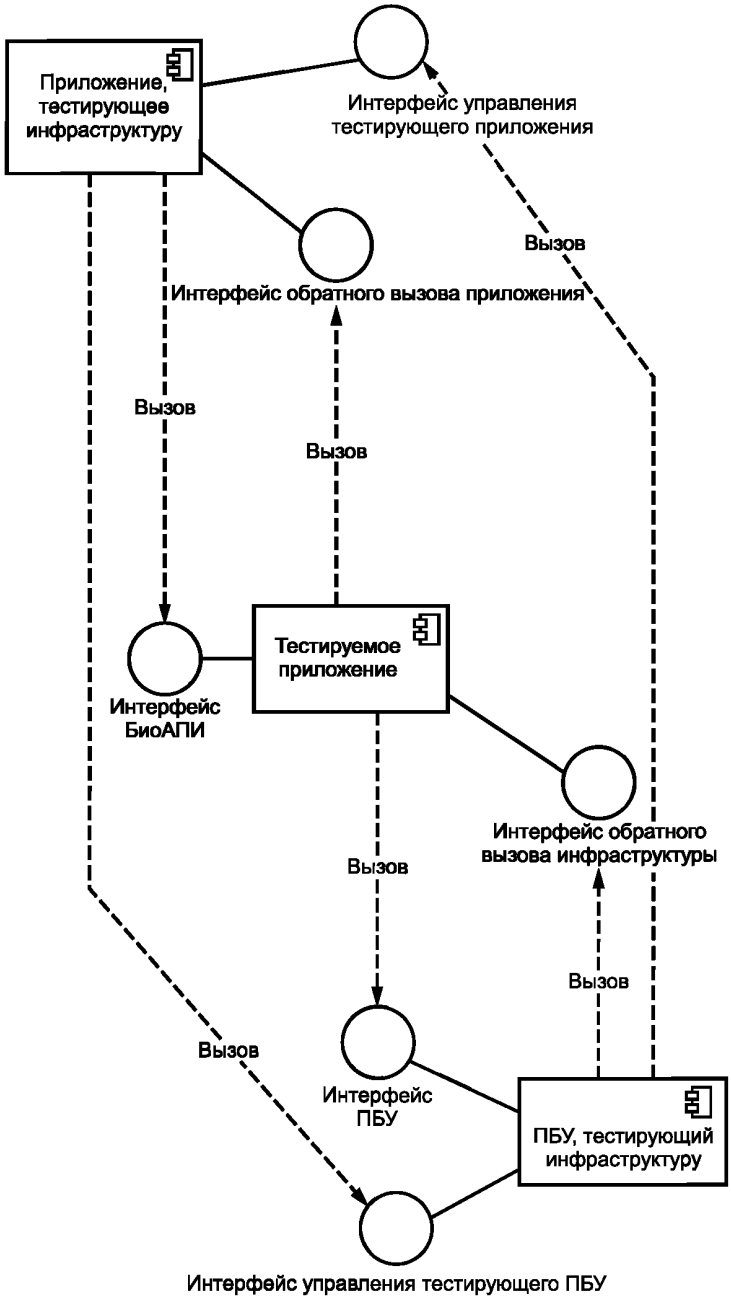


Рисунок 3 — Модель испытания на соответствие для инфраструктур BioAPI

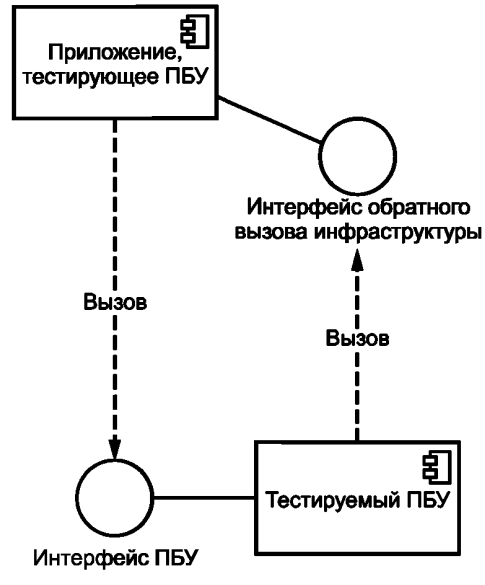


Рисунок 4 — Модель испытания на соответствие для ПБУ BioAPI

6.3 Абстрактная испытательная машина

6.3.1 Семантика языка формальных утверждений определена с использованием понятия «абстрактная испытательная машина».

6.3.2 Абстрактная испытательная машина является абстрактной вычислительной машиной, обеспечивающей возможность проведения испытания на соответствие стандартного компонента BioAPI (приложения, инфраструктуры или ПБУ). Абстрактная испытательная машина должна обеспечивать проведение испытания путем обработки формальных утверждений, написанных на языке формальных утверждений, определенном в разделе 7.

6.3.3 Абстрактная испытательная машина должна функционировать в соответствии с любой из трех моделей испытаний на соответствие, определенных в 6.2, в зависимости от типа испытуемого стандартного компонента BioAPI. При использовании каждой модели испытания абстрактная испытательная машина должна ассоциироваться со специальным компонентом тестирования в модели следующим образом:

a) при проведении испытания на соответствие приложений BioAPI (см. 6.2.5.1) абстрактная испытательная машина должна ассоциироваться со структурой для испытания приложения;

b) при проведении испытания на соответствие инфраструктур BioAPI (см. 6.2.5.2) абстрактная испытательная машина должна одновременно ассоциироваться с приложением для испытания инфраструктуры и ПБУ для испытания инфраструктуры;

c) при проведении испытаний на соответствие ПБУ BioAPI (см. 6.2.5.3) абстрактная испытательная машина должна ассоциироваться с приложением для испытания ПБУ.

6.3.4 В процессе обработки тестового утверждения, особый(е) компонент(ы) тестирования, связанный(ые) с абстрактной испытательной машиной, должен(ны) обеспечивать возможность формирования вызовов стандартных интерфейсов BioAPI тестируемых компонентов и других компонентов, присутствующих в модели испытаний. Компонент(ы) тестирования также должен(ны) обеспечивать возможность обработки входящих вызовов, сформированных другими компонентами к стандартным интерфейсам BioAPI, которые, в свою очередь, реализуются самими компонентами тестирования.

6.3.5 Тип формируемых исходящих вызовов и входящих вызовов, обрабатываемых компонентом(ами) тестирования, должен зависеть от типа тестируемого стандартного компонента BioAPI и, таким образом, от используемой модели испытаний на соответствие, в то время как реальное формирование исходящих вызовов и способ обработки входящих вызовов должны зависеть от содержания обрабатываемого утверждения.

6.3.6 В модели испытаний на соответствие, содержащей два особых компонента тестирования (модель испытания инфраструктуры BioAPI), компоненты тестирования, ассоциируемые с абстрактной испытательной машиной, должны взаимодействовать друг с другом при помощи специальных интерфейсов тестирования таким образом, чтобы абстрактная испытательная машина могла контролировать состояние и функционирование обоих компонентов одновременно. Связь между двумя компонентами тестирования не стандартизирована, а специальные интерфейсы тестирования в языке непосредственно не представлены.

6.3.7 Структура и функционирование абстрактной испытательной машины не должны зависеть от платформы, физической архитектуры BioAPI или внутренней архитектуры компонентов BioAPI. С точки зрения абстрактной испытательной машины каждый компонент должен представлять собой объект исследования с неизвестными свойствами («черный ящик»), состояние и функционирование которого зависят от:

a) вызовов, сформированных к точкам входа ее стандартных интерфейсов BioAPI;

b) течения времени или взаимодействия с другими действующими субъектами (невидимыми для абстрактной испытательной машины), поскольку они могут вызвать четко определенные процессы внутри компонента в неопределенные моменты времени.

6.3.8 Процессы, описанные в 6.3.7, перечисление b), могут опосредованно отслеживаться абстрактной испытательной машиной, поскольку эти процессы могут влиять на:

a) один или несколько последовательных вызовов, которые компонент формирует в адрес других компонентов;

b) ответы компонента на один или несколько последовательных вызовов, которые сформированы другими компонентами в его адрес.

6.3.9 Комплекты тестов на соответствие BioAPI должны содержать конкретную реализацию абстрактной испытательной машины. Нет необходимости отражать структуру абстрактной испытательной машины в подобных реализациях, однако состояние и функционирование машины, определенное индивидуальными признаками, должно быть отражено в реализации так, чтобы была сохранена семантика языка утверждения.

6.3.10 Конкретные действия, такие как загрузка и запуск исполняемой программы, обнаружение и загрузка динамически подсоединяемой библиотеки и так далее, находятся ниже уровня абстракции моделей испытаний на соответствие и абстрактной испытательной машины. Поэтому эти действия не описаны в настоящем стандарте.

7 Общие свойства языка утверждений

7.1 Общие положения

7.1.1 В разделах 7—10 установлены требования к языку, целью которого является выражение утверждений, используемых при проведении испытаний на соответствие BioAPI. Язык утверждений является неотъемлемой частью метода испытаний на соответствие BioAPI.

7.1.2 В других частях комплекса стандартов ИСО/МЭК 24709 приведены наборы утверждений, написанные на языке утверждений. Реализации методов испытания на соответствие BioAPI (комплекты тестов на соответствие биометрическому программному интерфейсу), которые претендуют на соответствие требованиям комплекса стандартов ИСО/МЭК 24709 должны использовать (или обеспечивать возможность выполнения) только утверждения, представленные в комплексе стандартов ИСО/МЭК 24709.

7.1.3 Сказанное выше не препятствует использованию конкретными реализациями методов испытаний на соответствие других наборов утверждений (которые могут быть образованы из утверждений, определенных в других частях комплекса стандартов ИСО/МЭК 24709, или нет), гарантирующих соответствие данной части комплекса стандартов ИСО/МЭК 24709. Однако такие реализации не могут претендовать на соответствие другим частям комплекса стандартов ИСО/МЭК 24709.

7.1.4 Синтаксис языка утверждений основан на языке W3C XML 1.0. Далее в настоящем разделе термины «элемент» и «атрибут» используются в значении «XML элемент» и «XML атрибут» соответственно.

7.1.5 Утверждения представлены в виде элементов **<assertion>** (см. 8.2). Каждое утверждение имеет ряд свойств, представленных в виде атрибутов и дочерних элементов элемента **<assertion>**. Среди свойств утверждения выделяют имя (атрибут **name**), модель испытаний на соответствие (атрибут **model**) и ссылку на исходный процесс (дочерний элемент **<invoke>**).

Пример

```
<assertion name="CreateTemplate1" model="BSPTesting">
    <description>
        Test the BioSPI_CreateTemplate function of a BSP.
        The UUID and version of the BSP must be provided as input to the test.
        Тестирование функции BioSPI_CreateTemplate ПБУ УУИД и версия ПБУ должны быть представлены в
        виде входных параметров теста:
    </description>
    <input name="_uuid"/>
    <input name="_version"/>
    <invoke activity="CreateTemplate" package="7346D660-1583-13D0-A3A5-00C0FFD756E3">
        <input name="BSPUuid" var="_uuid"/>
        <input name="BSPVersion" var="_version"/>
        <input name="deviceIDOrNull" value="0"/>
        <input name="inserttimeouttime" value="15000"/>
        <input name="sourcepresenttimeouttime" value="10000"/>
        <input name="capturetimeouttime" value="20000"/>
    </invoke>
    <bind activity="EventHandler" function="BioAPI_EventHandler"/>
    </assertion>
```

7.1.6 Утверждения группируют в пакеты. Каждый пакет, представленный в виде элемента **<package>**, являющегося корневым элементом документа XML, имеет имя (атрибут **name**) и другие свойства для идентификации. Пакет содержит ноль или более утверждений (элементы **<assertion>**), за которыми следует ноль или более процессов (элементы **<process>**). Допустимы пустые пакеты, а также пакеты, которые содержат только утверждения или только процессы, или и утверждения, и процессы (при этом все утверждения находятся перед первым процессом в пакете). Ни элемент **<assertion>**, ни элемент **<process>** не могут быть корневыми элементами документа XML.

Пример

```
<?xml version='1.0' encoding="utf-8"?>
<package name="73668660-1583-1AD0-A3A5-09C0FF4756E3">
    <author>
        ISO/IEC SC37
    </author>
```

```
    <description>
        Abcde abcde abcde
    </description>

    <assertion name="Capture2" model="BSPTesting">
        <description>
            Test the BioSPI_Capture function of a BSP.
        </description>
        <invoke activity="Capture"
            package="7346D660-1583-13D0-A3A5-00C0FFD756E3">
            <input name="BSPUuid" value=""/>
            <input name="BSPVersion" value="0"/>
            <input name="deviceIDOrNull" value="0"/>
            <input name="inserttimeouttime" value="15000"/>
            <input name="sourcepresenttimeouttime" value="10000"/>
            <input name="capturetimeouttime" value="20000"/>
        </invoke>
        <bind activity="EventHandler" function="BioAPI_EventHandler"/>
    </assertion>

    <assertion name="Capture5" model="BSPTesting">
        <description>
            Test the BioSPI_Capture function of a BSP with abcde abcde abcde.
        </description>
            <invoke activity="Capture"
            package="7346D660-1583-13D0-A3A5-00C0FFD756E3">
            <input name="BSPUuid" value=""/>
            <input name="BSPVersion" value="0"/>
            <input name="deviceIDOrNull" value="–1"/>
            <input name="inserttimeouttime" value="15000"/>
            <input name="sourcepresenttimeouttime" value="10000"/>
            <input name="capturetimeouttime" value="20000"/>
        </invoke>
        <bind activity="EventHandler" function="BioAPI_EventHandler"/>
    </assertion>
</package>
```

7.1.7 Утверждения могут иметь входные параметры (но не могут иметь выходных параметров). Входные параметры представляются элементами **<input>**. В процессе подготовки к испытанию входным параметрам присваиваются фактические значения. В случае утверждений, содержащих параметры, результаты теста могут зависеть от значений, присвоенных параметрам, поэтому эти значения являются важной частью выполняемого теста.

7.1.8 Рассмотрение средства присвоения значений входным параметрам утверждений выходит за рамки настоящего стандарта, но значения, используемые в тесте, должны быть указаны в стандартном отчете об испытании (см. раздел 12).

7.1.9 Каждое утверждение содержит ссылку на исходный процесс. Исходный процесс утверждения определяет процессы, которые должны быть выполнены во время обработки утверждения.

7.1.10 Процессы представлены в виде элементов **<activity>** (см. 8.9). Процесс — это последовательность (обычно, параметризованная) действий, которая может включать в себя вызов функций стандартных интерфейсов BioAPI (см. раздел 9). Данный процесс может вызывать другие процессы.

Пример

```
<?xml version='1.0' encoding="utf-8"?>
<package name="734ED660-1183-13D0-A3A5-0410FFD77AE9">
    <author>
        ISO/IEC SC37
    </author>
    <description>
        Abcde abcde abcde
    </description>
```

```
<activity name="CreateTemplate">
    <input name="BSPUuid"/>
    <input name="BSPVersion"/>
    <input name="deviceIDOrNull"/>
    <input name="inserttimeouttime"/>
    <input name="sourcepresenttimeouttime"/>
    <input name="capturetimeouttime"/>
    <invoke activity="LoadAndAttach" break_on_break="true">
      <input name="BSPUuid" var="BSPUuid"/>
      <input name="BSPVersion" var="BSPVersion"/>
      <input name="deviceIDOrNull" var="deviceIDOrNull"/>
      <input name="BSP" value="1"/>
      <input name="eventtimeouttime"
          var="inserttimeouttime"/>
    </invoke>
    <wait_until
      timeout_var="sourcepresenttimeouttime"
      setvar="eventtimeoutflag"
      var="_sourcePresent"/>
    <assert_condition
          response_if_true="undecided"
          break_if_false="true">
      <description>
          We are testing
      </description>
      <not var="eventtimeoutflag"/>
    </assert_condition>
    <invoke function="BioSPI_Capture">
      <input name="BSPHandle" value="1"/>
      <input name="Purpose" var="__BioAPI_PURPOSE_ENROLL"/>
      <input name="Timeout" var="capturetimeouttime"/>
      <output name="CapturedBIR" setvar="bir"/>
      <output name="AuditData" setvar="auditbir"/>
      <return setvar="return"/>
    </invoke>
</activity>
<activity name="LoadAndAttach">
    ...
</activity>
</package>
```

7.2 Переменные

7.2.1 Имена переменных в языке утверждений должны состоять из строк символов, соответствующих требованиям ИСО/МЭК 10646, которые подходят для создания «NCName» в пространствах имен W3C XML.

7.2.2 Переменные, имена которых начинаются с символа НИЖНЕЕ ПОДЧЕРКИВАНИЕ («_»), называют глобальными переменными. Любые другие переменные называют локальными переменными.

7.2.3 Глобальные переменные должны сохраняться в течение обработки всего утверждения. Они могут быть созданы в любом процессе (см. 8.6.2.3, 8.7.2.3, 8.12.2.5.1, 8.17.2.9), но должны быть связаны со всем утверждением и не должны быть уничтожены до завершения обработки утверждения. Глобальные переменные также могут быть созданы как входные параметры утверждений (см. 8.3.2.4).

7.2.4 Локальные переменные могут быть созданы в любом процессе (см. 8.5.2.5, 8.6.2.3, 8.7.2.3, 8.12.2.5.1, 8.17.2.9), должны быть связаны с соответствующим процессом и должны быть уничтожены по завершении этого процесса.

7.2.5 Входные и выходные параметры процессов являются локальными переменными процессов. Различия между входными параметрами (утверждений и процессов) и обычными переменными заключаются в способах их создания и присвоения начального значения. Различия между выходными параметрами процессов и обычными переменными заключаются в способах их уничтожения и расположении их конечного значения.

7.2.6 Переменные языка утверждений не имеют типа данных. Значения всех переменных представляют собой строки символов, соответствующие требованиям ИСО/МЭК 10646, неограниченной длины.

7.2.7 Значение должно быть воспринято как целое число в следующих случаях:

a) когда его оценивают с помощью числовой операции (см. 8.23—8.28);

b) когда оно передается функции стандартного интерфейса BioAPI, которая принимает в качестве входного параметра только целое число.

7.2.8 В 7.2.7, перечисление a) типовая числовая операция получает одно или более значений (строк символов, представляющих собой целые числа) в качестве входных параметров и возвращает значение. Некоторое время в течение операции значения могут быть представлены в одном из внутренних форматов, однако за пределами операции данный факт никогда не рассматривается.

7.2.9 В 7.2.7, перечисление b) преобразование строки символов в целое число является частью процесса вызова функции и не проявляется в синтаксисе языка утверждений.

7.2.10 Подобным образом переменной может быть присвоено значение (строка символов), получаемое после преобразования целого числа в строку символов. Это возможно в случае присвоения переменной выходного параметра или возвращаемого значения стандартной функции интерфейса BioAPI.

7.2.11 Преобразование целого числа в строку символов является частью процесса вызова функции и не проявляется в синтаксисе языка утверждения.

7.2.12 Обработка других типов данных, которые используются в стандартных интерфейсах BioAPI, происходит аналогично обработке целых чисел. Собственными типами данных являются списки, абстрактные идентификаторы, адреса функций и адреса данных (см. раздел 9); все эти типы данных представляют собой строки символов на языке утверждений.

7.3 Встроенные переменные

7.3.1 Ряд глобальных переменных являются частью языка утверждений. Их имена начинаются с двух следующих друг за другом символов НИЖНЕЕ ПОДЧЕРКИВАНИЕ («__»). Эти переменные определены в разделе 10.

7.3.2 Абстрактная испытательная машина должна создать все встроенные переменные и присвоить им начальные значения до начала основного процесса утверждения и не должна разрушать встроенные переменные до его завершения. Большинство встроенных переменных имеют неизменные значения, определенные в 10.1. Значения других встроенных переменных могут изменяться в соответствии с 10.2.

Примечание — Имена глобальных переменных, которые начинаются с двух следующих друг за другом символов НИЖНЕЕ ПОДЧЕРКИВАНИЕ («__»), не могут быть значением атрибута **setvar** элементов **<output>** (<выходной параметр>), **<return>** (<возвращаемое значение>) и **<wait_until>** (<ожидать_пока>) (см. 8.6.2.3, 8.7.2.3 и 8.17.2.9.1 соответственно) или значением атрибута **name** элементов **<input>** (<входной параметр>), **<set>** (<присвоить значение>), **<add>** (<добавить>) и **<subtract>** (<вычесть>) (см. 8.3.2.3, 8.12.2.2, 8.13.2.2 и 8.14.2.2 соответственно). Поэтому явное изменение значения встроенной переменной невозможно.

7.3.3 При выполнении любого процесса встроенные переменные должны обновляться только в течение тех интервалов времени, в которых процесс может быть прерван (см. 8.9.2.21). По окончании каждого такого интервала времени значения всех взаимосвязанных переменных должны быть согласованы.

Примечание — Для обеспечения надежности следует обращаться к двум или более взаимосвязанным переменным в пределах условий **<only_if>** (<только_если>), **<wait_until>** (<ожидать_пока>) и аналогичных. При необходимости обращения к нескольким встроенным переменным в последовательности элементов (например, присвоить значения двух или более встроенных переменных обычным переменным с помощью элемента **<set>** (<присвоить значение>)), элементы должны быть включены в процесс со значением атрибута **atomic="true"**.

7.4 Представление целых чисел

7.4.1 Неотрицательные целые числа должны быть представлены в виде строк, состоящих из одного или более символов, указанных в ИСО/МЭК 10646 в интервале от 0 до 9 (от «0» до «9»).

7.4.2 Отрицательные целые числа должны быть представлены в виде соответствующих положительных целых чисел, которым предшествует символ «ДЕФИС-МИНУС» («–»).

7.4.3 Стандартное представление положительного целого числа представляет собой число, которое не имеет «0» в старших разрядах. Стандартное представление целого числа 0 представляет собой число, состоящее из одного символа 0 («0»). Стандартное представление отрицательного целого числа состоит из символа «ДЕФИС-МИНУС» («–») и стандартного представления соответствующего положительного целого числа.

7.5 Представление логических значений

7.5.1 Логическое значение «ИСТИНА» (TRUE) должно быть представлено в виде строки символов, указанных в ИСО/МЭК 10646, **"true"** («ИСТИНА»). Логическое значение «ЛОЖЬ» (FALSE) должно быть представлено в виде строки символов, указанных в ИСО/МЭК 10646, **"false"** («ЛОЖЬ»).

7.5.2 Данное представление является стандартным.

7.6 Представление универсальных уникальных идентификаторов

7.6.1 УУИД должен быть представлен в виде строк символов, указанных в ИСО/МЭК 10646. Каждая строка должна состоять из символов, входящих в объединение следующих множеств:

a) цифры от «0» до «9», каждая из которых представляет собой шестнадцатеричную цифру от 0 до 9;

b) от прописной латинской буквы «A» до прописной латинской буквы «F», каждая из которых представляет собой шестнадцатеричную цифру от A до F;

c) от строчной латинской буквы «a» до строчной латинской буквы «f», каждая из которых представляет собой шестнадцатеричную цифру от A до F;

d) знак «ДЕФИС-МИНУС» («–»).

7.6.2 В состав УУИД должно входить ровно 32 шестнадцатеричные цифры. Также в состав УУИД должно входить четыре символа «ДЕФИС-МИНУС» («–») в следующих позициях:

a) между восьмой и девятой шестнадцатеричной цифрами;

b) между двенадцатой и тринадцатой шестнадцатеричными цифрами;

c) между шестнадцатой и семнадцатой шестнадцатеричными цифрами;

d) между двадцатой и двадцать первой шестнадцатеричной цифрами.

7.6.3 Стандартным представлением УУИД является такое его представление, в котором не используются символы, отнесенные к категории, указанной в 7.6.1, перечисление b).

7.7 Представление набора байтов

7.7.1 Набор байтов должен быть представлен в виде строк символов, указанных в ИСО/МЭК 10646. Каждая строка должна содержать четное количество (возможно, ноль) символов из объединения следующих множеств:

a) от цифры «0» до цифры «9», каждая из которых представляет собой шестнадцатеричную цифру от 0 до 9;

b) от прописной латинской буквы «A» до прописной латинской буквы «F», каждая из которых представляет собой шестнадцатеричную цифру от A до F;

c) от строчной латинской буквы «a» до строчной латинской буквы «f», каждая из которых представляет собой шестнадцатеричную цифру от A до F.

7.7.2 Стандартным представлением набора байтов является такое его представление, в котором не используются символы, отнесенные к категории, указанной в 7.7.1, перечисление b).

7.8 XML-документы

7.8.1 Утверждения и процессы должны быть приведены в документах формата W3C XML 1.0. Должен использоваться расширяемый язык разметки (The Extensible Markup Language, XML) версии «1.0». Для кодировки символов должна использоваться кодировка «utf-8» или «utf-16».

7.8.2 Корневым элементом всех документов формата XML должен быть элемент **<package>** (см. 8.1).

7.8.3 Документы XML должны соответствовать схеме документов XML, приведенной в приложении A, а также абстрактному синтаксису нотаций (Abstract Syntax Notation, ASN) ASN.1, приведенному в приложении B.

Примечание — Для проверки формальной корректности утверждений реализация может использовать любую из схем XML или ASN.1 в силу их эквивалентности.

8 Элементы языка утверждений

8.1 Элемент <package>

8.1.1 Синтаксис

8.1.1.1 Данный элемент должен иметь следующий атрибут:

- **name** (обязательный атрибут) — значением этого атрибута должно быть корректное имя пакета (см. 8.1.2.5).

8.1.1.2 В состав элемента в указанном порядке должны входить:

a) один элемент **<author>** — этот элемент должен содержать имя или описание автора пакета (строка символов);

b) один элемент **<description>** — этот элемент должен содержать описание пакета (строка символов);

c) ноль или более элементов **<assertion>** — этот элемент представляет собой утверждение и определен в 8.2;

d) ноль или более элементов **<activity>** — этот элемент представляет собой процесс и определен в 8.9.

8.1.2 Семантика

8.1.2.1 Пакет является контейнером для утверждений и процессов, имеющим имя.

8.1.2.2 К утверждениям и процессам возможно обращение из внешней по отношению к пакету среды.

8.1.2.3 Обращение из пакета к утверждениям не выполняется, однако возможно обращение к утверждениями из внешней по отношению к языку утверждений среды. В частности, обращение к утверждениям возможно во время подготовки к испытанию и в отчетах об испытании (см. раздел 12).

8.1.2.4 Обращение к процессам осуществляется, как правило, из утверждений (в элементах **<invoke>** или **<bind>**) или других процессов. В обоих случаях обращение к процессу возможно как из пакета, содержащего данный процесс, так и из внешнего по отношению к данному процессу пакета.

8.1.2.5 Имена пакетов должны состоять из строк символов, указанных в ИСО/МЭК 10646, представляющих собой универсальный уникальный идентификатор (см. 7.6). Каждое имя пакета должно быть уникальным.

8.1.2.6 Два имени пакета считаются одинаковыми только в том случае, если они содержат одинаковую последовательность цифр и букв независимо от регистра.

8.1.2.7 Каждое утверждение, входящее в пакет, должно иметь уникальное имя. Каждый процесс, входящий в пакет, должен иметь уникальное имя.

Примечание — Утверждение и процесс в пределах одного пакета могут иметь одно и то же имя.

8.1.3 Пример

```
<?xml version='1.0' encoding="utf-8"?>
<package name="734ED660-1183-13D0-A3A5-0410FFD77AE9">
    <author>
        ISO/IEC SC37
    </author>
    <description>
        Abcde abcde abcde
    </description>
    <assertion name="Capture2" model="BSPTesting">
        <description>
            Test the BioSPI_Capture function of a BSP.
        </description>
        <invoke activity="Capture"
            package="7346D660-1583-13D0-A3A5-00C0FFD756E3">
            <input name="BSPUuid" value=""/>
            <input name="BSPVersion" value="0"/>
            <input name="deviceIDOrNull" value="0"/>
            <input name="inserttimeouttime" value="15000"/>
            <input name="sourcepresenttimeouttime" value="10000"/>
            <input name="capturetimeouttime" value="20000"/>
        </invoke>
        <bind activity="EventHandler" function="BioAPI_EventHandler"/>
    </assertion>
</package>
```

8.2 Элемент <assertion> (дочерний для элемента <package>)

8.2.1 Синтаксис

8.2.1.1 Данный элемент должен иметь следующие атрибуты:

a) **name** (обязательный атрибут) — значением этого атрибута должно быть корректное имя утверждения (см. 8.2.2.7).

b) **model** (обязательный атрибут) — этот атрибут должен принимать одно из следующих значений: **"applicationTesting"**, **"frameworkTesting"**, **"BSPTesting"**; это значение должно определять модель испытаний на соответствие, используемую утверждением.

8.2.1.2 В состав элемента в указанном порядке должны входить:

a) один элемент **<description>** — этот элемент должен содержать описание утверждения (строка символов);

b) ноль или более элементов **<input>** — этот элемент является входным параметром утверждения и определен в 8.3;

c) один элемент **<invoke>** — этот элемент представляет собой вызов основного процесса утверждения и определен в 8.4;

d) ноль или более элементов **<bind>** — этот элемент представляет собой связь между функцией стандартного интерфейса BioAPI и процессом и определен в 8.8.

8.2.2 Семантика

8.2.2.1 Атрибут **model** определяет выбор модели испытаний на соответствие (см. 6.2), используемой для обработки утверждения, и, таким образом, определяет какой(ие) компонент(ы) этой модели должен(ны) ассоциироваться с абстрактной испытательной машиной.

8.2.2.2 Если есть только один компонент тестирования, то все вызовы стандартных функций других стандартных компонентов BioAPI, являющиеся результатом обработки утверждения, должны быть выполнены этим компонентом, а все входящие вызовы этого компонента должны быть обработаны в соответствии с содержимым утверждения.

8.2.2.3 Если есть два компонента тестирования, то любой из них может выполнять вызов стандартных функций интерфейса BioAPI других компонентов, а все входящие вызовы любого из компонентов тестирования должны быть обработаны в соответствии с содержимым утверждения.

8.2.2.4 Если атрибут **model** имеет значение **"applicationTesting"**, то для этого утверждения должна быть выбрана модель испытаний приложения BioAPI (см. 6.2.5.1), а абстрактная испытательная машина должна рассматриваться как инфраструктура, тестирующая приложение.

8.2.2.5 Если атрибут **model** имеет значение **"frameworkTesting"**, то для этого утверждения должна быть выбрана модель испытаний инфраструктуры BioAPI (см. 6.2.5.2), а абстрактная испытательная машина должна рассматриваться как приложение, тестирующее инфраструктуру, и, одновременно, как ПБУ, тестирующий инфраструктуру.

8.2.2.6 Если атрибут **model** имеет значение **"BSPTesting"**, то для этого утверждения должна быть выбрана модель испытаний ПБУ BioAPI (см. 6.2.5.1), а абстрактная испытательная машина должна рассматриваться как приложение, тестирующее ПБУ.

8.2.2.7 Имена утверждений должны состоять из строк символов, указанных в ИСО/МЭК 10646, которые подходят для создания имен «NCName» в пространствах имен W3C XML.

8.2.2.8 Имена утверждений в пакете должны быть уникальными.

8.2.3 Пример

```
<assertion name="CreateTemplate1" model="BSPTesting">
   <description>
      Test the BioSPI_CreateTemplate function of a BSP.
      The UUID and version of the BSP must be provided as input to the test.
   </description>
   <input name="_uuid"/>
   <input name="_version"/>
   <invoke activity="CreateTemplate"
      package="7346D660-1583-13D0-A3A5-00C0FFD756E3">
      <input name="BSPUuid" var="_uuid"/>
      <input name="BSPVersion" var="_version"/>
      <input name="deviceIDOrNull" value="0"/>
      <input name="inserttimeouttime" value="15000"/>
      <input name="sourcepresenttimeouttime" value="10000"/>
      <input name="capturetimeouttime" value="20000"/>
   </invoke>
   <bind activity="EventHandler" function="BioAPI_EventHandler"/>
</assertion>
```

8.3 Элемент <input> (дочерний для элемента <assertion>)

8.3.1 Синтаксис

8.3.1.1 Данный элемент должен иметь следующий атрибут:

- **name** (обязательный атрибут) — значением этого атрибута должно быть корректное имя глобальной переменной (см. 7.2), которое является именем входного параметра утверждения.

8.3.1.2 Данный элемент не должен иметь текстового содержимого.

8.3.2 Семантика

8.3.2.1 Данный элемент является входным параметром утверждения.

Примечание — Утверждения не должны содержать выходных параметров.

8.3.2.2 Утверждения с параметрами могут быть обработаны только в том случае, если каждому из параметров присвоено значение. Способы присваивания значений входным параметрам утверждений в настоящем стандарте не рассматриваются.

8.3.2.3 Значением атрибута **name** должно быть корректное имя глобальной переменной (см. 7.2) и не должно начинаться с двух следующих друг за другом символов «НИЖНЕЕ ПОДЧЕРКИВАНИЕ» («__»).

8.3.2.4 Необходимо создать входные параметры утверждений и присвоить им значения, представленные как входные данные теста.

8.3.2.5 Присвоение значений параметрам утверждений является неотъемлемой частью подготовки к проведению испытания. Результаты испытания некоторых реализаций могут зависеть от значений, которые присвоены входным параметрам утверждений. Значения, используемые в данном тесте, фиксируются в стандартном отчете о результатах испытания (см. раздел 12).

8.3.2.6 Именами входных параметров утверждения должны быть корректные имена глобальных переменных (см. 7.2). Имена всех глобальных переменных (включая входные параметры) должны быть уникальными.

Примечание — Значения, присвоенные параметрам утверждения, не могут быть изменены в процессе обработки утверждения, потому что все элементы языка утверждений, осуществляющие присваивание или изменение значения глобальной переменной (см. 8.6.2.3, 8.7.2.3, 8.12.2.2, 8.13.2.2, 8.14.2.2 и 8.17.9.1) не позволяют использовать параметр утверждения как переменную, значение которой можно изменять.

8.3.3 Пример

```
<input name="_uuid"/>
```

8.4 Элемент <invoke> (дочерний для элемента <assertion>)

8.4.1 Синтаксис

8.4.1.1 Данный элемент должен иметь следующие атрибуты:

a) **activity** (обязательный атрибут) — значением этого атрибута должно быть имя процесса;

b) **package** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть имя пакета, который содержит процесс, определенный атрибутом **activity**.

8.4.1.2 В состав элемента в указанном порядке должны входить:

- ноль или более элементов **<input>** — этот элемент предоставляет значение для входного параметра процесса и определен в 8.5.

8.4.2 Семантика

8.4.2.1 Этот элемент обозначает основной процесс утверждения и определяет фактические входные параметры его вызова. Обработка утверждения должна приводить к выполнению процесса с предоставленными входными параметрами. Каждое утверждение должно содержать только один такой процесс.

8.4.2.2 Процесс может содержать элементы **<assert_condition>** с атрибутом **break_if_false** (см. 8.18.2.3). Если в ходе выполнения процесса произошло прерывание, обработка всего утверждения должна быть завершена.

8.4.2.3 Набор элементов **<input>** вызова должен соответствовать входным параметрам процесса согласно 8.5.2.2.

8.4.2.4 Атрибут **package** является обязательным, если процесс находится не в том пакете, в котором находится утверждение; в противном случае атрибут **package** является необязательным.

8.4.2.5 Процесс может иметь выходные параметры или не иметь их. Если у основного процесса утверждения есть выходные параметры, то значения всех выходных параметров вызываемых процессов сбрасываются после завершения этих процессов.

Примечание — Это аналогично вызову процесса с выходными параметрами другим процессом без предоставления каких-либо элементов **<output>**, соответствующих выходным параметрам вызываемого процесса. Такой способ вызова допускается и используется тогда, когда вызывающему процессу не нужна информация о значениях выходных параметров вызываемого процесса.

8.4.2.6 У элемента утверждений **<invoke>** не должно быть атрибута **break_on_break** (см. 8.15.2.6.5) и дочернего объекта **<only_if>**.

8.4.3 Пример

```
<invoke activity="CreateTemplate"
   package="7346D660-1583-13D0-A3A5-00C0FFD756E3">
   <input name="BSPUuid" var="_uuid"/>
   <input name="BSPVersion" var="_version"/>
   <input name="deviceIDOrNull" value="0"/>
   <input name="inserttimeouttime" value="15000"/>
   <input name="sourcepresenttimeouttime" value="10000"/>
   <input name="capturetimeouttime" value="20000"/>
</invoke>
```

8.5 Элемент <input> (дочерний для элемента <invoke>)

8.5.1 Синтаксис

8.5.1.1 Данный элемент должен иметь следующие атрибуты:

a) **name** (обязательный атрибут) — значением этого атрибута должно быть имя входного параметра вызываемого процесса или вызываемой функции стандартного интерфейса BioAPI;

b) **value** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть значение, которое присваивается входному параметру с атрибутом **name**;

c) **var** (необязательный параметр) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2), значение которой присваивается входному параметру с атрибутом **name**.

8.5.1.2 Должен присутствовать только один из двух необязательных атрибутов — **value** или **var**.

8.5.1.3 Данный элемент не должен иметь текстового содержимого.

8.5.2 Семантика

8.5.2.1 Этот элемент представляет собой присвоение значения входному параметру процесса или функции стандартного интерфейса BioAPI (см. раздел 9) при вызове процесса или функции.

8.5.2.2 Набор элементов **<input>** вызова должен соответствовать входным параметрам процесса или вызываемой функции стандартного интерфейса BioAPI в соответствии с 8.5.2.2.1 и 8.5.2.2.2.

8.5.2.2.1 При вызове процесса для каждого элемента **<input>** вызывающего процесса должно быть не более одного элемента **<input>** вызова, значение атрибута **name** которого равно значению атрибута **name** элемента **<input>** вызывающего процесса, и наоборот. Для каждого элемента **<input>** вызова процесса должен быть элемент **<input>** вызываемого процесса, значение атрибута **name** которого равно значению атрибута **name** вызова. Порядок элементов **<input>** в процессе и в вызове процесса могут не совпадать.

8.5.2.2.2 При вызове функции для каждого входного параметра функции должно быть не более одного элемента **<input>** вызова, значение атрибута **name** которого совпадает с именем входного параметра функции, и наоборот. Для каждого элемента **<input>** в вызове должен быть входной параметр функции, имя которого совпадает со значением атрибута **name** элемента **<input>** в вызове. Элементы **<input>** в вызове могут быть перечислены в любом порядке.

Примечание — В разделе 9 приведены имена всех функций стандартного интерфейса BioAPI и имена их входных и выходных параметров; вызов функций должен осуществляться в соответствии с определенными именами параметров.

8.5.2.3 Если атрибут **var** присутствует, то его значением должно быть имя предварительно объявленной переменной (см. 7.2). Значение переменной должно быть присвоено входному параметру. Переменная может быть глобальной или локальной.

8.5.2.4 Если атрибут **value** присутствует, то его значение должно быть присвоено входному параметру.

8.5.2.5 При вызове процесса входной параметр должен создаваться как локальная переменная вызываемого процесса, и ему должно быть присвоено определенное значение.

8.5.2.6 При вызове функции определенное значение должно быть преобразовано в собственную форму, подходящую для соответствующего параметра данной функции стандартного интерфейса BioAPI в соответствии с разделом 9.

8.5.3 Пример

```
<input name="BSPUuid" var="_uuid"/>
```

8.6 Элемент <output> (дочерний для элемента <invoke>)

8.6.1 Синтаксис

8.6.1.1 Данный элемент должен иметь следующие атрибуты:

a) **name** (обязательный атрибут) — значением этого атрибута должно быть имя выходного параметра процесса или вызываемой функции стандартного интерфейса BioAPI;

b) **setvar** (обязательный атрибут) — значением этого атрибута должно быть корректное имя переменной (см. 7.2), которой должно быть присвоено значение выходного параметра.

8.6.1.2 Данный элемент не должен иметь текстового содержимого.

8.6.2 Семантика

8.6.2.1 Этот элемент представляет собой присвоение значения выходного параметра процесса или функции стандартного интерфейса BioAPI переменной (см. раздел 9) в вызове.

8.6.2.2 Набор элементов **<output>** вызова должен соответствовать выходным параметрам вызываемого процесса или функции стандартного интерфейса BioAPI в соответствии с 8.6.2.2.1 и 8.6.2.2.2.

8.6.2.2.1 При вызове процесса для каждого элемента **<output>** вызывающего процесса должно быть не более одного элемента **<output>** вызова, значение атрибута **name** которого равно значению атрибута **name** элемента **<output>** вызывающего процесса, и наоборот. Для каждого элемента **<output>** в вызове должен быть элемент **<output>** вызываемого процесса, значение атрибута **name** которого равно значению атрибута **name** элемента **<output>** в вызове. Порядок элементов **<output>** в процессе и в вызове процесса могут не совпадать.

8.6.2.2.2 При вызове функции для каждого выходного параметра вызываемой функции должно быть не более одного элемента **<output>** вызова, значение атрибута **name** которого совпадает с именем выходного параметра вызываемой функции. Для каждого элемента **<output>** в вызове должен быть выходной параметр функции, имя которого совпадает со значением атрибута **name** элемента **<output>** в вызове. Элементы **<output>** в вызове функции могут находиться в любом порядке.

Примечание — В разделе 9 приведены имена всех функций стандартного интерфейса BioAPI и имена их входных и выходных параметров; вызов функций должен осуществляться в соответствии с определенными именами параметров.

8.6.2.3 Значением атрибута **setvar** должно быть корректное имя глобальной или локальной переменной (см. 7.2). Значение атрибута **setvar** не должно быть входным параметром обрабатываемого утверждения (см. 8.3) и не должно начинаться со следующих друг за другом символов «НИЖНЕЕ ПОДЧЕРКИВАНИЕ» («__»). Переменная может существовать до вызова процесса или функции или может быть новой переменной, которую необходимо создать после завершения вызванного процесса или функции.

8.6.2.4 Все значения атрибутов **setvar** элементов **<output>** в вызове должны быть определены.

8.6.2.5 Если переменной не существует, она должна быть создана после завершения вызываемого процесса или функции. Если переменная является локальной, то она должна ассоциироваться с текущим (вызванным) процессом, чтобы ее можно было уничтожить после завершения процесса.

8.6.2.6 При вызове процесса переменной должно быть присвоено окончательное значение выходного параметра вызываемого процесса, которое соответствует значению параметра непосредственно перед его уничтожением. Выходной параметр должен существовать в момент завершения вызываемого процесса.

Примечание — Выходной параметр (как локальная переменная) может быть создан в любое время в течение выполнения вызванного процесса.

8.6.2.7 При вызове функции значение соответствующего выходного параметра данной функции стандартного интерфейса BioAPI должно быть преобразовано из собственной формы в строку символов в соответствии с разделом 9, а переменной в качестве значения должна быть присвоена полученная строка символов.

8.7 Элемент <return> (дочерний для элемента <invoke>)

8.7.1 Синтаксис

8.7.1.1 Данный элемент должен иметь следующие атрибуты:

- **setvar** (обязательный атрибут) — значением этого атрибута должно быть корректное имя переменной (см. 7.2), которой присваивается возвращаемое значение.

8.7.1.2 Данный элемент не должен иметь текстового содержимого.

8.7.2 Семантика

8.7.2.1 Этот элемент представляет собой присвоение возвращаемого значения функции стандартного интерфейса BioAPI переменной (см. раздел 9) при вызове этой функции.

8.7.2.2 В вызове функции стандартного интерфейса BioAPI должно быть не более одного элемента **<return>**.

8.7.2.3 Значением атрибута **setvar** должно быть корректное имя глобальной или локальной переменной (см. 7.2). Значение атрибута **setvar** не должно быть входным параметром обрабатываемого утверждения (см. 8.3) и не должно начинаться с двух следующих друг за другом символов «НИЖНЕЕ ПОДЧЕРКИВАНИЕ» («__»). Переменная может существовать до вызова функции или может быть новой переменной, которую необходимо создать после завершения вызываемого процесса или функции.

8.7.2.4 Если переменной не существует, она должна быть создана после завершения вызванной функции. Если переменная является локальной, то она должна ассоциироваться с текущим (вызываемым) процессом, чтобы ее можно было уничтожить после завершения процесса.

8.7.2.5 Возвращаемое значение функции стандартного интерфейса BioAPI должно быть преобразовано из собственной формы в строку символов в соответствии с разделом 9, а переменной в качестве значения должна быть присвоена полученная строка символов.

8.8 Элемент <bind> (дочерний для элемента <assertion>)

8.8.1 Синтаксис

8.8.1.1 Данный элемент должен иметь следующие атрибуты:

a) **function** (обязательный атрибут) — значением этого атрибута должно быть имя функции стандартного интерфейса BioAPI (см. раздел 9);

b) **activity** (обязательный атрибут) — значением этого атрибута должно быть имя процесса;

c) **package** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть имя пакета, в котором содержится процесс, определенный атрибутом **activity**.

8.8.1.2 Данный элемент не должен иметь текстового содержимого.

8.8.2 Семантика

8.8.2.1 Этот элемент представляет собой связь между функцией стандартного интерфейса BioAPI (на которую воздействует компонент тестирования) и процессом; с помощью указанной связи происходит автоматический вызов процесса в ответ на входящий вызов функции стандартного интерфейса BioAPI.

8.8.2.2 Указанная связь должна присутствовать в течение всей обработки утверждения.

Примечание — Все связи должны быть установлены до начала выполнения основного процесса утверждения и не должны быть впоследствии разорваны или изменены.

8.8.2.3 Функция стандартного интерфейса BioAPI, определенная атрибутом **function**, должна быть одной из функций, на которые воздействует(ют) компонент(ы) тестирования.

8.8.2.4 Нельзя обращаться к одной и той же функции стандартного интерфейса BioAPI в двух разных элементах **<bind>** одного и того же утверждения.

8.8.2.5 Связанный процесс должен иметь входные и выходные параметры, полученные из входных и выходных параметров схемы вызова функции (определенной в разделе 9) стандартного интерфейса BioAPI, с которой связан данный процесс:

a) в модели испытания на соответствие приложений BioAPI все входные и выходные параметры вызова функции должны быть входными параметрами связанного процесса; кроме того, должен присутствовать входной параметр с именем **return**;

b) в модели испытания на соответствие инфраструктуры BioAPI и ПБУ BioAPI все входные параметры вызова функции должны быть выходными параметрами связанного процесса, а все выходные параметры вызова функции должны быть входными параметрами связанного процесса; кроме того, должен присутствовать выходной параметр с именем **return**.

8.8.2.6 Связанный процесс следует вызывать каждый раз, когда компонент тестирования получает входящий вызов функции стандартного интерфейса BioAPI, с которой связан процесс согласно 8.9.2.7 и 8.9.2.8.

8.9 Элемент <activity> (дочерний для элемента <package>)

8.9.1 Синтаксис

8.9.1.1 Данный элемент должен иметь следующие атрибуты:

a) **name** (обязательный атрибут) — значением этого атрибута должно быть корректное имя процесса (см. 8.9.2.22);

b) **atomic** (необязательный атрибут) — этот атрибут должен определять непрерывность выполнения процесса (по умолчанию атрибут **atomic** имеет значение «ложно»).

8.9.1.2 В состав элемента в указанном порядке должны входить:

a) ноль или более элементов **<input>** — этот элемент представляет собой входной параметр процесса и определен в 8.10;

b) ноль или более элементов **<output>** — этот элемент представляет собой выходной параметр процесса и определен в 8.11;

c) ноль или более каких-либо из представленных ниже элементов в любом порядке:

1) **<set>** — этот элемент представляет собой присвоение значения новой переменной или существующей переменной и определен в 8.12;

2) **<add>** — этот элемент представляет собой прибавление целого числа к значению существующей переменной (являющейся целым числом) и определен в 8.13;

3) **<subtract>** — этот элемент представляет собой вычитание целого числа из значения существующей переменной (являющейся целым числом) и определен в 8.14;

4) **<wait_until>** — этот элемент представляет собой временное прекращение выполнения текущего процесса до тех пор, пока условие не подтверждено, и определен в 8.17;

5) **<assert_condition>** — этот элемент представляет собой выходной результат испытания на соответствие, основанного на заданном условии, и определен в 8.18;

6) **<invoke>** — этот элемент представляет собой вызов процесса или функции стандартного интерфейса BioAPI и определен в подразделе 8.15.

8.9.2 Семантика

8.9.2.1 Процессы являются исполняемыми модулями языка утверждений и состоят из упорядоченной последовательности нуля или более элементов, выполняющих следующие действия:

a) присвоение значения переменной;

b) временное прекращение выполнения текущего процесса до тех пор, пока условие не подтверждено;

c) вызов функции стандартного интерфейса BioAPI;

d) выдача результата проверки соответствия при заданном условии;

e) вызов процесса.

8.9.2.2 Абстрактная испытательная машина должна выполнять действия процесса в установленном порядке.

8.9.2.3 Каждый процесс должен иметь приоритет, который может быть низким, средним или высоким. Во время обработки утверждения может выполняться не более одного процесса. Абстрактная испытательная машина должна обеспечивать возможность прерывания процесса с целью выполнения процесса с более высоким приоритетом (см. 8.9.2.21).

8.9.2.4 Перед началом обработки утверждения абстрактная испытательная машина должна вызвать исходный процесс утверждения как процесс с высоким приоритетом.

8.9.2.5 Если один процесс вызывает другой процесс, то вызываемому процессу должен быть присвоен приоритет вызывающего процесса.

8.9.2.6 Когда один из следующих элементов

a) элемент **<wait_until>** или

b) элемент **<invoke>**, который вызывает функцию стандартного интерфейса BioAPI, в первый раз появляется в исходном процессе утверждения или в процессе, который был им (прямо или косвенно) вызван, приоритет процесса, в котором появился один из указанных элементов, должен быть изменен с высокого на низкий непосредственно перед началом выполнения этого элемента. Если процесс, в котором появился элемент, не является исходным процессом утверждения, приоритет каждого процесса (включая основной процесс), который (прямо или косвенно) активизировал этот процесс, также должен быть изменен на низкий.

8.9.2.7 Если в модели испытаний на соответствие для приложений BioAPI (см. 6.2.5.1) компонент тестирования получает входящий вызов функции стандартного интерфейса BioAPI, должны быть выполнены следующие действия.

8.9.2.7.1 Если вызванная функция принадлежит интерфейсу BioAPI, то входящий вызов должен быть передан аналогичной функции стандартного интерфейса BioAPI, объявленной нормальной структурой. Новый (переданный) вызов должен иметь те же значения собственных параметров, что и первоначальный входящий вызов.

8.9.2.7.2 Если вызванная функция принадлежит интерфейсу обратного вызова приложения, то входящий вызов должен быть передан аналогичной функции стандартного интерфейса BioAPI, объявленной тестируемым приложением. Новый (переданный) вызов должен иметь те же значения собственных параметров, что и первоначальный входящий вызов.

8.9.2.7.3 Если возврат из переданного вызова осуществляется до возврата первоначального входящего вызова и существует процесс, связанный с вызываемой функцией стандартного интерфейса BioAPI (см. 8.8), абстрактная испытательная машина должна прервать текущий процесс с низким приоритетом и вызвать процесс, связанный с вызываемой функцией стандартного интерфейса BioAPI, с присвоением данному процессу высокого приоритета (с учетом требований, приведенных в 8.9.2.9).

8.9.2.7.4 Входным параметрам вызова связанного процесса должны быть присвоены значения собственных входных параметров входящего вызова и собственных выходных параметров переданного вызова, а также возвращаемое значение переданного вызова сразу же после его завершения в соответствии с пунктами раздела 9 с заголовком «Входные параметры вызова связанного процесса», относящимися к функции стандартного интерфейса BioAPI.

8.9.2.8 В модели испытаний на соответствие инфраструктуры BioAPI (см. 6.2.5.2) и ПБУ BioAPI (см. 6.2.5.3) при получении компонентом тестирования входящего вызова должны быть выполнены действия, описанные в 8.9.2.8.1 — 8.9.2.8.4.

8.9.2.8.1 Если с вызываемой функцией связан процесс (см. 8.8), то абстрактная испытательная машина должна обеспечивать возможность прерывания выполнения текущего процесса с низким приоритетом и вызова связанного процесса (с учетом требований, приведенных в 8.9.2.9) с приоритетом, выбранным следующим образом:

a) если это функция **BioSPI_Cancel**, должен быть выбран высокий приоритет;

b) если функция принадлежит интерфейсу BioSPI, но не является функцией **BioSPI_Cancel**, должен быть выбран средний приоритет;

c) в остальных случаях должен быть выбран высокий приоритет.

8.9.2.8.2 Входным параметрам вызова связанного процесса должны быть присвоены значения собственных входных параметров входящего вызова в соответствии с пунктами раздела 9 с заголовком «Входные параметры вызова связанного процесса», относящимися к функции стандартного интерфейса BioAPI.

8.9.2.8.3 После завершения связанного процесса собственным выходным параметрам и возвращаемому значению входящего вызова должны быть присвоены значения выходных параметров процесса в соответствии с пунктами раздела 9 с заголовком «Выходные параметры вызова связанного процесса», относящимися к функции стандартного интерфейса BioAPI.

8.9.2.8.4 Если вызываемая функция не имеет связанных процессов, то абстрактная тестовая машина должна установить значения собственных выходных параметров и возвращаемое значение входящего вызова в соответствии с пунктами раздела 9 с заголовком «Возвращаемое значение по умолчанию».

8.9.2.9 Если входящий вызов функции, имеющей связанный процесс (с заданным приоритетом), поступает во время выполнения абстрактной испытательной машиной другого процесса, имеющего аналогичный или более высокий приоритет, вызов нового процесса должен быть добавлен в очередь, причем для каждого значения приоритета должна использоваться отдельная очередь.

8.9.2.10 Каждый раз после завершения процесса или изменения приоритета текущего процесса с высокого на низкий абстрактная испытательная машина должна определить, в какой из непустых очередей содержатся вызовы процессов с наивысшим приоритетом. Вызов процесса, находящийся в очереди дольше остальных (то есть, фактически, являющийся первым), должен быть извлечен из очереди и обработан.

8.9.2.11 Выполнение процесса должно начинаться с создания контекста выполнения, который соответствует одному конкретному вызову процесса. Контекст выполнения должен представлять собой контейнер для локальных переменных процесса (включая входные и выходные параметры) и включать в себя индикатор текущей стадии выполнения процесса.

8.9.2.12 После создания контекста выполнения необходимо создать входные параметры процесса (как локальные переменные) и присвоить им следующие значения:

a) предоставленное значение, если вызов процесса содержит элемент **<input>** для соответствующего выходного параметра (см. 8.5.2.5);

b) в противном случае — пустую строку.

8.9.2.13 После этого элементы процесса должны быть обработаны по порядку с первого до последнего до возникновения прерывания (см. 8.9.2.17).

8.9.2.14 Если прерывание не происходит, то после обработки последнего элемента каждый выходной параметр должен быть подвергнут одному из следующих действий:

a) переменной, к которой обращаются в элементе **<output>**, должно быть присвоено конечное значение выходного параметра, если вызов процесса содержит элемент **<output>** для соответствующего выходного параметра в соответствии с 8.6.2.3;

b) в противном случае конечное значение выходного параметра должно быть удалено.

8.9.2.15 После этого контекст выполнения данного вызова процесса должен быть удален.

8.9.2.16 Если процесс был вызван другим процессом, то выполнение вызывающего процесса должно быть продолжено. Если процесс является исходным процессом утверждения, то обработка всего утверждения завершается.

8.9.2.17 Прерывание выполнения процесса происходит в следующих случаях:

a) если процесс содержит элемент **<assert_condition>** с атрибутом **break_if_false**, имеющим значение **"true"**, а результат оценки условия в элементе **<assert_condition>** имеет значение **"false"** и

b) если процесс содержит элемент **<invoke>** с атрибутом **break_on_break**, имеющим значение **"true"**, когда вызываемый процесс завершается из-за прерывания, прерывание происходит и в вызывающем процессе (см. 8.9.2.18).

8.9.2.18 Если в процессе происходит прерывание (по одной из причин, указанных в 8.9.2.17, перечисления a), b)) то абстрактная испытательная машина должна:

a) пропустить обработку оставшихся элементов процесса;

b) пропустить присвоение значений выходных параметров переменным, к которым обращаются в соответствующих элементах **<output>** при вызове процесса (в случае их наличия);

c) выполнить удаление контекста выполнения вызова текущего процесса и затем возобновить выполнение вызывающего процесса (в случае его наличия).

Примечания

1 В 8.9.2.18, перечисление b) подразумевается, что если переменная, соответствующая выходному параметру, не существовала до вызова, то при возобновлении вызывающего процесса она не создается.

2 Если положение 8.9.2.17, перечисление b) относится к вызывающему процессу, то выполнение вызывающего процесса будет прекращено сразу же после его возобновления

8.9.2.19 Процессы могут создавать новые глобальные переменные или изменять значения уже существующих.

8.9.2.20 Процесс не может быть прерван другим процессом с тем же или более низким приоритетом (см. 8.9.2.9).

8.9.2.21 Процесс может быть прерван другим процессом, имеющим более высокий приоритет. Прерывание процесса может произойти в любое время: до, после или во время обработки элемента процесса, со следующими исключениями:

a) не может произойти прерывание обработки элементов **<set>**, **<add>**, **<subtract>** или **<assert_condition>**;

b) в процессе обработки элемента **<invoke>** функции стандартного интерфейса BioAPI прерывание не может произойти во время оценки условий (в случае их наличия), во время присвоения значения входным параметрам и во время передачи выходных параметров и возвращаемого значения.

Примечание — Прерывания могут происходить в процессе выполнения функции;

c) в процессе обработки элемента **<wait_until>** прерывание не может произойти в момент оценки условия.

Примечание — Прерывания могут происходить в промежутке между двумя следующими друг за другом оценками условия;

d) процесс с атрибутом **atomic,** имеющим значение **"true",** прерывать не допускается (с момента создания входных параметров и до момента удаления выходных параметров).

8.9.2.22 Имена процессов должны состоять из строк символов, указанных в ИСО/МЭК 10646, которые подходят для создания имени «NCName» (неквалифицированного имени) в пространствах имен W3C XML.

8.9.2.23 Имена процессов в пакете должны быть уникальными.

8.9.3 Примеры

```
<?xml version='1.0' encoding="utf-8"?>
<package name="73668660-1583-1AD0-A3A5-09C0FF4756E3">
    <author>
        ISO/IEC SC37
    </author>
    <description>
        Abcde abcde abcde
    </description>

<activity name="LoadAndAttach">
    <input name="BSPUuid"/>
    <input name="BSPVersion"/>
    <input name="deviceIDOrNull"/>
    <input name="BSP"/>
    <input name="eventtimeouttime"/>

    <set name="_deviceIDOrNull" var="deviceIDOrNull"/>

    <invoke function="BioSPI_BSPLoad">
        <input name="Reserved" value=""/>
        <input name="BSPUuid" var="BSPUuid"/>
        <input name="eventHandler" value="1"/>
        <input name="context" value="1"/>
        <return setvar="return"/>
    </invoke>

    <assert_condition
        response_if_true="undecided"
        break_if_false="true">
        <equal_to var1="return" var2="__BioAPI_OK"/>
    </assert_condition>

    <wait_until
        timeout_var="eventtimeouttime"
        setvar="eventtimeoutflag"
        var="_insert"/>

    <assert_condition
        response_if_true="undecided"
        break_if_false="true">
        <not var="eventtimeoutflag"/>
    </assert_condition>
    <invoke function="BioSPI_BSPAttach">
        <input name="BSPUuid" var="BSPUuid"/>
        <input name="Version" var="BSPVersion"/>
        <input name="DeviceID" var="_deviceID"/>
        <input name="Reserved1" value="0"/>
        <input name="Reserved2" value="0"/>
        <input name="BSPHandle" var="BSP"/>
        <input name="Reserved3" value="0"/>
        <input name="Reserved4" value=""/>
        <input name="Reserved5" value=""/>
        <input name="Reserved6" value=""/>
        <return setvar="return"/>
    </invoke>
    <assert_condition
            response_if_true="undecided"
            break_if_false="true">
        <equal_to var1="return" var2="__BioAPI_OK"/>
    </assert_condition>
</activity>
```

```
<activity name="EventHandler">
    <input name="BSPUuid"/>
    <input name="context"/>
    <input name="deviceID"/>
    <input name="reserved"/>
    <input name="eventType"/>
    <set name="_BSPUuid" var="BSPUuid"/>
    <set name="_context" var="context"/>
    <set name="_deviceID" var="deviceID">
        <only_if>
            <not>
                <existing var="_deviceID"/>
            </not>
            <or>
                <equal_to var1="eventType" var2="__BioAPI_NOTIFY_INSERT"/>
                <equal_to var1="eventType"
                        var2="__BioAPI_NOTIFY_SOURCE_PRESENT"/>
            </or>
            <or>
                <equal_to var1="_deviceIDOrNull" value2="0"/>
                <equal_to var1="_deviceIDOrNull" var2="deviceID"/>
            </or>
        </only_if>
    </set>
    <set name="_insert" value="true">
        <only_if>
            <equal_to var1="eventType" var2="__BioAPI_NOTIFY_INSERT"/>
            <equal_to var1="_deviceID" var2="deviceID"/>
        </only_if>
    </set>
    <set name="_insert" value="false">
        <only_if>
            <equal_to var1="eventType" var2="__BioAPI_NOTIFY_REMOVE"/>
            <equal_to var1="_deviceID" var2="deviceID"/>
        </only_if>
    </set>
    <set name="_sourcePresent" value="true">
        <only_if>
            <equal_to var1="eventType" var2="__BioAPI_NOTIFY_SOURCE_PRESENT"/>
            <equal_to var1="_deviceID" var2="deviceID"/>
        </only_if>
    </set>
    <set name="_sourcePresent" value="false">
        <only_if>
            <equal_to var1="eventType"
                    var2="__BioAPI_NOTIFY_SOURCE_REMOVED"/>
            <equal_to var1="_deviceID" var2="deviceID"/>
        </only_if>
    </set>
    <set name="_eventtype" var="eventType">
        <only_if>
            <equal_to var1="_deviceID" var2="deviceID"/>
        </only_if>
    </set>
    <assert_condition
            response_if_false="fail">
        <equal_to var1="reserved" value2="0"/>
    </assert_condition>
```

```
</activity>

<activity name="CreateTemplate">
    <input name="BSPUuid"/>
    <input name="BSPVersion"/>
    <input name="deviceIDOrNull"/>
    <input name="inserttimeouttime"/>
    <input name="sourcepresenttimeouttime"/>
    <input name="capturetimeouttime"/>

    <invoke activity="LoadAndAttach" break_on_break="true">
        <input name="BSPUuid" var="BSPUuid"/>
        <input name="BSPVersion" var="BSPVersion"/>
        <input name="deviceIDOrNull" var="deviceIDOrNull"/>
        <input name="BSP" value="1"/>
        <input name="eventtimeouttime" var="inserttimeouttime"/>
    </invoke>

    <wait_until
          timeout_var="sourcepresenttimeouttime"
          setvar="eventtimeoutflag"
          var="_sourcePresent"/>

    <assert_condition
          response_if_true="undecided"
          break_if_false="true">
        <not var="eventtimeoutflag"/>
    </assert_condition>

    <invoke function="BioSPI_Capture">
        <input name="BSPHandle" value="1"/>
        <input name="Purpose" var="__BioAPI_PURPOSE_ENROLL"/>
        <input name="Timeout" var="capturetimeouttime"/>
        <output name="CapturedBIR" setvar="bir"/>
        <output name="AuditData" setvar="auditbir"/>
        <return setvar="return"/>
    </invoke>

    ...
</activity>
</package>
```

8.10 Элемент <input> (дочерний для элемента <activity>)

8.10.1 Синтаксис

8.10.1.1 Данный элемент должен иметь следующий атрибут

- **name** (обязательный атрибут) — значением этого атрибута должно быть корректное имя локальной переменной (см. 7.2), которое является именем входного параметра процесса.

8.10.1.2 Данный элемент не должен иметь текстового содержимого.

8.10.2 Семантика

8.10.2.1 Этот элемент представляет собой входной параметр процесса.

8.10.2.2 Входные параметры процессов являются локальными переменными (см. 7.2).

8.10.2.3 Именами входных параметров процесса должны быть корректные имена локальных переменных (см. 7.2).

8.10.2.4 Имена всех локальных переменных процесса (включая входные и выходные параметры процесса) должны быть уникальными.

8.11 Элемент <output> (дочерний для элемента <activity>)

8.11.1 Синтаксис

8.11.1.1 Данный элемент должен иметь следующий атрибут:

- **name** (обязательный атрибут) — значением этого атрибута должно быть корректное имя локальной переменной (см. 7.2), которое должно быть именем выходного параметра процесса.

8.11.1.2 Данный элемент не должен иметь текстового содержимого.

8.11.2 Семантика

8.11.2.1 Данный элемент представляет собой входной параметр процесса.

8.11.2.2 Выходные параметры процессов являются локальными переменными (см. 7.2).

8.11.2.3 Имена выходных параметров процессов должны быть корректными именами локальных переменных (см. 7.2).

8.11.2.4 Имена всех локальных переменных процесса (включая входные и выходные параметры процесса) должны быть уникальными.

8.12 Элемент <set>

8.12.1 Синтаксис

8.12.1.1 Данный элемент должен иметь следующие атрибуты:

a) **name** (обязательный атрибут) — значением этого атрибута должно быть корректное имя переменной (см. 7.2), которой должно быть присвоено значение или которая подлежит созданию с последующим присвоением значения;

b) **value** (необязательный атрибут) — если этот атрибут присутствует, то он должен содержать значение, которое должно быть присвоено переменной с именем, определяемым обязательным атрибутом **name**;

c) **var** (необязательный атрибут) — если этот атрибут присутствует, то значением атрибута должно быть корректное имя переменной (см. 7.2), значение которой должно быть присвоено переменной с именем, определяемым обязательным атрибутом **name**;

8.12.1.2 Должен присутствовать только один из атрибутов **value** или **var**.

8.12.1.3 В состав данного элемента может также входить:

- необязательный элемент **<only_if>**, представляющий собой условие и определенный в 8.16.

8.12.2 Семантика

8.12.2.1 Этот элемент представляет собой присвоение значения переменной с возможной проверкой условия.

8.12.2.2 Значением атрибута **name** должно быть корректное имя глобальной или локальной переменной (см. 7.2). Значение атрибута **name** не должно быть входным параметром обрабатываемого утверждения (см. 8.3) и не должно начинаться с двух следующих друг за другом символов «НИЖНЕЕ ПОДЧЕРКИВАНИЕ» («__»). Переменная может существовать до присвоения значения, а может быть новой переменной, созданной в процессе присвоения значения.

8.12.2.3 Если присутствует дочерний элемент **<only_if>**, необходимо провести проверку условия (см. 8.16).

8.12.2.4 Результатом проверки должно быть корректное логическое значение (см. 7.5).

8.12.2.5 Если результат проверки имеет значение **"true"**, то присвоение значения переменной, имя которой определено значением атрибута **name**, осуществляется следующим образом.

8.12.2.5.1 Если переменной не существует, ее необходимо создать.

8.12.2.5.2 Если присутствует атрибут **var**, то его значением должно быть имя существующей переменной. Значение этой переменной должно быть присвоено переменной, имя которой определяется значением атрибута **name**.

8.12.2.5.3 Если присутствует атрибут **value**, то его значение должно быть присвоено переменной, имя которой определяется значением атрибута **name**.

8.13 Элемент <add>

8.13.1 Синтаксис

8.13.1.1 Данный элемент должен иметь следующие атрибуты:

a) **name** (обязательный атрибут) — значением этого атрибута должно быть корректное имя переменной (см. 7.2), значение которой должно быть изменено;

b) **value** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть целое число, которое необходимо прибавить к текущему значению (представляющему собой также целое число) переменной, имя которой определяется обязательным атрибутом **name**;

c) **var** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2), значение которой должно быть целым числом, которое необходимо прибавить к текущему значению переменной, имя которой определяется обязательным атрибутом **name**.

8.13.1.2 Должен присутствовать только один из атрибутов — **value** или **var**.

8.13.1.3 В состав данного элемента может также входить:

- дополнительный элемент **<only_if>**, представляющий собой условие и определенный в 8.16.

8.13.2 Семантика

8.13.2.1 Этот элемент представляет собой прибавление целого числа к значению (также являющемуся целым числом) переменной с возможной проверкой условия.

8.13.2.2 Значением атрибута **name** должно быть корректное имя глобальной или локальной переменной (см. 7.2). Значение атрибута **name** не должно быть входным параметром обрабатываемого утверждения (см. 8.3) и не должно начинаться с двух следующих друг за другом символов «НИЖНЕЕ ПОДЧЕРКИВАНИЕ» («__»). Переменная, имя которой определяется значением атрибута **name**, должна существовать до присвоения значения, а ее значением должно быть корректное представлением целого числа (см. 7.4).

8.13.2.3 Если присутствует дочерний элемент **<only_if>**, необходимо провести проверку условия (см. 8.16).

8.13.2.4 Результатом проверки условия должно быть корректное логическое значение (см. 7.5).

8.13.2.5 Если результатом проверки условия является значение **«true»**, то переменной, имя которой определяется значением атрибута **name**, присваивается новое значение следующим образом.

8.13.2.5.1 Переменной должно быть присвоено каноническое представление целого числа, которое является суммой двух операндов. Первый операнд является целым числом, соответствующим текущему значению переменной. Второй операнд определяют следующим образом.

8.13.2.5.2 Если присутствует атрибут **var**, то его значением должно быть имя существующей переменной. Значением этой переменной должно быть корректное представление целого числа (см. 7.4), которое является вторым операндом.

8.13.2.5.3 Если присутствует атрибут **value**, то его значением должно быть корректное представление целого числа (см. 7.4), которое является вторым операндом.

8.14 Элемент <subtract>

8.14.1 Синтаксис

8.14.1.1 Данный элемент должен иметь следующие атрибуты:

a) **name** (обязательный атрибут) — значением этого атрибута должно быть корректное имя переменной (см. 7.2), значение которой должно быть изменено;

b) **value** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть целое число, которое необходимо вычесть из текущего значения (представляющего собой целое число) переменной, определяемой значением обязательного атрибута **name**;

c) **var** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2), значение которой представляет собой целое число, которое необходимо вычесть из текущего значения переменной, определяемой значением обязательного атрибута **name**.

8.14.1.2 Должен присутствовать только один из атрибутов — **value** или **var**.

8.14.1.3 В состав элемента может также входить:

- дополнительный элемент **<only_if>**, представляющий собой условие и определенный в 8.16.

8.14.2 Семантика

8.14.2.1 Этот элемент представляет собой вычитание целого числа из значения (являющегося также целым числом) переменной с возможной проверкой условия.

8.14.2.2 Значением атрибута **name** должно быть корректное имя глобальной или локальной переменной (см. 7.2). Значение атрибута **name** не должно быть входным параметром обрабатываемого утверждения (см. 8.3) и не должно начинаться с двух следующих друг за другом символов «НИЖНЕЕ ПОДЧЕРКИВАНИЕ» («__»). Переменная, имя которой определяется значением атрибута **name**, должна существовать до присвоения значения, а ее значением должно быть корректное представление целого числа (см. 7.4).

8.14.2.3 Если присутствует дочерний элемент **<only_if>**, необходимо провести проверку условия (см. 8.16).

8.14.2.4 Результатом проверки должно быть корректное представление логического значения (см. 7.5).

8.14.2.5 Если результатом проверки условия является значение **"true"**, то переменной, имя которой определяется значением атрибута **name**, присваивается новое значение следующим образом.

8.14.2.5.1 Переменной должно быть присвоено каноническое представление целого числа, которое является разностью двух операндов. Первый операнд является целым числом, соответствующим текущему значению переменной. Второй операнд определяют следующим образом.

8.14.2.5.2 Если присутствует атрибут **var**, то его значением должно быть имя существующей переменной. Значением этой переменной должно быть корректное представление целого числа (см. 7.4), которое является вторым операндом.

8.14.2.5.3 Если присутствует атрибут **value**, то его значением должно быть корректное представлением целого числа (см. 7.4), которое является вторым операндом.

8.15 Элемент <invoke> (дочерний для элемента <activity>)

8.15.1 Синтаксис

8.15.1.1 Данный элемент должен иметь следующие атрибуты:

a) **activity** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть имя процесса;

b) **package** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть имя пакета, который содержит процесс, определенный значением атрибута **activity**;

c) **break_on_break** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть одно из логических выражений **"true"** или **"false"**; этот атрибут определяет, приводит ли прерывание в вызванном процессе к прекращению выполнения текущего (вызывающего) процесса; по умолчанию атрибут **break_on_break** имеет значение **"false"**;

d) **function** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть имя функции стандартного интерфейса BioAPI;

e) **timeout_value** (необязательный атрибут) — если этот атрибут присутствует, то его значение определяет максимальную допустимую продолжительность (в миллисекундах) вызова;

f) **timeout_var** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2), значение которой определяет максимальную допустимую продолжительность (в миллисекундах) вызова;

g) **setvar** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2).

8.15.1.2 Должен присутствовать только один из атрибутов **activity** или **function**.

8.15.1.3 Атрибуты **package** и **break_on_break** могут присутствовать только в том случае, если присутствует атрибут **activity**.

8.15.1.4 Атрибуты **timeout_var**, **timeout_value** и **setvar** могут присутствовать только в том случае, если присутствует атрибут **function**.

8.15.1.5 Должен присутствовать только один из атрибутов **timeout_var** или **timeout_value**.

8.15.1.6 Атрибут **setvar** может присутствовать только в том случае, если присутствует атрибут **timeout_var** или атрибут **timeout_value**.

8.15.1.7 В состав данного элемента в указанном порядке должны входить:

a) необязательный элемент **<only_if>**, представляющий собой условие и определенный в 8.16;

b) ноль или более элементов **<input>** — этот элемент представляет собой значение входного параметра процесса или вызываемой функции стандартного интерфейса BioAPI и определен в 8.5;

c) ноль или более элементов **<output>** — этот элемент указывает на переменную, которой должно быть присвоено значение выходного параметра процесса или вызываемой функции, и определен в 8.6;

d) необязательный элемент **<return>** — этот элемент указывает на переменную, которой должно быть присвоено возвращаемое значение вызываемой функции, и определен в 8.7.

8.15.2 Семантика

8.15.2.1 Этот элемент представляет собой вызов процессом другого процесса или функции стандартного интерфейса BioAPI.

8.15.2.2 Если присутствует дочерний элемент **<only_if>**, необходимо провести проверку условия (см. 8.16).

8.15.2.3 Результатом проверки условия должно быть корректное представление логическое значения (см. 7.5).

8.15.2.4 Если результат проверки условия имеет значение **"true"**, то должен быть осуществлен вызов процесса или функции стандартного интерфейса BioAPI в соответствии с изложенным ниже, в противном случае дальнейшую обработку элемента **<invoke>** прекращают.

8.15.2.5 При вызове процесса он должен существовать и находиться или в пакете, содержащем вызывающий процесс, или в любом другом пакете (см. 8.15.2.6.4). При вызове функции она должна быть одной из функций стандартного интерфейса BioAPI в соответствии с разделом 9.

8.15.2.6 При вызове процесса должны быть соблюдены пять условий, описанных в 8.15.2.6.1—8.15.2.6.5.

8.15.2.6.1 Набор элементов **<input>** вызова должен соответствовать входным параметрам процесса в соответствии с 8.5.2.2.

8.15.2.6.2 Набор элементов **<output>** вызова должен соответствовать выходным параметрам процесса в соответствии с 8.6.2.2.

8.15.2.6.3 Элемент **<return>** должен отсутствовать.

8.15.2.6.4 Если атрибут **package** присутствует, то он должен определять пакет, в котором находится вызываемый процесс. Если вызывающий и вызываемый процессы находятся в разных пакетах, атрибут **package** является обязательным. В противном случае атрибут **package** является необязательным.

8.15.2.6.5 Если атрибут **break_on_break** присутствует и имеет значение **"true"**, а вызываемый процесс завершается из-за возникновения прерывания (см. 8.9.2.17), то и текущий (вызывающий) процесс должен быть завершен.

8.15.2.7 При вызове функции должны быть соблюдены три условия, описанные в 8.15.2.7.1—8.15.2.7.3.

8.15.2.7.1 Набор элементов **<input>** вызова должен соответствовать входным параметрам функции в соответствии с 8.5.2.2.

8.15.2.7.2 Набор элементов **<output>** вызова должен соответствовать выходным параметрам функции в соответствии с 8.6.2.2.

8.15.2.7.3 Допускается присутствие элемента **<return>** (см. 8.7.2.2)

8.15.2.8 Если присутствует атрибут **timeout_value**, то его значение должно быть корректным представлением целого числа (см. 7.4), которое определяет максимальную допустимую продолжительность вызова (в миллисекундах). Если присутствует атрибут **timeout_var**, то его значение должно быть именем существующей переменной, а значение этой переменной должно быть корректным представлением целого числа (см. 7.4), которое определяет максимальную допустимую продолжительность вызова. В противном случае максимальную допустимую продолжительность вызова не устанавливают.

Примечание — Тестируемая реализация, соответствующая требованиям настоящего стандарта, может прекратить проведение испытания, время выполнения которого превышает установленные допустимые пределы. Данное действие не следует рассматривать как противоречащее требованиям, установленным в 8.15.2.8.

8.15.2.9 Вызов должен продолжаться независимо от максимальной продолжительности, даже если она равна нулю или отрицательная. В этом случае абстрактная испытательная машина сразу же должна определить, что максимальная допустимая продолжительность выполнения вызова превышена.

8.15.2.10 Если присутствует атрибут **setvar**, то должны быть соблюдены четыре условия, описанные в 8.15.2.10.1—8.15.2.10.4.

8.15.2.10.1 Значение атрибута **setvar** должно быть корректным именем глобальной или локальной переменной (см. 7.2). Значение атрибута **setvar** не должно быть входным параметром обрабатываемого утверждения (см. 8.3) и не должно начинаться с двух следующих друг за другом символов «НИЖНЕЕ ПОДЧЕРКИВАНИЕ» («__»). Переменная, имя которой определяется значением атрибута **setvar**, может существовать до присвоения ей значения или может быть новой переменной, созданной во время присвоения.

8.15.2.10.2 Если возврат из вызванной функции произошел после превышения максимальной допустимой продолжительности вызова функции, то переменная должна быть создана (если она не существует), и этой переменной должно быть присвоено значение **"true"**.

8.15.2.10.3 Если возврат из вызванной функции произошел до превышения максимальной допустимой продолжительности вызова функции, то переменная должна быть создана (если она не существует), и этой переменной должно быть присвоено значение **"false"**.

8.15.2.10.4 Переменная не должна создаваться (если она не существует до вызова функции) и ей не должно присваиваться значение, кроме случаев, описанных в 8.15.2.10.2 и 8.15.2.10.3.

8.15.2.11 Когда процесс или функция вызваны, выполнение текущего процесса должно быть приостановлено до тех пор, пока вызванные функция или процесс не завершатся, или до тех пор, пока не будет превышена максимальная допустимая продолжительность вызова.

8.15.2.12 До начала выполнения вызванных процесса или функции всем входным параметрам, для которых в вызове (при их наличии) был представлен элемент **<input>**, должны быть присвоены значения переменных или параметров, представленных в элементах **<input>** в соответствии с 8.5.2.5. Остальным входным параметрам в качестве значения должна быть присвоена пустая строка символов.

8.15.2.13 Если вызванные процесс или функция завершаются и не превышают максимальную допустимую продолжительность вызова, необходимо выполнить два действия, описанные в 8.15.2.13.1 и 8.15.2.13.2.

8.15.2.13.1 Значение каждого выходного параметра, для которого в вызове (в случае их наличия) предоставлен элемент **<output>**, должно быть присвоено переменной, указанной в элементе **<output>** в соответствии с 8.6.2.3.

8.15.2.13.2 Если осуществляется вызов функции, в котором присутствует элемент **<return>**, возвращаемое значение должно быть присвоено переменной, указанной в элементе **<return>**, в соответствии с 8.7.2.3.

8.15.2.14 Если превышена максимальная допустимая продолжительность вызова функции, необходимо выполнить два действия, описанные в 8.15.2.14.1 и 8.15.2.14.2.

8.15.2.14.1 Присвоение выходных параметров и возвращаемых значений переменным производиться не должно. Более того, если переменная, соответствующая выходным параметрам или возвращаемым значениям, не существовала до вызова, то она не должна создаваться при возобновлении выполнения вызывающего процесса.

8.15.2.14.2 Реализация не должна предпринимать попытки прерывания собственной функции, даже если реализация определила, что выполнение собственной функции будет длиться бесконечно.

8.15.2.15 Выполнение вызывающего процесса (при его наличии) должно быть продолжено (с учетом положений 8.9.2.17).

8.15.2.16 Собственная функция, выполнение которой превысило максимальную допустимую продолжительность вызова, в итоге может завершиться. При возврате из вызова собственной функции ее собственные выходные параметры (при их наличии) и возвращаемое значение должны быть удалены.

8.15.3 Примеры

```
<invoke activity="LoadAndAttach">
   <only_if var="myflag"/>
   <output name="BSP" setvar="BSP"/>
</invoke>

<invoke function="BioSPI_CreateTemplate">
   <input name="BSPHandle" var="BSP"/>
   <input name="CapturedBIR" var="inputbir"/>
   <input name="StoredTemplate" value=""/>
   <input name="Payload" value=""/>
   <output name="NewTemplate" setvar="template"/>
   <return setvar="return"/>
</invoke>
```

8.16 Элемент <only_if>

8.16.1 Синтаксис

8.16.1.1 Данный элемент должен иметь следующий атрибут:

- **var** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2).

8.16.1.2 Данный элемент должен включать в себя:

a) необязательный элемент **<description>**, который (при наличии) должен содержать описание условия (срока символов);

b) ноль или более появлений любого из следующих элементов в произвольном порядке:

1) **<and>** — этот элемент представляет собой логический оператор AND («И») и определен в 8.19;

2) **<or>** — этот элемент представляет собой логический оператор OR («ИЛИ») и определен в 8.20;

3) **<xor>** — этот элемент представляет собой логический оператор XOR («исключающее ИЛИ») и определен в 8.21;

4) **<not>** — этот элемент представляет собой логический оператор NOT («НЕ») и определен в 8.22;

5) **<equal_to>** — этот элемент представляет собой числовой оператор «equal to» («равно») и определен в 8.23;

6) **<not_equal_to>** — этот элемент представляет собой числовой оператор «not equal to» («не равно») и определен в 8.24;

7) **<greater_than>** — этот элемент представляет собой числовой оператор «greater than» («больше») и определен в 8.25;

8) **<greater_than_or_equal_to>** — этот элемент представляет собой числовой оператор «greater than or equal to» («больше или равно») и определен в 8.26;

9) **<less_than>** — этот элемент представляет собой числовой оператор «less than» («меньше») и определен в 8.27;

10) **<less_than_or_equal_to>** — этот элемент представляет собой числовой оператор «less than or equal to» («меньше или равно») и определен в 8.28;

11) **<same_as>** — этот элемент представляет собой оператор для работы со строками символов «equals» («соответствует») и определен в 8.29;

12) **<not_same_as>** — этот элемент представляет собой оператор для работы со строками символов «not equal to» («не соответствует») и определен в 8.30;

13) **<existing>** — этот элемент представляет собой проверку условия, что переменная существует, и определен в 8.31;

14) **<not_existing>** — этот элемент представляет собой проверку условия, что переменной не существует, и определен в 8.32.

8.16.1.3 Если атрибут **var** присутствует, то элемент **<only_if>** не должен иметь текстового содержимого и дочерних элементов.

8.16.2 Семантика

8.16.2.1 Этот элемент представляет собой условие, основанное на одной переменной или на сочетании переменных, значений, логических операторов, числовых операторов и других операторов.

8.16.2.2 Если присутствует атрибут **var**, то его значением должно быть имя существующей переменной, а значение переменной должно быть корректным представлением логического значения (см. 7.5). Результат проверки условия должен совпадать со значением переменной.

8.16.2.3 Если атрибут **var** отсутствует и нет дочерних элементов, то результатом проверки условия должно быть значение **"true"**.

8.16.2.4 Если присутствуют дочерние элементы, то результатом проверки условия каждого дочернего элемента должно быть корректное представление логического значения (см. 7.5). Результатом проверки условия в элементе **<only_if>** должно быть значение **"true"**, если результаты проверки условий всех дочерних элементов имеют значение **"true"**, в противном случае результатом проверки условия в элементе **<only_if>** должно быть значение **"false"**.

8.16.2.5 Проверка условий дочерних элементов проводится во время обработки родительского элемента (см. 8.12.2.3 и 8.15.2.2). Если в результате проверки условий дочерних элементов получают значение **"true"**, то родительский элемент обрабатывается в обычном режиме (как если бы он не содержал дочерних элементов **<only_if>**), в противном случае его обработка не оказывает никакого влияния на окончательный результат проверки условия.

8.17 Элемент <wait_until>

8.17.1 Синтаксис

8.17.1.1 Данный элемент должен иметь следующие атрибуты:

a) **timeout_value** (необязательный атрибут) — если этот атрибут присутствует, то его значение определяет максимальную допустимую продолжительность (в миллисекундах) ожидания;

b) **timeout_var** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2), значение которой определяет максимальную допустимую продолжительность (в миллисекундах) ожидания;

c) **setvar** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2);

d) **var** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2).

8.17.1.2 Должен присутствовать только один из атрибутов **timeout_var** или **timeout_value**.

8.17.1.3 Атрибут **setvar** не должен присутствовать в том случае, если не присутствует атрибут **timeout_value** или атрибут **timeout_var**.

8.17.1.4 Содержимое этого элемента должно включать в себя те же дочерние элементы, которые содержит элемент **<only_if>** (см. 8.16).

8.17.1.5 Если атрибут **var** присутствует, то элемент **<wait_until>** не должен иметь текстового содержимого и дочерних элементов.

8.17.2 Семантика

8.17.2.1 Этот элемент определяет приостановку выполнения текущего процесса до тех пор, пока не подтвердится определенное условие или пока не будет достигнута максимальная допустимая продолжительность.

8.17.2.2 Условие в элементе **<wait_until>** основано на одной переменной или на сочетании переменных, значений, логических операторов, числовых операторов и иных операторов.

8.17.2.3 Если атрибут **var** присутствует, то его значением должно быть имя существующей переменной, а значением переменной должно быть корректное представление логического значения (см. 7.5). Результат проверки условия должен совпадать со значением переменной.

8.17.2.4 Если атрибут **var** отсутствует и нет дочерних элементов, то результатом проверки условия должно быть значение **"true"**.

8.17.2.5 Если присутствуют дочерние элементы, то результатом проверки условия каждого дочернего элемента должно быть корректное представление логического значения (см. 7.5). Результатом проверки условия в элементе **<wait_until>** должно быть значение **"true"**, если результаты проверки условий всех дочерних элементов имеют значение **"true"**, в противном случае результатом проверки условия в элементе **<wait_until>** должно быть значение **"false"**.

8.17.2.6 Если атрибут **timeout_value** присутствует, то его значением должно быть корректное представление целого числа (см. 7.4), которое определяет максимальную допустимую продолжительность ожидания (в миллисекундах). Если присутствует атрибут **timeout_var**, то его значением должно быть имя существующей переменной, а значением этой переменной должно быть корректное представление целого числа (см. 7.4), которое определяет максимальную допустимую продолжительность. В противном случае ожидание может длиться (концептуально) в течение неопределенного времени.

Примечание — См. примечание к 8.15.2.8.

8.17.2.7 Проверка условия должна проводиться многократно и постоянно в течение времени ожидания до тех пор, пока в результате проверки условия не получено значение **"true"** или не достигнута максимальная допустимая продолжительность ожидания (в случае ее наличия).

Примечание — На практике необходимо проверять условие в дискретных точках во времени каждый раз, когда есть вероятность того, что входящий вызов компонента тестирования или прошедшее время повлияли на значение условия. Испытания, проводимые в соответствии с требованиями настоящего стандарта, должны гарантировать, что любой переход от **"false"** к **"true"** в течение времени ожидания зафиксирован, даже если условие имеет значение **"true"** в течение очень короткого промежутка времени, а затем снова принимает значение **"false"**.

8.17.2.8 Проверку условия следует проводить не менее одного раза. Если в результате первой проверки получено значение **"true"**, то в этом случае максимальную допустимую продолжительность не считают превышенной.

Примечание — Проверку условия следует проводить даже в том случае, когда максимальная допустимая продолжительность равна нулю или является отрицательным значением.

8.17.2.9 Если присутствует атрибут **setvar**, то должны быть соблюдены четыре условия, описанные в 8.17.2.9.1— 8.17.2.9.4.

8.17.2.9.1 Значением атрибута **setvar** должно быть корректное имя глобальной или локальной переменной (см. 7.2). Значением атрибута **setvar** не должен быть входной параметр обрабатываемого утверждения (см. 8.3) и оно не должно начинаться с двух следующих друг за другом символов «НИЖНЕЕ ПОДЧЕРКИВАНИЕ» («__»). Переменная, имя которой определяется значением атрибута **setvar**, может существовать до присвоения ей значения или может быть новой переменной, созданной в процессе присвоения.

8.17.2.9.2 Если максимальная допустимая продолжительность превышена, в то время как результат проверки условия все еще имеет значение **"false"**, то переменная должна быть создана (если она не существует) и ей должно быть присвоено значение **"true"**.

8.17.2.9.3 Если в начале ожидания результат проверки условия уже имеет значение **"true"** или приобретает значение **"true"** в течение ожидания, то переменная должна быть создана (если она не существует) и ей должно быть присвоено значение **"false"**.

8.17.2.9.4 Переменная не должна создаваться (если она не существует до ожидания) и ей не должны присваиваться значения, кроме случаев, указанных в 8.17.2.9.2 и 8.17.2.9.3.

Примечание — Это предполагает, что переменная не может быть инициализирована до начала ожидания.

8.18 Элемент <assert_condition>

8.18.1 Синтаксис

8.18.1.1 Данный элемент должен иметь следующие атрибуты:

a) **response_if_true** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть **"pass"** или **"undecided"**; этот атрибут определяет, какой выдать ответ о соответствии, если в результате проверки условия получено значение **"true"**; по умолчанию атрибут **response_if_true** имеет значение **"pass"**;

b) **response_if_false** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть **"fail"** или **"undecided"**; этот атрибут определяет, какой выдать ответ о соответствии, если в результате проверки условия получено значение **"false"**; по умолчанию атрибут **response_if_false** имеет значение **"fail"**.

c) **break_if_false** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть **"false"** или **"true"**; этот атрибут определяет, должно ли выполнение текущей операции быть прекращено в случае, если в результате проверки условия получено значение **"false"**; по умолчанию атрибут **break_if_false** имеет значение **"false"**;

d) **var** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2).

8.18.1.2 Содержимое этого элемента должно включать в себя те же дочерние элементы, которые включает элемент **<only_if>** (см. 8.16).

8.18.1.3 Если атрибут **var** присутствует, то элемент не должен иметь текстового содержимого и дочерних элементов.

8.18.2 Семантика

8.18.2.1 Этот элемент определяет:

a) условие, которое позволяет установить определенный аспект функционирования тестируемой реализации;

b) заключение о соответствии (**"pass"**, **"undecided"**), которое необходимо выдать в случае выполнения условия и

c) заключение о соответствии (**"fail"**, **"undecided"**), которое необходимо выдать в случае невыполнения условия.

8.18.2.2 Атрибуты элемента **<assert_condition>** позволяют определить критерий соответствия, согласно которому выполнение определенного условия приводит к выдаче заключения о соответствии **"pass"**, а невыполнение — к выдаче заключения о соответствии **"undicided"**, в то время как невыполнение другого условия приводит к выдаче заключения о соответствии **"fail"**, а его выполнение — к выдаче заключения о соответствии **"undecided"**.

8.18.2.3 Атрибут **break_if_false** определяет, станет ли отрицательный результат проверки условия (**"false"**) причиной того, что абстрактная испытательная машина прекратит выполнение текущего процесса (см. 8.9.2.18). Этот атрибут позволяет обрабатывать такие типовые случаи, в которых совокупность последовательных действий, соответствующая выдаче определенного заключения о соответствии, является бессмысленной.

8.18.2.4 Условие в элементе **<assert_condition>** основано на одной переменной или на сочетании переменных, значений, логических операторов, числовых операторов и иных операторов.

8.18.2.5 Если атрибут **var** присутствует, то его значение должно быть именем существующей переменной, а значение переменной должно быть корректным представлением логического значения (см. 7.5). Результат проверки условия должен совпадать со значением переменной.

8.18.2.6 Если атрибут **var** отсутствует и нет дочерних элементов, то результатом проверки условия должно быть значение **"true"**.

8.18.2.7 Если присутствуют дочерние элементы, то результатом проверки условия каждого дочернего элемента должно быть корректное представление логического значения (см. 7.5). Результатом проверки условия в элементе **<assert_condition>** должно быть значение **"true"**, если результаты проверки условий всех дочерних элементов имеют значение **"true"**, в противном случае результатом проверки условия в элементе **<assert_condition>** должно быть значение **"false"**.

8.18.2.8 Если в результате проверки условия получено значение **"true"**, то соответствующее заключение о соответствии (см. 8.18.2.1, перечисление b)) должно быть выдано согласно требованиям раздела 11.

8.18.2.9 Если в результате проверки условия получено значение **"false"**, то соответствующее заключение о соответствии (см. 8.18.2.1, перечисление c)) должно быть выдано согласно требованиям раздела 11. Если атрибут **break_if_false** имеет значение **"true"**, то выполнение текущего процесса должно быть прекращено (см. 8.9.2.18).

8.19 Элемент <and>

8.19.1 Синтаксис

8.19.1.1 Данный элемент должен иметь следующие атрибуты:

a) **var1** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2);

b) **var2** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2).

8.19.1.2 Содержимое этого элемента должно включать в себя те же дочерние элементы, что и содержимое элемента **<only_if>** (см. 8.16), но с ограничением, определенным в 8.19.1.3.

8.19.1.3 Если присутствует один из атрибутов **var1** или **var2**, то должны присутствовать оба атрибута (**var1** и **var2**), а элемент **<and>** не должен иметь текстового содержимого и дочерних элементов. В противном случае элемент **<and>** должен иметь два или более дочерних элемента.

8.19.2 Семантика

8.19.2.1 Этот элемент представляет собой условие, основанное на двух переменных или на сочетании переменных, значений, логических операторов, числовых операторов и иных операторов. Проверка условия осуществляется во время обработки родительского элемента.

8.19.2.2 Этот элемент представляет собой логический оператор AND («И»), чьи операнды могут быть предоставлены в качестве атрибутов этого элемента или в качестве дочерних элементов. Если применяются атрибуты, то число операндов должно равняться двум, в противном случае — двум или больше.

8.19.2.3 Если присутствует атрибут **var1**, то его значением должно быть имя существующей переменной, а значением переменной должно быть корректное представление логического значения (см. 7.5).

8.19.2.4 Если присутствует атрибут **var2**, то его значением должно быть имя существующей переменной, а значением переменной должно быть корректное представление логического значения (см. 7.5).

8.19.2.5 Если присутствуют оба атрибута **var1** и **var2**, то операндам присваиваются значения двух переменных, имена которых определены значениями атрибутов **var1** и **var2**. В противном случае операндам присваиваются значения, полученные из текстового содержимого дочерних элементов элемента **<and>**.

8.19.2.6 Каждый операнд должен быть корректным представлением логического значения (см. 7.5). В результате проверки условия должно быть получено значение **"true"** в том случае, если все операнды имеют значение **"true"**, в противном случае должно быть получено значение **"false"**.

8.20 Элемент <or>

8.20.1 Синтаксис

8.20.1.1 Данный элемент должен иметь следующие атрибуты:

a) **var1** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2);

b) **var2** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2).

8.20.1.2 Содержимое этого элемента должно включать в себя те же дочерние элементы, которые содержит элемент **<only_if>** (см. 8.16), но с ограничением, указанным в 8.20.1.3.

8.20.1.3 Если присутствует один из атрибутов **var1** или **var2**, то должны присутствовать оба атрибута (**var1** и **var2**), а элемент **<or>** не должен иметь текстового содержимого и дочерних элементов. В противном случае элемент **<or>** должен иметь два или более дочерних элемента.

8.20.2 Семантика

8.20.2.1 Этот элемент представляет собой условие, основанное на двух переменных или на сочетании переменных, значений, логических операторов, числовых операторов и иных операторов. Проверка условия осуществляется во время обработки родительского элемента.

8.20.2.2 Этот элемент представляет собой логический оператор OR («ИЛИ»), чьи операнды могут быть предоставлены в качестве атрибутов этого элемента или в качестве дочерних элементов. Если применяются атрибуты, то число операндов должно равняться двум, в противном случае — двум или больше.

8.20.2.3 Если присутствует атрибут **var1**, то его значением должно быть имя существующей переменной, а значением переменной должно быть корректное представление логического значения (см. 7.5).

8.20.2.4 Если присутствует атрибут **var2**, то его значением должно быть имя существующей переменной, а значением переменной должно быть корректное представление логического значения (см. 7.5).

8.20.2.5 Если присутствуют оба атрибута **var1** и **var2**, то операндам присваиваются значения двух переменных, имена которых определены значениями атрибутов **var1** и **var2**. В противном случае операндам присваиваются значения, полученные из текстового содержимого дочерних элементов элемента **<or>**.

8.20.2.6 Каждый операнд должен быть корректным представлением логического значения (см. 7.5). В результате проверки условия должно быть получено значение **"false"** в том случае, если все операнды имеют значение **"false"**, в противном случае должно быть получено значение **"true"**.

8.21 Элемент <xor>

8.21.1 Синтаксис

8.21.1.1 Данный элемент должен иметь следующие атрибуты:

a) **var1** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2);

b) **var2** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2).

8.21.1.2 Содержимое этого элемента должно включать в себя те же дочерние элементы, которые содержит элемент **<only_if>** (см. 8.16), но с ограничением, указанным в 8.21.1.3.

8.21.1.3 Если присутствует один из атрибутов **var1** или **var2**, то должны присутствовать оба атрибута (**var1** и **var2**), а элемент **<xor>** не должен иметь текстового содержимого и дочерних элементов. В противном случае элемент **<xor>** должен иметь два дочерних элемента.

8.21.2 Семантика

8.21.2.1 Этот элемент представляет собой условие, основанное на двух переменных или на сочетании переменных, значений, логических операторов, числовых операторов и иных операторов. Проверка условия осуществляется во время обработки родительского элемента.

8.21.2.2 Этот элемент представляет собой логический оператор XOR (eXclusive OR, «исключающее ИЛИ»), чьи операнды могут быть предоставлены в качестве атрибутов этого элемента или в качестве дочерних элементов. Число операндов должно быть равно двум.

8.21.2.3 Если присутствует атрибут **var1**, то его значением должно быть имя существующей переменной, а значением переменной должно быть корректное представление логического значения (см. 7.5).

8.21.2.4 Если присутствует атрибут **var2**, то его значением должно быть имя существующей переменной, а значением переменной должно быть корректное представление логического значения (см. 7.5).

8.21.2.5 Если присутствуют оба атрибута **var1** и **var2**, то операндам присваиваются значения двух переменных, имена которых определены значениями атрибутов **var1** и **var2**. В противном случае операндам присваиваются значения, полученные из текстового содержимого дочерних элементов элемента **<xor>**.

8.21.2.6 Каждый операнд должен быть корректным представлением логического значения (см. 7.5). В результате проверки условия должно быть получено значение **"true"** в том случае, если два операнда имеют различное значение, в противном случае должно быть получено значение **"false"**.

8.22 Элемент <not>

8.22.1 Синтаксис

8.22.1.1 Данный элемент должен иметь следующий атрибут:

- **var** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2).

8.22.1.2 Содержимое этого элемента должно включать в себя те же дочерние элементы, которые содержит элемент **<only_if>** (см. 8.16), но с ограничением, указанным в 8.22.1.3.

8.22.1.3 Если присутствует атрибут **var**, то элемент **<not>** не должен иметь текстового содержимого и дочерних элементов. В противном случае элемент **<not>** должен иметь ровно один дочерний элемент.

8.22.2 Семантика

8.22.2.1 Этот элемент представляет собой условие, основанное на одной переменной или на сочетании переменных, значений, логических операторов, числовых операторов и иных операторов. Проверка условия осуществляется во время обработки родительского элемента.

8.22.2.2 Этот элемент представляет собой логический оператор NOT («НЕ»), операнд которого может быть предоставлен в качестве атрибута этого элемента или в качестве дочернего элемента. Число операндов должно быть равно одному.

8.22.2.3 Если атрибут **var** присутствует, то его значением должно быть имя существующей переменной, а значением переменной должно быть корректное представление логического значения (см. 7.5).

8.22.2.4 Если присутствует атрибут **var**, то операнду присваивается значение переменной, имя которой определено значением атрибута **var**. В противном случае операнду присваивается значение, полученное из текстового содержимого дочернего элемента для элемента **<not>**.

8.22.2.5 Операнд должен быть корректным представлением логического значения (см. 7.5). В результате проверки условия должно быть получено значение **"false"**, если операнд имеет значение **"true"**, в противном случае должно быть получено значение **"true"**.

8.23 Элемент <equal_to>

8.23.1 Синтаксис

8.23.1.1 Данный элемент должен иметь следующие атрибуты:

a) **var1** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2);

b) **var2** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2);

c) **value1** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное представление целого числа;

d) **value2** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное представление целого числа.

8.23.1.2 Содержимое этого элемента должно включать в себя те же дочерние элементы, которые содержит элемент **<only_if>** (см. 8.16), но с ограничением, указанным в 8.23.1.3.

8.23.1.3 Должен присутствовать только один из атрибутов **var1** или **value1**.

8.23.1.4 Должен присутствовать только один из атрибутов **var2** или **value2**.

8.23.1.5 Если присутствуют атрибуты **var1** или **value1** и атрибуты **var2** или **value2**, то элемент **<equal_to>** не должен иметь текстового содержимого и дочерних элементов. Если не присутствует ни один из атрибутов, то должно быть два дочерних элемента, в противном случае должен быть только один дочерний элемент.

8.23.2 Семантика

8.23.2.1 Этот элемент представляет собой условие, основанное на двух переменных или на сочетании переменных, значений, логических операторов, числовых операторов и иных операторов. Проверка условия осуществляется во время обработки родительского элемента.

8.23.2.2 Этот элемент представляет собой числовую операцию "equals" («равно»), чьи операнды могут быть предоставлены в качестве атрибутов этого элемента или в качестве дочерних элементов. Число операндов должно быть равно двум.

8.23.2.3 Если атрибут **var1** присутствует, то его значением должно быть имя существующей переменной, а значением переменной должно быть корректное представление целого числа (см. 7.4).

8.23.2.4 Если атрибут **var2** присутствует, то его значением должно быть имя существующей переменной, а значением переменной должно быть корректное представление целого числа (см. 7.4).

8.23.2.5 Если атрибут **var1** присутствует, то первому операнду присваивается значение переменной, имя которой определено значением атрибута **var1**. Если атрибут **value1** присутствует, то первому операнду присваивается значение этого атрибута. В противном случае первому операнду присваивается значение, полученное из текстового содержимого первого (или единственного) дочернего элемента.

8.23.2.6 Если атрибут **var2** присутствует, то второму операнду присваивается значение переменной, имя которой определено значением атрибута **var2**. Если атрибут **value2** присутствует, то второму операнду присваивается значение этого атрибута. В противном случае второму операнду присваивается значение, полученное из текстового содержимого второго (или единственного) дочернего элемента.

8.23.2.7 Каждый операнд должен быть корректным представлением целого числа (см. 7.4). В результате проверки условия должно быть получено значение **"true"** в том случае, если целое число, представленное первым операндом, равно целому числу, представленному вторым операндом, в противном случае должно быть получено значение **"false"**.

8.24 Элемент <not_equal_to>

8.24.1 Синтаксис

Синтаксис данного элемента аналогичен синтаксису элемента **<equal_to>** (см. 8.23).

8.24.2 Семантика

8.24.2.1 Семантика данного элемента аналогична семантике элемента **<equal_to>**, за исключением следующего.

8.24.2.2 Этот элемент представляет собой числовую операцию «not equal to» («не равно»). В результате проверки условия должно быть получено значение **"true"** в том случае, если целое число, представленное первым операндом, не равно целому числу, представленному вторым операндом, в противном случае должно быть получено значение **"false"**.

8.25 Элемент <greater_than>

8.25.1 Синтаксис

Синтаксис данного элемента аналогичен синтаксису элемента **<equal_to>** (см. 8.23).

8.25.2 Семантика

8.25.2.1 Семантика данного элемента аналогична семантике элемента **<equal_to>**, за исключением следующего.

8.25.2.2 Этот элемент представляет собой числовую операцию **"greater than"** («больше»). В результате проверки условия должно быть получено значение **"true"**, если целое число, представленное первым операндом, больше целого числа, представленного вторым операндом, в противном случае должно быть получено значение **"false"**.

8.26 Элемент <greater_than_or_equal_to>

8.26.1 Синтаксис

Синтаксис данного элемента аналогичен синтаксису элемента **<equal_to>** (см. 8.23).

8.26.2 Семантика

8.26.2.1 Семантика данного элемента аналогична семантике элемента **<equal_to>**, за исключением следующего.

8.26.2.2 Этот элемент представляет собой числовую операцию "greater than or equal to" («больше или равно»). В результате проверки условия должно быть получено значение **"true"** в том случае, если целое число, представленное первым операндом, больше или равно целому числу, представленному вторым операндом, в противном случае должно быть получено значение **"false"**.

8.27 Элемент <less_than>

8.27.1 Синтаксис

Синтаксис данного элемента аналогичен синтаксису элемента **<equal_to>** (см. 8.23).

8.27.2 Семантика

8.27.2.1 Семантика данного элемента аналогична семантике элемента **<equal_to>**, за исключением следующего.

8.27.2.2 Этот элемент представляет собой числовую операцию **"less than"** («меньше»). В результате проверки условия должно быть получено значение **"true"** в том случае, если целое число, представленное первым операндом, меньше целого числа, представленного вторым операндом, в противном случае должно быть получено значение **"false"**.

8.28 Элемент <less_than_or_equal_to>

8.28.1 Синтаксис

Синтаксис данного элемента аналогичен синтаксису элемента **<equal_to>** (см. 8.23).

8.28.2 Семантика

8.28.2.1 Семантика данного элемента аналогична семантике элемента **<equal_to>**, за исключением следующего.

8.28.2.2 Этот элемент представляет собой числовую операцию "less than or equal to" («меньше или равно»). В результате проверки условия должно быть получено значение **"true"**, если целое число, представленное первым операндом, меньше или равно целому числу, представленному вторым операндом, в противном случае должно быть получено значение **"false"**.

8.29 Элемент <same_as>

8.29.1 Синтаксис

8.29.1.1 Данный элемент должен иметь следующие атрибуты:

a) **var1** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2);

b) **var2** (необязательный атрибут) — если этот атрибут присутствует, то его значением должно быть корректное имя переменной (см. 7.2);

c) **value1** (необязательный атрибут);

d) **value2** (необязательный атрибут).

8.29.1.2 Этот элемент должен включать в себя те же дочерние элементы, что и элемент **<only_if>** (см. 8.16), за исключением следующего.

8.29.1.3 Должен присутствовать только один из атрибутов **var1** или **value1**.

8.29.1.4 Должен присутствовать только один из атрибутов **var2** или **value2**.

8.29.1.5 Если присутствуют атрибуты **var1** или **value1** и атрибуты **var2** или **value2**, то элемент **<same_as>** не должен иметь текстового содержимого и дочерних элементов. Если не присутствует ни один из атрибутов, то должно быть два дочерних элемента. В противном случае должен быть один дочерний элемент.

8.29.2 Семантика

8.29.2.1 Этот элемент представляет собой условие, основанное на двух переменных или на сочетании переменных, значений, логических операторов, числовых операторов и иных операторов. Проверка условия осуществляется во время обработки родительского элемента.

8.29.2.2 Этот элемент представляет собой строковую операцию "equals" («соответствует»), чьи операнды могут быть предоставлены в качестве атрибутов этого элемента или в качестве дочерних элементов. Число операндов должно быть равно двум.

8.29.2.3 Если атрибут **var1** присутствует, то его значением должно быть имя существующей переменной.

8.29.2.4 Если атрибут **var2** присутствует, то его значением должно быть имя существующей переменной.

8.29.2.5 Если атрибут **var1** присутствует, то первому операнду присваивается значение переменной, имя которой определено значением атрибута **var1**. Если атрибут **value1** присутствует, то первому операнду присваивается значение этого атрибута. В противном случае первому операнду присваивается значение, полученное из текстового содержимого первого (или единственного) дочернего элемента.

8.29.2.6 Если атрибут **var2** присутствует, то второму операнду присваивается значение переменной, имя которой определено значением атрибута **var2**. Если атрибут **value2** присутствует, то второму операнду присваивается значение этого атрибута. В противном случае второму операнду присваивается значение, полученное из текстового содержимого второго (или единственного) дочернего элемента.

8.29.2.7 В результате проверки условия должно быть получено значение **"true"** в том случае, если строка символов, содержащаяся в первом операнде, совпадает со строкой символов, содержащейся во втором операнде, в противном случае должно быть получено значение **"false"**.

8.30 Элемент <not_same_as>

8.30.1 Синтаксис

Синтаксис данного элемента аналогичен синтаксису элемента **<same_as>** (см. 8.29).

8.30.2 Семантика

8.30.2.1 Семантика данного элемента аналогична семантике элемента **<same_as>**, за исключением следующего.

8.30.2.2 Этот элемент представляет собой строковую операцию "not equal to" («не соответствует»). В результате проверки условия должно быть получено значение **"true"** в том случае, если строка символов, содержащаяся в первом операнде, не совпадает со строкой символов, содержащейся во втором операнде, в противном случае должно быть получено значение **"false"**.

8.31 Элемент <existing>

8.31.1 Синтаксис

8.31.1.1 Данный элемент должен иметь следующий атрибут:

- var (обязательный атрибут) — значением этого атрибута должно быть корректное имя переменной (см. 7.2).

8.31.1.2 Данный элемент не должен иметь текстового содержимого и дочерних элементов.

8.31.2 Семантика

8.31.2.1 Этот элемент представляет собой условие, основанное на переменной. Проверка условия осуществляется во время обработки родительского элемента.

8.31.2.2 Если значением атрибута **var** является имя существующей глобальной или локальной переменной, то в результате проверки условия должно быть получено значение **"true"**, в противном случае должно быть получено значение **"false"**.

8.32 Элемент <not_existing>

8.32.1 Синтаксис

Синтаксис данного элемента аналогичен синтаксису элемента **<existing>** (см. 8.31).

8.32.2 Семантика

8.32.2.1 Семантика данного элемента аналогична семантике элемента **<existing>**, за исключением следующего.

8.32.2.2 Если значением атрибута **var** является имя существующей глобальной или локальной переменной, то в результате проверки должно быть получено значение **"false"**, в противном случае должно быть получено значение **"true"**.

9 Функции стандартного интерфейса

9.1 Общие положения

9.1.1 В данном разделе установлены требования к использованию функций BioAPI стандартного интерфейса BioAPI при проведении испытания на соответствие. В 9.2 установлены группы параметров с особыми ограничениями, которые необходимо учитывать в дополнение к ограничениям параметров, определенных в последующих подразделах. Каждый подраздел настоящего раздела (начиная с подраздела 9.3 и до конца раздела) устанавливает требования к следующим свойствам функций стандартного интерфейса BioAPI:

1) схеме вызова функции — наименования функции стандартного интерфейса BioAPI, а также наименования входных и выходных параметров, которые необходимо использовать для вызова данной функции;

2) ограничениям на параметры;

3) входным параметрам вызова функции — установке собственных входных параметров базовой функции на основе входных параметров вызова функции;

4) выходным данным вызова функции — установке выходных параметров и возвращаемого значения вызова функции на основе собственных выходных параметров и возвращаемого значения базовой функции;

5) входным данным вызова связанного процесса — настройке входных параметров процесса, связанного с функцией стандартного интерфейса BioAPI на основе собственных входных параметров входящего вызова;

6) выходным данным вызова связанного процесса — установке собственных выходных параметров и возвращаемого значения входящего вызова на основе выходных параметров процесса, связанного с функцией стандартного интерфейса BioAPI;

7) выходным данным по умолчанию — установке собственных выходных параметров и возвращаемого значения входящего вызова в случае отсутствия процесса, связанного с функцией стандартного интерфейса BioAPI.

Примечание — Некоторые переменные и группы параметров языка утверждения поддерживают большее количество значений, чем те, которые BioAPI разрешено поддерживать соответствующим собственным параметрам переменных и функций с целью создания утверждений, которые могут оценить ответ TP значениям, которые находятся за рамками разрешенных, и оценивать правильность и пригодность ответов TP.

9.1.2 «Исходящим значением» (для входного параметра функции) является:

a) значение атрибута **value**, если соответствующий элемент **<input>** содержит атрибут **value**;

b) значение переменной, чье имя является значением атрибута **var**, если соответствующий элемент **<input>** содержит атрибут **var**.

9.1.3 «Входящим значением» (для выходного параметра функции) является значение, которое должно быть присвоено переменной, чье имя является значением атрибута **setvar** соответствующего элемента **<output>** (если этот элемент представлен в вызове).

9.1.4 «Возвращаемым значением» функции является значение, которое должно быть присвоено переменной, чье имя является значением атрибута **setvar** элемента **<return>** (если этот элемент представлен в вызове).

9.1.5 «Входящим значением» (для входного параметра связанного процесса) является:

a) значение атрибута **value**, если соответствующий элемент **<input>** содержит атрибут **value**;

b) значение переменной, чье имя является значением атрибута **var**, если соответствующий элемент **<input>** содержит атрибут **var**.

9.1.6 «Исходящим значением» (для выходного параметра связанного процесса) является значение, которое должно быть присвоено переменной, чье имя является значением атрибута **setvar** соответствующего элемента **<output>** (если этот элемент присутствует в вызове).

9.1.7 Во всех вызовах функции возвращаемое (входящее) значение должно быть приведено к стандартному представлению целого числа в интервале от 0 до 4294967295, возвращенному базовой функцией (см. 7.4.3).

9.1.8 В каждой модели испытаний на соответствие функции стандартного интерфейса BioAPI должны применяться в соответствии с таблицей 1.

Таблица 1 — Применение функций стандартного интерфейса BioAPI в разных моделях испытания на соответствие

	Модель испытания на соответствие приложений BioAPI (см. 6.2.5.1)	Модель испытания на соответствие инфраструктур BioAPI (см. 6.2.5.2)	Модель испытания на соответствие ПБУ BioAPI (см. 6.2.5.3)
Функции интерфейса BioAPI	Инфраструктура для испытания приложения должна объявлять функцию. Обычная инфраструктура объявляет функцию.	Испытуемая инфраструктура объявляет функцию. Ни приложение, ни ПБУ, испытывающие инфраструктуру, не должны объявлять функцию.	Приложение, испытывающее ПБУ, не должно объявлять функцию. Ни один способ вызова функции не поддерживается

Окончание таблицы 1

	Модель испытания на соответствие приложений BioAPI (см. 6.2.5.1)	Модель испытания на соответствие инфраструктур BioAPI (см. 6.2.5.2)	Модель испытания на соответствие ПБУ BioAPI (см. 6.2.5.3)
Функции интерфейса BioAPI	Испытуемое приложение может вызывать функцию, объявленную инфраструктурой для испытания приложения. Любой входящий вызов в адрес функции, объявленной инфраструктурой для испытания приложения, должен быть отправлен аналогичной функции, объявленной обычной инфраструктурой. Ни один другой способ вызова функции не поддерживается	Приложение, испытывающее инфраструктуру, может вызвать функцию, объявленную испытуемой инфраструктурой. Ни один другой способ вызова функции не поддерживается	
Функции интерфейса обратного вызова приложения	Испытуемое приложение может объявить функцию. Инфраструктура, испытывающая приложение, должна объявлять функцию. Обычная инфраструктура может вызывать функцию, объявленную инфраструктурой, испытывающей приложение. Любой входящий вызов в адрес функции,объявленной инфраструктурой, испытывающей приложение, должен быть отправлен аналогичной функции, объявленной испытуемым приложением (при его наличии). Ни один другой способ вызова функции не поддерживается	Приложение, испытывающее инфраструктуру, должно объявлять функцию. Испытуемая инфраструктура может вызывать функцию, объявленную приложением, испытывающим инфраструктуру. Ни один другой способ вызова функции не поддерживается	Приложение, испытывающее ПБУ, не должно объявлять функцию. Ни один способ вызова функции не поддерживается
Интерфейс обратного вызова инфраструктуры	Обычная инфраструктура объявляет функцию. Инфраструктура, испытывающая приложение, не должна объявлять функцию. Обычный ПБУ может вызывать функцию, объявленную обычной инфраструктурой. Ни один другой способ вызова функции не поддерживается	Испытуемая инфраструктура объявляет функцию. Ни приложение, ни ПБУ, испытывающие инфраструктуру, не должны объявлять функцию. ПБУ, тестирующий инфраструктуру, может вызывать функцию, объявленную испытуемой инфраструктурой. Ни один другой способ вызова функции не поддерживается	Приложение, испытывающее ПБУ, должно объявлять функцию. Испытуемый ПБУ может вызывать функцию, объявленную приложением, испытывающим ПБУ. Ни один другой способ вызова функции не поддерживается
Функции интерфейса BioSPI	Обычный ПБУ объявляет функцию. Инфраструктура, испытывающая приложение, не должна объявлять функцию. Обычная инфраструктура может вызывать функцию, объявленную обычным ПБУ. Ни один другой способ вызова функции не поддерживается	ПБУ, испытывающий инфраструктуру, должен объявлять функцию. Приложение, испытывающее инфраструктуру, не должно объявлять функцию. Тестируемая инфраструктура может вызывать функцию, объявленную ПБУ, испытывающим инфраструктуру. Ни один другой способ вызова функции не поддерживается	Испытуемый ПБУ может объявлять функцию. Приложение, испытывающее ПБУ, может вызывать функцию, объявленную испытуемым ПБУ. Ни один другой способ вызова функции не поддерживается

9.1.9 Интерфейс BioAPI включает в себя следующие стандартные функции:

- **BioAPI_Init;**
- **BioAPI_Terminate;**
- **BioAPI_GetFrameworkInfo;**
- **BioAPI_EnumBSPs;**
- **BioAPI_BSPLoad;**
- **BioAPI_BSPUnload;**
- **BioAPI_BSPAttach;**
- **BioAPI_BSPDetach;**
- **BioAPI_QueryUnits;**
- **BioAPI_EnumBFPs;**
- **BioAPI_QueryBFPs;**
- **BioAPI_ControlUnit;**
- **BioAPI_FreeBIRHandle;**
- **BioAPI_GetBIRFromHandle;**
- **BioAPI_GetHeaderFromHandle;**
- **BioAPI_EnableEvents;**
- **BioAPI_SetGUICallbacks;**
- **BioAPI_Capture;**
- **BioAPI_CreateTemplate;**
- **BioAPI_Process;**
- **BioAPI_ProcessWithAuxBIR;**
- **BioAPI_VerifyMatch;**
- **BioAPI_IdentifyMatch;**
- **BioAPI_Enroll;**
- **BioAPI_Verify;**
- **BioAPI_Identify;**
- **BioAPI_Import;**
- **BioAPI_PresetIdentifyPopulation;**
- **BioAPI_DbOpen;**
- **BioAPI_DbClose;**
- **BioAPI_DbCreate;**
- **BioAPI_DbDelete;**
- **BioAPI_DbSetMarker;**
- **BioAPI_DbFreeMarker;**
- **BioAPI_DbStoreBIR;**
- **BioAPI_DbGetBIR;**
- **BioAPI_DbGetNextBIR;**
- **BioAPI_DbDeleteBIR;**
- **BioAPI_SetPowerMode;**
- **BioAPI_SetIndicatorStatus;**
- **BioAPI_GetIndicatorStatus;**
- **BioAPI_CalibrateSensor;**
- **BioAPI_Cancel;**
- **BioAPI_Free;**
- **BioAPI_Util_InstallBSP;**
- **BioAPI_Util_InstallBFP.**

9.1.10 Интерфейс BioSPI включает в себя следующие стандартные функции интерфейса BioAPI:

- **BioSPI_BSPLoad;**
- **BioSPI_BSPUnload;**
- **BioSPI_BSPAttach;**
- **BioSPI_BSPDetach;**
- **BioSPI_QueryUnits;**
- **BioSPI_QueryBFPs;**
- **BioSPI_ControlUnit;**
- **BioSPI_FreeBIRHandle;**

- **BioSPI_GetBIRFromHandle;**
- **BioSPI_GetHeaderFromHandle;**
- **BioSPI_EnableEvents;**
- **BioSPI_SetGUICallbacks;**
- **BioSPI_Capture;**
- **BioSPI_CreateTemplate;**
- **BioSPI_Process;**
- **BioSPI_ProcessWithAuxBIR;**
- **BioSPI_VerifyMatch;**
- **BioSPI_IdentifyMatch;**
- **BioSPI_Enroll;**
- **BioSPI_Verify;**
- **BioSPI_Identify;**
- **BioSPI_Import;**
- **BioSPI_PresetIdentifyPopulation;**
- **BioSPI_DbOpen;**
- **BioSPI_DbClose;**
- **BioSPI_DbCreate;**
- **BioSPI_DbDelete;**
- **BioSPI_DbSetMarker;**
- **BioSPI_DbFreeMarker;**
- **BioSPI_DbStoreBIR;**
- **BioSPI_DbGetBIR;**
- **BioSPI_DbGetNextBIR;**
- **BioSPI_DbDeleteBIR;**
- **BioSPI_SetPowerMode;**
- **BioSPI_SetIndicatorStatus;**
- **BioSPI_GetIndicatorStatus;**
- **BioSPI_CalibrateSensor;**
- **BioSPI_Cancel;**
- **BioSPI_Free.**

9.1.11 Интерфейс обратного вызова приложения включает в себя следующие функции стандартного интерфейса BioAPI:

- **BioAPI_EventHandler;**
- **BioAPI_GUI_STATE_CALLBACK;**
- **BioAPI_GUI_STREAMING_CALLBACK.**

9.1.12 Интерфейс обратного вызова инфраструктуры включает в себя следующие функции стандартного интерфейса BioAPI:

- **BioSPI_EventHandler;**
- **BioSPI_GUI_STATE_CALLBACK;**
- **BioSPI_GUI_STREAMING_CALLBACK;**
- **BioSPI_BFP_ENUMERATION_HANDLER;**
- **BioSPI_MEMORY_FREE_HANDLER.**

Примечание — В ИСО/МЭК 19784-1 функции стандартного интерфейса BioAPI обратного вызова инфраструктуры не имеют имен функций, но определены как типы указателей на функцию (которые имеют имя). В настоящем стандарте функциям стандартного интерфейса BioAPI даны имена для того, чтобы было проще на них ссылаться в разделах настоящего стандарта и для применения в языке утверждений.

9.2 Группы параметров

В данном подразделе определены группы параметров (функций или процессов). На каждую группу ссылается одна или несколько следующих групп или один или несколько подразделов настоящего раздела, определяющего стандартные функции интерфейса BioAPI.

9.2.1 Группа параметров «Биометрический тип»

9.2.1.1 Группа параметров «Биометрический тип» включает в себя следующие параметры:

- **TypeMultiple;**
- **TypeFacialFeatures;**

- **TypeVoice;**
- **TypeFingerprint;**
- **TypeIris;**
- **TypeRetina;**
- **TypeHandGeometry;**
- **TypeSignatureDynamics;**
- **TypeKeystrokeDynamics;**
- **TypeLipMovement;**
- **TypeThermalFaceImage;**
- **TypeThermalHandImage;**
- **TypeGait;**
- **TypeOther;**
- **TypePassword.**

9.2.1.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_BIR_BIOMETRIC_TYPE** (который является целочисленным типом).

9.2.1.3 Исходящим значением каждого из вышеприведенных параметров должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.2.1.4 Значение (например, **V**) собственного типа **BioAPI_BIR_BIOMETRIC_TYPE**, представленное исходящим значением этой группы параметров, должно определяться следующим образом:

1) сначала должно быть установлено значение **V**, равное нулю;

2) для каждого элемента группы параметров, исходящее значение которого соответствует значению **"true"**, к значению **V** должно быть прибавлено целое число в соответствии с таблицей 2.

Таблица 2

TypeMultiple	$1 = 2^0$
TypeFacialFeatures	$2 = 2^1$
TypeVoice	$4 = 2^2$
TypeFingerprint	$8 = 2^3$
TypeIris	$16 = 2^4$
TypeRetina	$32 = 2^5$
TypeHandGeometry	$64 = 2^6$
TypeSignatureDynamics	$128 = 2^7$
TypeKeystrokeDynamics	$256 = 2^8$
TypeLipMovement	$512 = 2^9$
TypeThermalFaceImage	$1024 = 2^{10}$
TypeThermalHandImage	$2048 = 2^{11}$
TypeGait	$4096 = 2^{12}$
TypeOther	$1073741824 = 2^{30}$
TypePassword	$2147483648 = 2^{31}$

Примечание — Если параметр **TypeMultiple** имеет значение **"false"**, то только один из параметров, соответствующих другим типам, может иметь значение **"true"**.

9.2.1.5 Если задано целое число (например, **V**), которое является значением собственного типа **BioAPI_BIR_BIOMETRIC_TYPE**, то входящее значение этой группы параметров, которое канонически представляет **V**, определяют следующим образом:

1) целое число **V** должно быть разбито на сумму показателей степеней числа 2, каждый из которых присутствует в показателях не более одного раза;

2) для каждого показателя степени числа 2, указанного в вышеприведенной таблице, который появляется в разбиении целого числа **V**, входящее значение параметра в соответствующей строке таблицы должно быть **"true"**;

3) входящим значением всех остальных параметров (в случае их наличия) должно быть **"false"**;

4) четное значение **V**, не являющееся единственным показателем степени числа 2, указывает на некорректное значение.

9.2.2 Группа параметров «Операции»

9.2.2.1 Группа параметров «Операции» включает в себя следующие параметры:

- **OperationEnableEvents;**
- **OperationSetGUICallbacks;**
- **OperationCapture;**
- **OperationCreateTemplate;**
- **OperationProcess;**
- **OperationProcessWithAuxBIR;**
- **OperationVerifyMatch;**
- **OperationIdentifyMatch;**
- **OperationEnroll;**
- **OperationVerify;**
- **OperationIdentify;**
- **OperationImport;**
- **OperationPresetIdentifyPopulation;**
- **OperationDatabaseOperations;**
- **OperationSetPowerMode;**
- **OperationSetIndicatorStatus;**
- **OperationGetIndicatorStatus;**
- **OperationCalibrateSensor;**
- **OperationUtilities;**
- **OperationQueryUnits;**
- **OperationQueryBFPs.**

9.2.2.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_OPERATIONS_MASK** (который является целочисленным типом).

9.2.2.3 Исходящим значением каждого из вышеприведенных параметров должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.2.2.4 Значение (например, **V**) собственного типа **BioAPI_OPERATIONS_MASK**, представленное исходящим значением этой группы параметров, определяют следующим образом:

1) сначала должно быть установлено значение **V**, равное нулю;

2) для каждого элемента группы параметров, исходящее значение которого соответствует значению **"true"**, к значению **V** должно быть прибавлено целое число в соответствии с таблицей 3.

Таблица 3

OperationEnableEvents	$1 = 2^0$
OperationSetGUICallbacks	$2 = 2^1$
OperationCapture	$4 = 2^2$
OperationCreateTemplate	$8 = 2^3$
OperationProcess	$16 = 2^4$
OperationProcessWithAuxBIR	$32 = 2^5$
OperationVerifyMatch	$64 = 2^6$
OperationIdentifyMatch	$128 = 2^7$
OperationEnroll	$256 = 2^8$
OperationVerify	$512 = 2^9$
OperationIdentify	$1024 = 2^{10}$

Окончание таблицы 3

OperationImport	$2048 = 2^{11}$
OperationPresetIdentifyPopulation	$4096 = 2^{12}$
OperationDatabaseOperations	$8192 = 2^{13}$
OperationSetPowerMode	$16384 = 2^{14}$
OperationSetIndicatorStatus	$32768 = 2^{15}$
OperationGetIndicatorStatus	$65536 = 2^{16}$
OperationCalibrateSensor	$131072 = 2^{17}$
OperationUtilities	$262144 = 2^{18}$
OperationQueryUnits	$1048576 = 2^{20}$
OperationQueryBFPs	$2097152 = 2^{21}$
OperationControlUnit	$4194304 = 2^{22}$

9.2.2.5 Если задано целое число (например, **V**), которое является значением собственного типа **BioAPI_OPERATIONS_MASK**, то входящее значение этой группы параметров, которое канонически представляет **V**, определяют следующим образом:

1) целое число **V** должно быть разбито на сумму показателей степеней числа 2, каждый из которых присутствует в показателях не более одного раза;

2) для каждого показателя степени числа 2, указанного в вышеприведенной таблице, который появляется в разбиении целого числа **V**, входящее значение параметра в соответствующей строке таблицы должно соответствовать значению **"true"**;

3) входящим значением всех остальных параметров (в случае их наличия) должно быть **"false"**.

9.2.3 Группа параметров «Опции»

9.2.3.1 Группа параметров «Опции» включает в себя следующие параметры:

- **OptionRaw;**
- **OptionQualityRaw;**
- **OptionQualityIntermediate;**
- **OptionQualityProcessed;**
- **OptionAppGUI;**
- **OptionSourcePresent;**
- **OptionPayload;**
- **OptionBIRSign;**
- **OptionBIREncrypt;**
- **OptionTemplateUpdate;**
- **OptionAdaptation;**
- **OptionBinning;**
- **OptionSelfContainedDevice;**
- **OptionMOC;**
- **OptionSubtypeToCapture;**
- **OptionSensorBFP;**
- **OptionArchiveBFP;**
- **OptionMatchingBFP;**
- **OptionProcessingBFP;**
- **OptionCoarseScores.**

9.2.3.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_OPTIONS_MASK** (который является целочисленным типом).

9.2.3.3 Исходящим значением каждого из вышеприведенных параметров должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.2.3.4 Значение (например, **V**) собственного типа **BioAPI_OPTIONS_MASK**, представленное исходящим значением этой группы параметров, определяют следующим образом:

1) сначала должно быть установлено значение **V**, равное нулю;

2) для каждого элемента группы параметров, исходящее значение которого соответствует значению **"true"**, к значению **V** должно быть прибавлено целое число в соответствии с таблицей 4.

Таблица 4

OptionRaw	$1 = 2^0$
OptionQualityRaw	$2 = 2^1$
OptionQualityIntermediate	$4 = 2^2$
OptionQualityProcessed	$8 = 2^3$
OptionAppGUI	$16 = 2^4$
OptionSourcePresent	$64 = 2^6$
OptionPayload	$128 = 2^7$
OptionBIRSign	$256 = 2^8$
OptionBIREncrypt	$512 = 2^9$
OptionTemplateUpdate	$1024 = 2^{10}$
OptionAdaptation	$2048 = 2^{11}$
OptionBinning	$4096 = 2^{12}$
OptionSelfContainedDevice	$8192 = 2^{13}$
OptionMOC	$16384 = 2^{14}$
OptionSubtypeToCapture	$32768 = 2^{15}$
OptionSensorBFP	$65536 = 2^{16}$
OptionArchiveBFP	$131072 = 2^{17}$
OptionMatchingBFP	$262144 = 2^{18}$
OptionProcessingBFP	$524288 = 2^{19}$
OptionCoarseScores	$1048576 = 2^{20}$

9.2.3.5 Если задано целое число (например, **V**), которое является значением собственного типа **BioAPI_OPTIONS_MASK**, то входящее значение этой группы параметров, которое канонически представляет **V**, определяют следующим образом:

1) целое число **V** должно быть разбито на сумму показателей степеней числа 2, каждый из которых присутствует в показателях не более одного раза;

2) для каждого показателя степени числа 2, указанного в вышеприведенной таблице, который появляется в разбиении целого числа **V**, входящее значение параметра в соответствующей строке таблицы должно быть **"true"**;

3) входящим значением всех остальных параметров (в случае их наличия) должно быть **"false"**.

9.2.4 Группа параметров «События»

9.2.4.1 Группа параметров «События» включает в себя следующие параметры:

- **EventNotifyInsert;**
- **EventNotifyRemove;**
- **EventNotifyFault;**
- **EventNotifySourcePresent;**
- **EventNotifySourceRemoved.**

9.2.4.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_EVENT_MASK** (который является целочисленным типом).

9.2.4.3 Исходящим значением каждого из вышеприведенных параметров должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.2.4.4 Значение (например, **V**) собственного типа **BioAPI_EVENT_MASK**, представленное исходящим значением этой группы параметров, определяют следующим образом:

1) сначала должно быть установлено значение **V**, равное нулю;

2) для каждого элемента группы параметров, исходящее значение которого соответствует значению **"true"**, к значению **V** должно быть прибавлено целое число в соответствии с таблицей 5.

Таблица 5

EventNotifyInsert	$1 = 2^0$
EventNotifyRemove	$2 = 2^1$
EventNotifyFault	$4 = 2^2$
EventNotifySourcePresent	$8 = 2^3$
EventNotifySourceRemoved	$16 = 2^4$

9.2.4.5 Если задано целое число (например, **V**), которое является значением собственного типа **BioAPI_EVENT_MASK**, то входящее значение этой группы параметров, которое канонически представляет значение **V**, определяют следующим образом:

1) целое число **V** должно быть разбито на сумму показателей степеней числа 2, каждый из которых присутствует в показателях не более одного раза;

2) для каждого показателя степени числа 2, указанного в вышеприведенной таблице, который появляется в разбиении целого числа **V**, входящее значение параметра в соответствующей строке таблицы должно быть **"true"**;

3) входящим значением всех остальных параметров (в случае их наличия) должно быть **"false"**.

9.2.5 Группа параметров «Тип биометрических данных»

9.2.5.1 Группа параметров «Тип биометрических данных» включает в себя следующие параметры:

- **ProcessedLevel;**
- **Encrypted;**
- **Signed;**
- **IndexPresent.**

9.2.5.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_BIR_DATA_MASK** (который является целочисленным типом).

9.2.5.3 У элементов данной группы параметров должны быть входящие и исходящие значения, соответствующие приведенным в 9.2.5.3.1 и 9.2.5.3.2.

9.2.5.3.1 Исходящим значением **ProcessedLevel** должно быть или корректно представленное целое число (см. 7.4) в интервале от 0 до 15, или пустая строка.

Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.2.5.3.2 Исходящими значениями **Encrypted, Signed** и **IndexPresent** должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.2.5.4 Значение (например, **V**) собственного типа **BioAPI_BIR_DATA_TYPE**, представленное исходящим значением этой группы параметров, определяют следующим образом:

1) сначала должно быть установлено значение **V**, равное целому числу, представленному исходящим значением **ProcessedLevel** (или равное нулю, если исходящее значение является пустой строкой);

2) для каждого элемента группы параметров (кроме **ProcessedLevel**), исходящее значение которого соответствует значению **"true"**, к значению **V** должно быть прибавлено целое число в соответствии с таблицей 6.

Таблица 6

Encrypted	$16 = 2^4$
Signed	$32 = 2^5$
IndexPresent	$128 = 2^7$

9.2.5.5 Если задано целое число (например, **V**), которое является значением собственного типа **BioAPI_BIR_DATA_TYPE**, входящее значение этой группы параметров, которое канонически представляет значение **V**, определяют следующим образом:

1) целое число **V** должно быть разбито на сумму:

i) целого числа меньше 16;

ii) значения 16;

iii) значения 32 и

iv) значения 128,

каждое из которых присутствует в сумме показателей не более одного раза;

2) для каждого показателя степени числа 2, указанного в вышеприведенной таблице (16, 32 и 128), который появляется в разбиении целого числа **V**, входящим значением параметра в соответствующей строке таблицы должно быть **"true"**;

3) входящим значением **ProcessedLevel** должно быть стандартно представленное целое число, указанное в 1) перечисление i) (см. 7.4.3);

4) входящим значением всех остальных параметров (в случае их наличия) должно быть **"false"**.

9.2.6 Группа параметров «Биометрический подтип»

9.2.6.1 Группа параметров «Биометрический подтип» включает в себя следующие параметры:

- **Left (левый);**
- **Right (правый);**
- **Thumb (большой палец);**
- **PointerFinger (указательный палец);**
- **MiddleFinger (средний палец);**
- **RingFinger (безымянный палец);**
- **LittleFinger (мизинец);**
- **Multiple.**

9.2.6.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_BIR_SUBTYPE** (который является целочисленным типом).

9.2.6.3 Исходящим значением каждого из вышеуказанных параметров должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.2.6.4 Значение (например, **V**) собственного типа **BioAPI_BIR_SUBTYPE**, представленное исходящим значением этой группы параметров, определяют следующим образом:

1) первоначально значение **V** должно быть установлено равным нулю;

2) для каждого элемента группы параметров, исходящее значение которого соответствует значению **"true"**, к значению **V** должно быть прибавлено целое число в соответствии с таблицей 7.

Таблица 7

Left	$1 = 2^0$
Right	$2 = 2^1$
Thumb	$4 = 2^2$
PointerFinger	$8 = 2^3$
MiddleFinger	$16 = 2^4$
RingFinger	$32 = 2^5$
LittleFinger	$64 = 2^6$
Multiple	$128 = 2^7$

9.2.6.5 Если задано целое число (например, **V**), которое является значением собственного типа **BioAPI_BIR_SUBTYPE**, то входящее значение этой группы параметров, которое канонически представляет значение **V**, определяют следующим образом:

1) целое число **V** должно быть разбито на сумму показателей степеней числа 2, каждый из которых присутствует в сумме показателей не более одного раза;

2) для каждого показателя степени числа 2, указанного в таблице выше, который появляется в разбиении целого числа **V**, входящим значением параметра в соответствующей строке таблицы должно быть **"true"**;

3) входящим значением всех остальных параметров (в случае их наличия) должно быть **"false"**.

9.2.7 Группа параметров «Дата»

9.2.7.1 Группа параметров «Дата» включает в себя следующие параметры:

- Year;

- Month;

- Day.

9.2.7.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_DATE**.

9.2.7.3 У элементов данной группы параметров должны быть входящие и исходящие значения, определенные в соответствии с 9.2.7.3.1 и 9.2.7.3.2.

9.2.7.3.1 Исходящим значением **Year** должно быть или корректно представленное целое число (см. 7.4) в интервале от 0 до 65535, или пустая строка. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.2.7.3.2 Исходящими значениями **Month** и **Day** должны быть или корректно представленные целые числа (см. 7.4) в интервале от 0 до 255, или пустые строки. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале.

9.2.7.4 Значение (например, **V**) собственного типа **BioAPI_DATE**, представленное исходящим значением этой группы параметров, определяют следующим образом.

Целые числа, представленные исходящими значениями **Year**, **Month** и **Day** (или ноль, если исходящими значениями являются пустые строки), должны быть записаны в поля **Year**, **Month** и **Day** (соответственно) поля **Date** значения **V**.

9.2.7.5 Если задано значение (например, **V**), то входящее значение **V** этой группы параметров, которое канонически представляет значение **V**, определяют следующим образом.

Входящими значениями **Year**, **Month** и **Day** должны быть стандартно представленные целые числа (см. 7.4.3) в полях **Year**, **Month** и **Day** (соответственно) поля **Date** значения **V**.

9.2.8 Группа параметров «Дата и время»

9.2.8.1 Группа параметров «Дата и время» включает в себя следующие параметры:

- Year;

- Month;

- Day;

- Hour;

- Minute;

- Second.

9.2.8.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_DTG**.

9.2.8.3 У элементов данной группы параметров должны быть входящие и исходящие значения, соответствующие приведенным в 9.2.8.3.1 и 9.2.8.3.2.

9.2.8.3.1 Исходящим значением **Year** должно быть или корректно представленное целое число (см. 7.4) в интервале от 0 до 65535, или пустая строка. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.2.8.3.2 Исходящими значениями **Month**, **Day**, **Hour**, **Minute** и **Second** должны быть или корректно представленные целые числа (см. 7.4) в интервале от 0 до 255, или пустые строки. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале.

9.2.8.4 Значение (например, **V**) собственного типа **BioAPI_DTG**, представленное исходящим значением этой группы параметров, определяют в соответствии с 9.2.8.4.1 и 9.2.8.4.2.

9.2.8.4.1 Целые числа, представленные исходящими значениями **Year**, **Month** и **Day** (или 0, если исходящими значениями являются пустые строки), должны быть записаны в поля **Year**, **Month** и **Day** (соответственно) поля **Date** значения **V**.

9.2.8.4.2 Целые числа, представленные исходящими значениями **Hour**, **Minute** и **Second** (или 0, если исходящими значениями являются пустые строки), должны быть записаны в поля **Hour**, **Minute** и **Second** (соответственно) поля **Time** значения **V**.

9.2.8.5 Если задано значение (например, **V**), то входящее значение этой группы параметров, которое канонически представляет значение **V**, следует определять в соответствии с 9.2.8.5.1 и 9.2.8.5.2.

9.2.8.5.1 Входящими значениями **Year**, **Month** и **Day** должны быть стандартно представленные целые числа (см. 7.4.3) в полях **Year**, **Month** и **Day** (соответственно) поля **Date** значения **V**.

9.2.8.5.2 Входящими значениями **Hour**, **Minute** и **Second** должны быть стандартно представленные целые числа (см. 7.4.3) в полях **Hour**, **Minute** и **Second** (соответственно) поля **Time** значения **V**.

9.2.9 Группа параметров «Схема инфраструктуры»

9.2.9.1 Группа параметров «Схема структуры» включает в себя следующие параметры:

- **FrameworkUuid;**
- **Description;**
- **Path;**
- **SpecVersion;**
- **ProductVersion;**
- **Vendor;**
- **FwPropertyId;**
- **FwProperty.**

9.2.9.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_FRAMEWORK_SCHEMA**.

9.2.9.3 У элементов данной группы параметров должны быть входящие и исходящие значения, соответствующие приведенным в 9.2.9.3.1—9.2.9.3.5.

9.2.9.3.1 Исходящими значениями **FrameworkUuid** и **Fwpropertyld** должны быть или корректно представленные УУИД (см. 7.6), или пустые строки. Входящими значениями должны быть стандартно представленные УУИД (см. 7.6.3).

9.2.9.3.2 Исходящим значением **SpecVersion** должно быть или корректно представленное целое число в интервале от 0 до 255, или пустая строка. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.2.9.3.3 Исходящие или входящие значения **ProductVersion**, **Vendor** и **Description** должны представлять собой строку символов, кодировка UTF-8 которой должна быть не длиннее 268 байтов, не содержащую символы NUL (0).

9.2.9.3.4 Исходящим значением **FwProperty** должен быть корректно представленный набор байтов (см. 7.7). Входящим значением должен быть стандартно представленный набор байтов (см. 7.7.2).

9.2.9.3.5 Исходящее или входящее значение **Path** должно представлять собой строку символов, не содержащую символов NUL (0).

9.2.9.4 Значение (например, **V**) собственного типа **BioAPI_FRAMEWORK_SCHEMA**, представленное исходящим значением этой группы параметров, следует определять в соответствии с 9.2.9.4.1—9.2.9.4.5.

9.2.9.4.1 УУИД, представленные исходящими значениями **FrameworkUuid** и **Fwpropertyld** (или УУИД «00000000-0000-0000-0000-000000000000», если исходящим значением является пустая строка) должны быть записаны в поле значения **V** с тем же именем.

9.2.9.4.2 Целое число, представленное исходящим значением **SpecVersion** (или 0, если исходящим значением является пустая строка), должно быть записано в поле **SpecVersion** значения **V**.

9.2.9.4.3 Строки символов, являющиеся исходящими значениями **Description**, **ProductVersion** и **Vendor**, должны быть записаны в поле значения **V** с теми же именами, заканчивающимися символом конца строки NUL (0).

9.2.9.4.4 Набор байтов, представленный исходящим значением **FwProperty**, должен быть записан в блок памяти достаточного объема. Адрес и длина блока памяти должны быть записаны в поля **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, адрес которой должен быть записан в поле **FwProperty** значения **V**.

9.2.9.4.5 Строка символов, являющаяся исходящим значением **Path**, должна быть записана с символом NUL (0) в конце в блок памяти достаточного объема, адрес которого должен быть записан в поле **Path** значения **V**.

9.2.9.5 Если задано значение (например, **V**) собственного типа **BioAPI_FRAMEWORK_SCHEMA**, то входящее значение этой группы параметров, которое канонически представляет **V**, следует определять в соответствии с 9.2.9.5.1—9.2.9.5.2.

9.2.9.5.1 Входящими значениями **FrameworkUuid** и **Fwpropertyld** должны быть стандартно представленные (см. 7.6.3) УУИД в поле значения **V** с теми же именами.

9.2.9.5.2 Входящим значением **SpecVersion** должно быть стандартно представленное целое число в поле значения **V** с тем же именем.

9.2.9.5.3 Входящие значения **Description**, **ProductVersion** и **Vendor** должны представлять собой строки символов в поле **V** с теми же именами без символа NUL (0) в конце строки.

9.2.9.5.4 Входящим значением **FwProperty** должен быть стандартно представленный набор байтов в блоке памяти, адрес и длина которого находятся в полях **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, на которую указывает значение, находящееся в поле **FwProperty** значения **V**.

9.2.9.5.5 Входящее значение **Path** должно представлять собой строку символов, на которую указывает поле значения **V** с тем же именем и без символа NUL (0) в конце строки.

9.2.10 Группа параметров «Схема ПБУ»

9.2.10.1 Группа параметров «Схема ПБУ» включает в себя следующие параметры:

- **BSPUuid;**
- **Description;**
- **Path;**
- **SpecVersion;**
- **ProductVersion;**
- **Vendor;**
- **Format_1_FormatOwner;**
- **Format_1_FormatType;**
- **Format_2_FormatOwner;**
- **Format_2_FormatType;**
- **Format_3_FormatOwner;**
- **Format_3_FormatType;**
- **Format_4_FormatOwner;**
- **Format_4_FormatType;**
- **NumSupportedFormats;**
- элементы группы параметров «Биометрический тип» (см. 9.2.1);
- элементы группы параметров «Операции» (см. 9.2.2);
- элементы группы параметров «Опции» (см. 9.2.3);
- **PayloadPolicy;**
- **MaxPayloadSize;**
- **DefaultVerifyTimeout;**
- **DefaultIdentifyTimeout;**
- **DefaultCaptureTimeout;**
- **DefaultEnrollTimeout;**
- **DefaultCalibrateTimeout;**
- **MaxBSPDbSize;**
- **MaxIdentify.**

9.2.10.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_BSP_SCHEMA**.

9.2.10.3 У элементов данной группы параметров должны быть входящие и исходящие значения, соответствующие приведенным в 9.2.10.3.1—9.2.10.3.2.

9.2.10.3.1 Исходящим значением **BSPUuid** должен быть или корректно представленный УУИД (см. 7.6), или пустая строка. Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.2.10.3.2 Исходящие или входящие значения **Description**, **ProductVersion** и **Vendor** должны представлять собой строки символов, кодировка UTF-8 которых не должна быть длиннее 268 байтов, и которые не должны содержать символов NUL (0).

9.2.10.3.3 Исходящее или входящее значение **Path** должно представлять собой строку символов, не содержащую символов NUL (0).

9.2.10.3.4 Исходящим значением **SpecVersion** должно быть или корректно представленное целое число (см. 7.4) в интервале от 0 до 255, или пустая строка. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.2.10.3.5 Исходящими значениями **Format_X_FormatOwner** и **Format_X_FormatType** (где **X** = 1, 2, 3 или 4) должны быть или корректно представленные целые числа (см. 7.4) в интервале от 0 до 65535, или пустые строки. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале или пустые строки.

9.2.10.3.6 Исходящими значениями **NumSupportedFormats**, **MaxPayloadSize**, **MaxBSPDbSize** и **MaxIdentify** должны быть или корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295, или пустые строки. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале.

9.2.10.3.7 Исходящими значениями группы параметров «Биометрический тип» должны быть корректно представленные значения собственного типа **BioAPI_BIR_DATA_TYPE** (см. 9.2.5.4). Входящими значениями должны быть стандартно представленные значения этого типа (см. 9.2.5.5).

9.2.10.3.8 Исходящими значениями группы параметров «Операции» должны быть корректно представленные значения собственного типа **BioAPI_OPERATIONS_MASK** (см. 9.2.2.4). Входящими значениями должны быть стандартно представленные значения этого типа (см. 9.2.2.5).

9.2.10.3.9 Исходящими значениями группы параметров «Опции» должны быть корректно представленные значения собственного типа **BioAPI_OPTIONS_MASK** (см. 9.2.3.4). Входящими значениями должны быть стандартно представленные значения этого типа (см. 9.2.3.5).

9.2.10.3.10 Исходящими значениями **PayloadPolicy**, **DefaultVerifyTimeout**, **DefaultIdentifyTimeout**, **DefaultCaptureTimeout**, **DefaultEnrollTimeout** и **DefaultCalibrateTimeout** должны быть или корректно представленные целые числа (см. 7.4) в интервале от минус 2147483648 до плюс 2147483647, или пустые строки. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале.

9.2.10.4 Значение (например, **V**) собственного типа **BioAPI_BSP_SCHEMA**, представленное исходящим значением этой группы параметров, определяют в соответствии с 9.2.10.4.1—9.2.10.4.8.

9.2.10.4.1 УУИД, представленный исходящим значением **BSPUuid** (или УУИД «00000000-0000-0000-0000-000000000000», если исходящим значением является пустая строка), должен быть записан в поле значения **V** с тем же именем.

9.2.10.4.2 Строки символов, которые являются исходящими значениями **Description**, **ProductVersion** и **Vendor** должны быть записаны в поле **V** с тем же именем, заканчивающимся символом конца строки NUL (0).

9.2.10.4.3 Строка символов, которая является выходным значением **Path**, должна быть записана в блок памяти достаточного объема, адрес которого должен быть записан в поле **Path** значения **V**.

9.2.10.4.4 Целые числа, представленные исходящими значениями **SpecVersion**, **NumSupportedFormats**, **PayloadPolicy**, **MaxPayloadSize**, **DefaultVerifyTimeout**, **DefaultIdentifyTimeout**, **DefaultCaptureTimeout**, **DefaultEnrollTimeout**, **DefaultCalibrateTimeout**, **MaxBSPDbSize** и **MaxIdentify** (или 0, если исходящими значениями являются пустые строки) должно быть записано в поле значения **V** с тем же именем.

9.2.10.4.5 Целые числа, представленные исходящими значениями **Format_X_FormatOwner** и **Format_X_FormatType** (где **X** = 1, 2, 3 или 4), или 0, если исходящими значениями являются пустые строки, должны быть записаны в поля **FormatOwner** и **FormatType** (соответственно) элемента в позиции **X** массива четырех элементов типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**. Адрес этого массива должен быть записан в поле **BSPSupportedFormats** значения **V**.

9.2.10.4.6 Значение типа **BioAPI_BIR_BIOMETRIC_TYPE**, представленное (см. 9.2.1.4) исходящим значением группы параметров «Биометрический тип», должно быть записано в поле **FactorsMask** значения **V**.

9.2.10.4.7 Значение типа **BioAPI_OPERATIONS**, представленное (см. 9.2.2.4) исходящим значением группы параметров «Операции», должно быть записано в поле **Operations** значения **V**.

9.2.10.4.8 Значение типа **BioAPI_OPTIONS**, представленное (см. 9.2.3.4) исходящим значением группы параметров «Опции», должно быть записано в поле **Options** значения **V**.

9.2.10.5 Если задано значение (например, **V**) собственного типа **BioAPI_BSP_SCHEMA**, то входящее значение этой группы параметров, которое канонически представляет **V**, следует определять в соответствии с 9.2.10.5.1—9.2.10.5.8.

9.2.10.5.1 Входящим значением **BSPUuid** должен быть стандартно представленный (см. 7.6.3) УУИД в поле значения **V** с тем же именем.

9.2.10.5.2 Входящие значения **Description**, **ProductVersion** и **Vendor** должны представлять собой строки символов в поле значения **V** с тем же именем.

9.2.10.5.3 Входящее значение **Path** должно представлять собой строку символов, на которую указывает значение, находящееся в поле значения **V** с тем же именем.

9.2.10.5.4 Входящими значениями **SpecVersion**, **NumSupportedFormats**, **PayloadPolicy**, **MaxPayloadSize**, **DefaultVerifyTimeout**, **DefaultIdentifyTimeout**, **DefaultCaptureTimeout**, **DefaultEnrollTimeout**, **DefaultCalibrateTimeout**, **MaxBSPDbSize** и **MaxIdentify** должны быть стандартно представленные целые числа в поле значения **V** с тем же именем.

9.2.10.5.5 Входящие значения **Format_X_FormatOwner** и **Format_X_FormatType** (где **X** = 1, 2, 3 или 4) определяют следующим образом. Если поле **BSPSupportedFormats** значения **V** содержит NULL-значение или поле **NumSupportedFormats** значения **V** содержит значение меньше **X**, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в полях **FormatOwner** и **FormatType** (соответственно) элемента в позиции **X**

массива типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, на которую указывает поле **BSPSupportedFormats** значения **V**.

9.2.10.5.6 Входящими значениями группы параметров «Биометрический тип» должны быть стандартно представленные (см. 9.2.1.5) значения типа **BioAPI_BIR_BIOMETRIC_TYPE** в поле **FactorsMask** значения **V**.

9.2.10.5.7 Входящими значениями группы параметров «Операции» должны быть стандартно представленные (см. 9.2.2.5) значения типа **BioAPI_OPERATIONS** в поле **Operations** значения **V**.

9.2.10.5.8 Входящими значениями группы параметров «Опции» должны быть стандартно представленные (см. 9.2.3.5) значения типа **BioAPI_OPTIONS** в поле **Options** значения **V**.

9.2.11 Группа параметров «Схема ПБФ»

9.2.11.1 Группа параметров «Схема ПБФ» включает в себя следующие параметры:

- **BFPUuid;**
- **BFPCategory;**
- **BFPDescription;**
- **Path;**
- **SpecVersion;**
- **ProductVersion;**
- **Vendor;**
- **Format_1_FormatOwner;**
- **Format_1_FormatType;**
- **Format_2_FormatOwner;**
- **Format_2_FormatType;**
- **Format_3_FormatOwner;**
- **Format_3_FormatType;**
- **Format_4_FormatOwner;**
- **Format_4_FormatType;**
- **NumSupportedFormats;**
- элементы группы параметров «Биометрический тип» (см. 9.2.1);
- **BFPPropertyID;**
- **BFPProperty.**

9.2.11.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_BFP_SCHEMA**.

9.2.11.3 У элементов данной группы параметров должны быть входящие и исходящие значения, соответствующие приведенным в 9.2.11.3.1—9.2.11.3.8.

9.2.11.3.1 Исходящими значениями **BFPUuid** и **BFPPropertyID** должны быть или корректно представленные УУИД (см. 7.6), или пустые строки. Входящими значениями должны быть стандартно представленные УУИД (см. 6.3).

9.2.11.3.2 Исходящими значениями **BFPCategory** и **NumSupportedFormats** должны быть или корректно представленные целые числа в интервале от 0 до 4294967295, или пустые строки. Входящими значениями должны быть стандартно представленные целые числа в том же интервале.

9.2.11.3.3 Исходящие или входящие значения **BFPDescription**, **ProductVersion** и **Vendor** должны представлять собой строки символов, кодировка UTF-8 которых не должна быть длиннее 268 байтов и которые не должны содержать символов NUL (0).

9.2.11.3.4 Исходящее или входящее значение **Path** должно представлять собой строку символов, не содержащую символов NUL (0).

9.2.11.3.5 Исходящим значением **SpecVersion** должно быть или корректно представленное целое число в интервале от 0 до 255, или пустая строка. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.2.11.3.6 Исходящими значениями **Format_X_FormatOwner** и **Format_X_FormatType** (где **X** = 1, 2, 3 или 4) должны быть или корректно представленные целые числа (см. 7.4) в интервале от 0 до 65535, или пустые строки. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале или пустые строки.

9.2.11.3.7 Исходящими значениями группы параметров «Биометрический тип» должны быть корректно представленные значения собственного типа **BioAPI_BIR_DATA_TYPE** (см. 9.2.5.4). Входящими значениями должны быть стандартно представленные значения этого типа (см. 9.2.5.5).

9.2.11.3.8 Исходящим значением **BFPProperty** должен быть корректно представленный набор байтов (см. 7.7). Входящим значением должен быть стандартно представленный набор байтов (см. 7.7.2).

9.2.11.4 Значение (например, **V**) собственного типа **BioAPI_BFP_SCHEMA**, представленное исходящим значением этой группы параметров, определяют в соответствии с 9.2.11.4.1—9.2.11.4.8.

9.2.11.4.1 УУИД, представленный исходящим значением **BSPUuid** и **BFPPropertyID** (или УУИД «00000000-0000-0000-0000-000000000000», если исходящим значением является пустая строка) должен быть записан в поле значения **V** с тем же именем.

9.2.11.4.2 Целые числа, представленные исходящими значениями **BFPCategory**, и **NumSupportedFormats** (или 0, если исходящими значениями являются пустые строки), должны быть записаны в поле значения **V** с тем же именем.

9.2.11.4.3 Строки символов, являющиеся исходящими значениями **BFPDescription**, **ProductVersion** и **Vendor** должны быть записаны в поле значения **V** с теми же именами, заканчивающимися символом конца строки NUL (0).

9.2.11.4.4 Строка символов, которая является выходным значением **Path**, должна быть записана с символом NUL (0) в конце в блок памяти достаточного объема, адрес которого должен быть вписан в поле **Path** значения **V**.

9.2.11.4.5 Целое число, представленное исходящим значением **SpecVersion** (или 0, если исходящим значением является пустая строка), должно быть записано в поле значения **V** с тем же именем.

9.2.11.4.6 Целые числа, представленные исходящими значениями **Format_X_FormatOwner** и **Format_X_FormatType** (где **X** = 1, 2, 3 или 4), или 0, если исходящим значением является пустая строка, должны быть записаны в поля **FormatOwner** и **FormatType** (соответственно) элемента в позиции **X** массива четырех элементов типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**. Адрес этого массива должен быть записан в поле **BFPSupportedFormats** значения **V**.

9.2.11.4.7 Значение типа **BioAPI_BIR_BIOMETRIC_TYPE**, представленное (см. 9.2.1.4) исходящим значением группы параметров «Биометрический тип», должно быть записано в поле **FactorsMask** значения **V**.

9.2.11.4.8 Набор байтов, представленный исходящим значением **BFPProperty**, должен быть записан в блок памяти достаточного объема. Адрес и длина блока памяти должны быть записаны в поля **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, адрес которой должен быть записан в поле **BFPProperty** значения **V**.

9.2.11.5 Если задано значение (например, **V**) собственного типа **BioAPI_BFP_SCHEMA**, то входящее значение этой группы параметров, которое канонически представляет значение **V**, следует определять в соответствии с 9.2.11.5.1—9.2.11.5.8.

9.2.11.5.1 Входящими значениями **BFPUuid** и **BFPPropertyID** должны быть стандартно представленные (см. 7.6.3) УУИД в поле значения **V** с тем же именем.

9.2.11.5.2 Входящими значениями **BFPCategory** и **NumSupportedFormats** должны быть стандартно представленные целые числа в поле значения **V** с тем же именем.

9.2.11.5.3 Входящие значения **BFPDescription**, **ProductVersion** и **Vendor** должны представлять собой строки символов в поле значения **V** с тем же именем и без символа NUL (0) в конце строки.

9.2.11.5.4 Входящее значение **Path** должно представлять собой строку символов, на которую указывает поле значения **V** с тем же именем и без символа NUL (0) в конце строки.

9.2.11.5.5 Входящим значением **SpecVersion** должно быть стандартно представленное целое число в поле значения **V** с тем же именем.

9.2.11.5.6 Входящие значения **Format_X_FormatOwner** и **Format_X_FormatType** (где **X** = 1, 2, 3 или 4) определяют следующим образом.

Если поле **BFPSupportedFormats** значения **V** содержит значение NULL или значение поля **NumSupportedFormats** значения **V** меньше **X**, то входящим значением должна быть пустая строка. В противном случае входящими значениями должны быть стандартно представленные целые числа в полях **FormatOwner** и **FormatType** (соответственно) элемента в позиции **X** массива типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, на которую указывает поле **BFPSupportedFormats** значения **V**.

9.2.11.5.7 Входящими значениями группы параметров «Биометрический тип» должны быть стандартно представленные (см. 9.2.1.5) значения типа **BioAPI_BIR_BIOMETRIC_TYPE** в поле **FactorsMask** значения **V**.

9.2.11.5.8 Входящим значением **BFPProperty** должен быть стандартно представленный набор байтов в блоке памяти, адрес и длина которого находятся в полях **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, на которую указывает поле **BFPProperty** значения **V**.

9.2.12 Группа параметров «Структура элемента»

9.2.12.1 Группа параметров «Структура элемента» включает в себя следующие параметры:

- **BSPUuid;**
- **UnitManagerUuid;**
- **UnitID;**
- **UnitCategory;**
- **UnitProperties;**
- **VendorInformation;**
- элементы группы параметров «События» (см. пункт 9.2.4);
- **UnitPropertyID;**
- **UnitProperty;**
- **HardwareVersion;**
- **FirmwareVersion;**
- **SoftwareVersion;**
- **HardwareSerialNumber;**
- **AuthenticatedHardware;**
- **MaxBSPDbSize;**
- **MaxIdentify.**

9.2.12.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_UNIT_SCHEMA**.

9.2.12.3 У элементов данной группы параметров должны быть входящие и исходящие значения, соответствующие приведенным в 9.2.12.3.1—9.2.12.3.6.

9.2.12.3.1 Исходящими значениями **BSPUuid**, **UnitManagerUuid**, **UnitProperties** и **UnitPropertyID** должны быть или корректно представленные УУИД (см. 7.6), или пустые строки. Входящими значениями должны быть стандартно представленные УУИД (см. 7.6.3).

9.2.12.3.2 Исходящими значениями **UnitID**, **UnitCategory**, **MaxBSPDbSize** и **MaxIdentify** должны быть или корректно представленные целые числа в интервале от 0 до 4294967295, или пустые строки. Входящими значениями должны быть стандартно представленные целые числа в том же интервале.

9.2.12.3.3 Исходящие или входящие значения **VendorInformation**, **HardwareVersion**, **FirmwareVersion**, **SoftwareVersion** и **HardwareSerialNumber** должны представлять собой строки символов, кодировка UTF-8 которых не должна быть длиннее 268 байтов и которые не должны содержать символы NUL (0).

9.2.12.3.4 Исходящими значениями группы параметров «События» должны быть корректно представленные значения собственного типа **BioAPI_EVENT_MASK**. Входящими значениями должны быть стандартно представленные значения этого типа.

9.2.12.3.5 Исходящим значением **UnitProperty** должен быть корректно представленный набор байтов (см. 7.7). Входящим значением должен быть стандартно представленный набор байтов (см. 7.7.2).

9.2.12.3.6 Исходящим значением **AuthenticatedHardware** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.2.12.4 Значение (например, **V**) собственного типа **BioAPI_UNIT_SCHEMA**, представленное исходящим значением этой группы параметров, должно определяться в соответствии с 9.2.12.4.1—9.2.12.4.6.

9.2.12.4.1 УУИД, представленные исходящими значениями **BSPUuid**, **UnitManagerUuid**, **UnitProperties** и **UnitPropertyID** (или УУИД «00000000-0000-0000-0000-000000000000», если исходящее значение является пустой строкой) должны быть записаны в поле значения **V** с тем же именем.

9.2.12.4.2 Целые числа, представленные исходящими значениями **UnitID**, **UnitCategory**, **MaxBSPDbSize** и **MaxIdentify** (или 0, если исходящим значением является пустая строка), должны быть записаны в поле значения **V** с тем же именем.

9.2.12.4.3 Значение типа **BioAPI_EVENT_MASK**, представленное (см. 9.2.4.4) исходящим значением группы параметров «События», должно быть записано в поле **SupportedEvents** значения **V**.

9.2.12.4.4 Строки символов, которые являются исходящими значениями **VendorInformation**, **HardwareVersion**, **FirmwareVersion**, **SoftwareVersion** и **HardwareSerialNumber**, должны быть записаны в поле значения **V** с тем же именем, заканчивающимся символом конца строки NUL (0).

9.2.12.4.5 Набор байтов, представленный исходящим значением **UnitProperty**, должен быть записан в блок памяти достаточного объема. Адрес и длина блока памяти должны быть записаны в поля **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, адрес которой должен быть записан в поле **UnitProperty** значения **V**.

9.2.12.4.6 Логическое значение, представленное исходящим значением **AuthenticatedHardware** (или **"false"**, если исходящим значением является пустая строка), должно быть записано в поле значения **V** с тем же именем.

9.2.12.5 Если задано значение (например, **V**) собственного типа **BioAPI_UNIT_SCHEMA**, то входящее значение этой группы параметров, которое канонически представляет значение **V**, определяют в соответствии с 9.2.12.5.1—9.2.12.5.6.

9.2.12.5.1 Входящими значениями **BSPUuid**, **UnitManagerUuid**, **UnitProperties** и **UnitPropertyID** должны быть стандартно представленные (см. 7.6.3) УУИД в поле значения **V** с тем же именем.

9.2.12.5.2 Входящими значениями **UnitID**, **UnitCategory**, **MaxBSPDbSize** и **MaxIdentify** должны быть стандартно представленные целые числа в поле значения **V** с тем же именем.

9.2.12.5.3 Входящими значениями группы параметров «События» должны быть стандартно представленные (см. 9.2.1.5) значения типа **BioAPI_EVENT_MASK** в поле **SupportedEvents** значения **V**.

9.2.12.5.4 Входящие значения **VendorInformation**, **HardwareVersion**, **FirmwareVersion**, **SoftwareVersion** и **HardwareSerialNumber** должны представлять собой строки символов в поле **V** с тем же именем и без символа NUL (0) в конце строки.

9.2.12.5.5 Входящим значением **UnitProperty** должен быть стандартно представленный набор байтов в блоке памяти, адрес и длина которого находятся в полях **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, на которую указывает поле **UnitProperty** значения **V**.

9.2.12.5.6 Входящим значением **AuthenticatedHardware** должно быть стандартно представленное (см. 7.5.2) логическое значение в поле значения **V** с тем же именем.

9.2.13 Группа параметров «Заголовок ЗБИ»

9.2.13.1 Группа параметров «Заголовок ЗБИ» включает в себя следующие параметры:

- **HeaderVersion;**
- элементы группы параметров «Тип биометрических данных» (см. 9.2.5);
- **FormatOwner;**
- **FormatType;**
- **Quality;**
- **Purpose;**
- элементы группы параметров «Биометрический тип» (см. 9.2.1);
- **ProductOwner;**
- **ProductType;**
- элементы группы параметров «Дата и время» (см. 9.2.8), с именами, начинающимися с **"Creation_"**;
- элементы группы параметров «Биометрический подтип» (см. 9.2.6);
- элементы группы параметров «Дата» (см. 9.2.7) с именами, начинающимися с **"Expiration_"**;
- **SBFormatOwner;**
- **SBFormatType;**
- **Index.**

9.2.13.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_BIR_HEADER**.

9.2.13.3 У элементов данной группы параметров должны быть входящие и исходящие значения, соответствующие приведенным в 9.2.13.3.1—9.2.13.3.9.

9.2.13.3.1 Исходящими значениями **HeaderVersion** и **Purpose** должны быть или корректно представленные целые числа в интервале от 0 до 255, или пустые строки. Входящими значениями должны быть стандартно представленные целые числа в том же интервале.

9.2.13.3.2 Исходящими значениями группы параметров «Тип биометрических данных» должны быть корректно представленные значения собственного типа **BioAPI_BIR_DATA_TYPE**. Входящими значениями должны быть стандартно представленные значения этого типа.

9.2.13.3.3 Исходящими значениями **FormatOwner**, **FormatType**, **ProductType**, **SBFormatOwner** и **SBFormatType** должны быть или корректно представленные целые числа в интервале от 0 до 65535, или пустые строки. Входящими значениями должны быть стандартно представленные целые числа в том же интервале.

9.2.13.3.4 Исходящим значением **Quality** должно быть или корректно представленное целое число в интервале от минус 128 до плюс 127, или пустая строка. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.2.13.3.5 Исходящими значениями группы параметров «Биометрический тип» должны быть корректно представленные значения собственного типа **BioAPI_BIR_BIOMETRIC_TYPE**. Входящими значениями должны быть стандартно представленные значения этого типа.

9.2.13.3.6 Исходящими значениями группы параметров «Дата и время» должны быть корректно представленные значения собственного типа **BioAPI_DTG**. Входящими значениями должны быть стандартно представленные значения этого типа.

9.2.13.3.7 Исходящими значениями группы параметров «Биометрический подтип» должны быть корректно представленные значения собственного типа **BioAPI_BIR_SUBTYPE**. Входящими значениями должны быть стандартно представленные значения этого типа.

9.2.13.3.8 Исходящими значениями группы параметров «Дата» должны быть корректно представленные значения собственного типа **BioAPI_DATE**. Входящими значениями должны быть стандартно представленные значения этого типа.

9.2.13.3.9 Исходящим значением **Index** должен быть или корректно представленный УУИД (см. 7.6), или пустая строка. Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.2.13.4 Значение (например, **V**) собственного типа BioAPI_BIR_HEADER, представленное исходящим значением этой группы параметров, определяют в соответствии с 9.2.13.4.1—9.2.13.4.10.

9.2.13.4.1 Целые числа, представленные исходящими значениями **HeaderVersion**, **Quality** и **Purpose** (или 0, если исходящим значением является пустая строка), должны быть записаны в поле значения **V** с тем же именем.

9.2.13.4.2 Значение типа **BioAPI_BIR_DATA_TYPE**, представленное (см. 9.2.5.4) исходящим значением группы параметров «Тип биометрических данных», должно быть записано в поле **Type** значения **V**.

9.2.13.4.3 Целые числа, представленные исходящими значениями **FormatOwner** и **FormatType** (или 0, если исходящим значением является пустая строка), должны быть записаны в поля **FormatOwner** и **FormatType** (соответственно) поля **Format** значения **V**.

9.2.13.4.4 Значение типа **BioAPI_BIR_BIOMETRIC_TYPE**, представленное (см. 9.2.1.4) исходящим значением группы параметров «Биометрический тип», должно быть записано в поле **FactorsMask** значения **V**.

9.2.13.4.5 Целые числа, представленные исходящими значениями **ProductOwner** и **ProductType** (или 0, если исходящим значением является пустая строка), должны быть записаны в поля **ProductOwner** и **ProductType** (соответственно) поля **ProductID** значения **V**.

9.2.13.4.6 Значение типа **BioAPI_DTG**, представленное (см. 9.2.8.4) исходящим значением группы параметров «Дата и время» (с префиксом **"Creation_"**), должно быть записано в поле **CreationDTG** значения **V**.

9.2.13.4.7 Значение типа **BioAPI_BIR_SUBTYPE**, представленное (см. 9.2.6.4) исходящим значением группы параметров «Биометрический подтип», должно быть записано в поле **Subtype** значения **V**.

9.2.13.4.8 Значение типа **BioAPI_DATE**, представленное (см. 9.2.7.4) исходящим значением группы параметров «Дата»(с префиксом **"Expiration_"**), должно быть записано в поле **ExpirationDate** значения **V**.

9.2.13.4.9 Целые числа, представленные исходящими значениями **SBFormatOwner** и **SBFormatType** (или 0, если исходящим значением является пустая строка), должны быть записаны в поля **SecurityFormatOwner** и **SecurityFormatType** (соответственно) поля **SBFormat** значения **V**.

9.2.13.4.10 УУИД, представленный исходящим значением **Index** (или УУИД «00000000-0000-0000-0000-000000000000», если исходящим значением является пустая строка), должен быть записан в поле значения **V** с тем же именем.

9.2.13.5 Если задано значение (например, **V**) собственного типа **BioAPI_BIR_HEADER**, то входящее значение этой группы параметров, которое канонически представляет значение **V**, определяют в соответствии с 9.2.13.5.1—9.2.13.5.2.

9.2.13.5.1 Входящими значениями **HeaderVersion**, **Quality** и **Purpose** должны быть стандартно представленные целые числа в поле значения **V** с тем же именем.

9.2.13.5.2 Входящими значениями группы параметров «Тип биометрических данных» должны быть стандартно представленные (см. 9.2.5.5) значения типа **BioAPI_BIR_DATA_TYPE** в поле **Type** значения **V**.

9.2.13.5.3 Входящими значениями **FormatOwner** и **FormatType** должны быть стандартно представленные целые числа в полях **FormatOwner** и **FormatType** (соответственно) поля **Format** значения **V**.

9.2.13.5.4 Входящим значением группы параметров «Биометрический тип» должно быть стандартно представленное (см. 9.2.1.5) значение типа **BioAPI_BIR_BIOMETRIC_TYPE** в поле **FactorsMask** значения **V**.

9.2.13.5.5 Входящими значениями **ProductOwner** и **ProductType** должны быть стандартно представленные целые числа в полях **ProductOwner** и **ProductType** (соответственно) поля **ProductID** значения **V**.

9.2.13.5.6 Входящими значениями группы параметров «Дата и время» (с префиксом **"Creation_"**) должны быть стандартно представленные (см. 9.2.8.5) значения типа **BioAPI_DTG** в поле **CreationDTG** значения **V**.

9.2.13.5.7 Входящими значениями группы параметров «Биометрический подтип» должны быть стандартно представленные (см. 9.2.6.5) значения типа **BioAPI_BIR_SUBTYPE** в поле **Subtype** значения **V**.

9.2.13.5.8 Входящими значениями группы параметров «Дата» (с префиксом **"Expiration_"**) должны быть стандартно представленные (см. 9.2.7.5) значения типа **BioAPI_DATE** в поле **ExpirationDate** значения **V**.

9.2.13.5.9 Входящими значениями **SBFormatOwner** и **SBFormatType** должны быть стандартно представленные целые числа в полях **SecurityFormatOwner** и **SecurityFormatType** (соответственно) поля **SBFormat** значения **V**.

9.2.13.5.10 Входящим значением **Index** должен быть стандартно представленный (см. 7.6.3) УУИД в поле значения **V** с тем же именем.

9.2.14 Группа параметров «ЗБИ»

9.2.14.1 Группа параметров «ЗБИ» включает в себя следующие параметры:

- элементы группы параметров «Заголовок ЗБИ» (см. 9.2.13);
- **BiometricData;**
- **SecutityBlock.**

9.2.14.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_BIR**.

9.2.14.3 У элементов данной группы параметров должны быть входящие и исходящие значения, соответствующие приведенным в 9.2.14.3.1 и 9.2.14.3.2.

9.2.14.3.1 Исходящими значениями группы параметров «Заголовок ЗБИ» должны быть корректно представленные значения собственного типа **BioAPI_BIR_HEADER**. Входящими значениями должны быть стандартно представленные значения этого типа.

9.2.14.3.2 Исходящими значениями **BiometricData** и **SecurityBlock** должны быть корректно представленные наборы байтов (см. 7.7). Входящими значениями должны быть стандартно представленные наборы байтов (см. 7.7.2).

9.2.14.4 Значение (например, **V**) собственного типа **BioAPI_BIR**, представленное исходящим значением этой группы параметров, определяют в соответствии с 9.2.14.4.1—9.2.14.4.3.

9.2.14.4.1 Значение типа **BioAPI_BIR_HEADER**, представленное (см. 9.2.13.4) исходящим значением группы параметров «Заголовок ЗБИ», должно быть записано в поле **Header** значения **V**.

9.2.14.4.2 Набор байтов, представленный исходящим значением **BiometricData**, должен быть записан в блок памяти достаточного объема. Адрес и длина блока памяти должны быть записаны в поля **Data** и **Length** (соответственно) поля **BiometricData** значения **V**.

9.2.14.4.3 Набор байтов, представленный исходящим значением **SecurityBlock**, должен быть записан в блок памяти достаточного объема. Адрес и длина блока памяти должны быть записаны в поля **Data** и **Length** (соответственно) поля **SecurityBlock** значения **V**.

9.2.14.5 Если задано значение (например, **V**) собственного типа **BioAPI_BIR**, то входящее значение этой группы параметров, которое канонически представляет значение **V**, определяют в соответствии с 9.2.14.5.1— 9.2.14.5.3.

9.2.14.5.1 Входящими значениями группы параметров «Заголовок ЗБИ» должны быть стандартно представленные (см. 9.2.13.5) значения типа **BioAPI_BIR_HEADER** в поле **Header** значения **V**.

9.2.14.5.2 Входящим значением **BiometricData** должен быть стандартно представленный набор байтов в блоке памяти, адрес и длина которого находятся в полях **Data** и **Length** (соответственно) поля **BiometricData** значения **V**.

9.2.14.5.3 Входящим значением **SecurityBlock** должен быть стандартно представленный набор байтов в блоке памяти, адрес и длина которого находятся в полях **Data** и **Length** (соответственно) поля **SecurityBlock** значения **V**.

9.2.15 Группа «Входные параметры ЗБИ»

9.2.15.1 Группа «Входные параметры ЗБИ» включает в себя следующие параметры:

- **Form;**
- **DbHandle;**
- **KeyValue;**

- BIRHandle;

- элементы группы параметров «ЗБИ»(см. 9.2.14).

9.2.15.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_INPUT_BIR**.

9.2.15.3 У элементов данной группы параметров должны быть входящие и исходящие значения, соответствующие приведенным в 9.2.15.3.1—9.2.15.3.4.

9.2.15.3.1 Исходящим значением **Form** должно быть или корректно представленное целое число (см. 7.4) в интервале от 0 до 255, или пустая строка. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.2.15.3.2 Исходящими значениями **DbHandle** и **BIRHandle** должны быть или корректно представленные целые числа в интервале от минус 2147483648 до плюс 2147483647, или пустые строки. Входящими значениями должны быть стандартно представленные целые числа в том же интервале.

9.2.15.3.3 Исходящим значением **KeyValue** должен быть или корректно представленный УУИД (см. 7.6), или пустая строка. Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.2.15.3.4 Исходящими значениями группы параметров «ЗБИ» должны быть корректно представленные значения собственного типа **BioAPI_BIR**. Входящими значениями должны быть стандартно представленные значения этого типа.

9.2.15.4 Значение (например, **V**) собственного типа **BioAPI_INPUT_BIR**, представленное исходящим значением этой группы параметров, следует определять в соответствии с 9.2.15.4.1—9.2.15.4.6.

9.2.15.4.1 Целое число (например, **F**), представленное исходящим значением **Form** (или 0, если исходящим значением является пустая строка), должно быть записано в поле значения **V** с тем же именем.

9.2.15.4.2 Если **F** равно 1, то:

a) целое число, представленное исходящим значением **DbHandle** (или 0, если исходящим значением является пустая строка), должно быть записано в поле **DbHandle** переменной типа **BioAPI_DBBIR_ID**;

b) УУИД, представленный исходящим значением **KeyValue** (или УУИД «00000000-0000-0000-0000--000000000000», если исходящим значением является пустая строка) должен быть записан в поле **KeyVariable** переменной;

c) адрес этой переменной должен быть записан в поле **BIRInDb** поля **InputBIR** значения **V**.

9.2.15.4.3 Если **F** равно 2, то:

a) целое число, представленное исходящим значением **BIRHandle** (или 0, если исходящим значением является пустая строка), должно быть записано в переменную типа **BioAPI_BIR_HANDLE**;

b) адрес этой переменной должен быть записан в поле **BIRInBSP** поля **InputBIR** значения **V**.

9.2.15.4.4 Если **F** равно 3, то:

a) значение типа **BioAPI_BIR**, представленное (см. 9.2.14.4) исходящим значением группы параметров «ЗБИ», должно быть записано в переменную типа **BioAPI_BIR**;

b) адрес этой переменной должен быть записан в поле **BIR** поля **InputBir** значения **V**.

9.2.15.4.5 Если **F** меньше 1 или больше 3, то в поле **BIR** поля **InputBIR** должно быть записано NULL-значение.

9.2.15.5 Если задано значение (например, **V**) собственного типа **BioAPI_INPUT_BIR**, то входящее значение этой группы параметров, которое канонически представляет значение **V**, следует определять в соответствии с 9.2.15.5.1—9.2.15.5.5.

9.2.15.5.1 Входящим значением **Form** должно быть стандартно представленное целое число (например, **F**) в поле значения **V** с тем же именем.

9.2.15.5.2 Если **F** равно 1, то:

a) входящим значением **DbHandle** должно быть стандартно представленное целое число в поле **DbHandle** переменной типа **BioAPI_DBBIR**, на которую указывает поле **BIRInDb** поля **InputBIR** значения **V**;

b) входящим значением **KeyValue** должен быть стандартно представленный (см. 7.6.3) УУИД в поле **KeyValue** этой переменной;

c) входящими значениями всех остальных параметров должны быть пустые строки.

9.2.15.5.3 Если **F** равно 2, то:

a) входящим значением **BIRHandle** должно быть стандартно представленное целое число в переменной типа **BioAPI_BIR_HANDLE**, на которую указывает поле **BIRInBSP** поля **InputBIR** значения **V**, и

b) входящими значениями всех остальных параметров должны быть пустые строки.

9.2.15.5.4 Если **F** равно 3, то:

a) входящими значениями группы параметров «ЗБИ» должны быть стандартно представленные (см. 9.2.14.5) значения типа **BioAPI_BIR** в поле **BIR** поля **InputBIR** значения **V**;

b) входящими значениями всех остальных параметров должны быть пустые строки.

9.2.15.5.5 Если **F** меньше 1 или больше 3, то входящими значениями всех параметров кроме **Form** должны быть пустые строки.

9.2.16 Группа параметров «Определение заполнения»

9.2.16.1 Группа параметров «Определение заполнения» включает в себя следующие параметры:

- Type;

- BIRDataBase;

- NumberOfMembers;

- элементы группы параметров «ЗБИ»(см. 9.2.14) с именами, начинающимися с «BIR_1_»;
- элементы группы параметров «ЗБИ» с именами, начинающимися с «BIR_2_»;
- элементы группы параметров «ЗБИ» с именами, начинающимися с «BIR_3_»;
- элементы группы параметров «ЗБИ» с именами, начинающимися с «BIR_4_».

9.2.16.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_IDENTIFY_POPULATION**.

9.2.16.3 У элементов данной группы параметров должны быть входящие и исходящие значения, соответствующие определенным в 9.2.16.3.1—9.2.16.3.4.

9.2.16.3.1 Исходящим значением **Type** должно быть или корректно представленное целое число (см. 7.4) в интервале от 0 до 255, или пустая строка. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.2.16.3.2 Исходящим значением **BIRDataBase** должно быть или корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647, или пустая строка. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.2.16.3.3 Исходящим значением **NumberOfMembers** должно быть или корректно представленное целое число в интервале от 0 до 4, или пустая строка. Входящим значением **NumberOfMembers** должно быть стандартно представленное целое число в интервале от 0 до 4294967295.

9.2.16.3.4 Исходящими значениями каждой группы параметров «ЗБИ» должны быть корректно представленные значения собственного типа **BioAPI_BIR**. Входящими значениями должны быть стандартно представленные значения этого типа или наборы пустых строк.

9.2.16.4 Значение (например, **V**) собственного типа **BioAPI_IDENTIFY_POPULATION**, представленное исходящим значением этой группы параметров, следует определять в соответствии с 9.2.16.4.1—9.2.16.4.4.

9.2.16.4.1 Целое число (например, **T**), представленное исходящим значением **Type** (или 0, если исходящим значением является пустая строка), должно быть записано в поле значения **V** с тем же именем.

9.2.16.4.2 Если **T** равно 1, то:

a) целое число, представленное исходящим значением **BIRDataBase** (или 0, если исходящим значением является пустая строка), должно быть записано в переменную типа **BioAPI_DB_HANDLE**.

b) адрес этой переменной должен быть записан в поле **BIRDataBase** поля **BIRs** значения **V**.

9.2.16.4.3 Если **T** равно 2, то:

a) целое число, представленное исходящим значением **NumberOfMembers** (или 0, если исходящим значением является пустая строка), должно быть записано в поле **NumberOfMembers** переменной типа **BioAPI_BIR_ARRAY_POPULATION**;

b) значение типа **BioAPI_BIR**, представленное исходящим значением группы параметров «ЗБИ» с префиксом **"BIR_X_"** (где **X** = 1, 2, 3 или 4), должно быть записано в переменную типа **BioAPI_BIR**;

c) адреса всех переменных типа **BioAPI_BIR**, определенных в перечислении b), должны быть записаны по порядку в массив указателей;

d) адрес массива указателей должен быть записан в поле **Members** переменной типа **BioAPI_BIR_ARRAY_POPULATION**, указанной в перечислении a);

e) адрес этой переменной должен быть записан в поле **BIRArray** поля **BIRs** значения **V**.

9.2.16.4.4 Если **T** меньше 1 или больше 2, то в поле **BIRArray** поля **BIRs** значения **V** должно быть записано NULL-значение.

9.2.16.5 Если задано значение (например, **V**) собственного типа **BioAPI_IDENTIFY_POPULATION**, то входящее значение этой группы параметров, которое канонически представляет значение **V**, следует определять в соответствии с 9.2.16.5.1—9.2.16.5.4.

9.2.16.5.1 Входящим значением **Type** должно быть стандартно представленное целое число (например, **T**) в поле значения **V** с тем же именем.

9.2.16.5.2 Если **T** равно 1, то:

a) входящим значением **BIRDataBase** должно быть стандартно представленное целое число в поле **BioAPI_DB_Handle**, на которое указывает поле **BIRDataBase** поля **BIRs** значения **V**;

b) входящими значениями всех остальных параметров должны быть пустые строки.

9.2.16.5.3 Если **T** равно 2, то:

a) входящим значением **NumberOfMembers** должно быть стандартно представленное целое число в поле **NumberOfMembers** переменной типа **BioAPI_BIR_ARRAY**, на которую указывает поле **BIRArray** поля **BIRs** значения **V**;

b) входящие значения группы параметров «ЗБИ» с префиксом **"BIR_X_"** (где **X** = 1, 2, 3 или 4) следует определять следующим образом. Если поле **Members** переменной, указанной в перечислении a), имеет значение NULL или поле **NumberOfMembers** меньше **X**, то входящими значениями должен быть набор пустых строк. В противном случае входящими значениями должны быть стандартно представленные входящие значения типа **BioAPI_BIR**, на которые указывает элемент в позиции **X** массива указателей, на который указывает поле **Members**;

c) входящим значением **BIRDataBase** должна быть пустая строка.

9.2.16.5.4 Если **T** меньше 1 или больше 2, то входящими значениями всех параметров, кроме **Type,** должны быть пустые строки.

9.2.17 Группа параметров «Кандидат»

9.2.17.1 Группа параметров «Кандидат» включает в себя следующие параметры:

- Type;

- BIRInDataBase;

- BIRInArray;

- FMRAchieved.

9.2.17.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_CANDIDATE**.

9.2.17.3 У элементов данной группы параметров должны быть входящие и исходящие значения, соответствующие приведенным в 9.2.17.3.1—9.2.17.3.4.

9.2.17.3.1 Исходящим значением **Type** должно быть или корректно представленное целое число (см. 7.4) в интервале от 0 до 255, или пустая строка. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.2.17.3.2 Исходящим значением **BIRInDataBase** должен быть или корректно представленный УУИД (см. 7.6), или пустая строка. Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.2.17.3.3 Исходящим значением **BIRInArray** должно быть или корректно представленное целое число в интервале от 0 до 4294967295, или пустая строка. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.2.17.3.4 Исходящим значением **FMRAchieved** должно быть или корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647, или пустая строка. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.2.17.4 Значение (например, **V**) собственного типа **BioAPI_CANDIDATE**, представленное исходящим значением этой группы параметров, следует определять в соответствии с 9.2.17.4.1—9.2.17.4.5.

9.2.17.4.1 Целое число (например, **T**), представленное исходящим значением **Type** (или 0, если исходящее значение является пустой строкой), должно быть записано в поле **Type** значения **V**.

9.2.17.4.2 Если **T** равно 1, то:

a) УУИД, представленный исходящим значением **BIRInDataBase** (или УУИД «00000000-0000-0000--0000-000000000000», если исходящим значением является пустая строка), должен быть записан в переменную **BioAPI_UUID**;

b) адрес этой переменной должен быть записан в поле **BIRInDataBase** поля **BIR** значения **V**.

9.2.17.4.3 Если **T** равно 2 или 3, то:

a) целое число, представленное исходящим значением **BIRInArray** (или 0, если исходящее значение является пустой строкой), должно быть записано в переменную типа **uint32_t**;

b) адрес этой переменной должен быть записан в поле **BIRInArray** поля **BIR** значения **V**.

9.2.17.4.4 Если **T** меньше 1 или больше 3, то поле **BIRInArray** поля **BIR** значения **V** должно быть равно нулю.

9.2.17.4.5 Целое число, представленное исходящим значением **FMRAchieved** (или 0, если исходящее значение является пустой строкой), должно быть записано в поле значения **V** с тем же именем.

9.2.17.5 Если задано значение (например, **V**) собственного типа **BioAPI_CANDIDATE**, то входящее значение этой группы параметров, которое канонически представляет значение **V**, следует определять в соответствии с 9.2.17.5.1—9.2.17.5.5.

9.2.17.5.1 Входящим значением **Type** должно быть стандартно представленное целое число (например, **T**) в поле значения **V** с тем же именем.

9.2.17.5.2 Если **T** равно 1, то входящим значением **BIRInDataBase** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает поле **BIRInDataBase** значения **V**. В противном случае входящим значением **BIRInDataBase** должна быть пустая строка.

9.2.17.5.3 Если **T** равно 2 или 3, то входящим значением **BIRInArray** должно быть стандартно представленное целое число типа **uint32_t**, на которое указывает поле **BIRInArray** значения **V**. В противном случае входящим значением **BIRInArray** должна быть пустая строка.

9.2.17.5.4 Входящим значением **FMRAchieved** должно быть стандартно представленное целое число (например, **T**) в поле значения **V** с тем же именем.

9.2.18 Группа параметров «Состояние ГИП»

9.2.18.1 Группа параметров «Состояние ГИП» включает в себя следующие параметры:

- **SampleAvailable;**
- **MessageProvided;**
- **ProgressProvided.**

9.2.18.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_GUI_STATE** (являющегося целочисленным типом).

9.2.18.3 Исходящими значениями этих параметров должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.2.18.4 Значение (например, **V**) собственного типа **BioAPI_GUI_STATE**, представленное исходящим значением этой группы параметров, следует определять следующим образом:

1) сначала устанавливают значение **V**, равное нулю;

2) для каждого элемента группы параметров, исходящее значение которых соответствует значению **"true"**, к значению **V** следует прибавить целое число в соответствии с таблицей 8.

Таблица 8

SampleAvailable	$1 = 2^0$
MessageProvided	$2 = 2^1$
ProgressProvided	$4 = 2^2$

9.2.18.5 Если задано целое число (например, **V**), которое является значением собственного типа **BioAPI_BIR_GUI_STATE**, то входящее значение этой группы параметров, которое канонически представляет значение **V**, следует определять следующим образом:

1) целое число **V** должно быть разбито на сумму показателей степеней числа 2, каждый из которых присутствует в показателях не более одного раза;

2) для каждого показателя степени числа 2, который появляется в разбиении целого числа **V**, входящим значением соответствующего параметра (согласно таблице 8) должно быть **"true"**;

3) входящими значениями всех остальных параметров (в случае их наличия) должно быть **"false"**.

9.2.19 Группа параметров «Тип доступа»

9.2.19.1 Группа параметров «Тип доступа» включает в себя следующие параметры:

- **ReadAccess;**
- **WriteAccess.**

9.2.19.2 Эта группа параметров поддерживает представление значений собственного типа **BioAPI_DB_ACCESS_TYPE** (являющегося типом целого числа).

9.2.19.3 Исходящими значениями этих параметров должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.2.19.4 Значение (например, **V**) собственного типа **BioAPI_DB_ACCESS_TYPE**, представленное исходящим значением этой группы параметров, следует определять следующим образом:

1) сначала устанавливают значение **V**, равное нулю;

2) для каждого элемента группы параметров, исходящее значение которого соответствует значению **"true"**, к значению **V** следует прибавить целое число в соответствии с таблицей 9.

Таблица 9

ReadAccess	$1 = 2^0$
WriteAccess	$2 = 2^1$

9.2.19.5 Если задано целое число (например, **V**), которое является значением собственного типа **BioAPI_DB_ACCESS_TYPE**, то входящее значение этой группы параметров, которое канонически представляет значение **V**, следует определять следующим образом:

1) целое число **V** должно быть разбито на сумму показателей степеней числа 2, каждый из которых присутствует в показателях не более одного раза;

2) для каждого показателя степени числа 2, который появляется в разбиении целого числа **V**, входящим значением соответствующего параметра (согласно таблице) должно быть **"true"**;

3) входящим значением всех остальных параметров (при их наличии) должно быть **"false"**.

9.3 Функция BioAPI_Init

9.3.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

BioAPI_RETURN BioAPI
BioAPI_Init (
BioAPI_VERSION Version);

и параметры, указанные в таблице 10.

Таблица 10

Параметр	Вызов функции (модель испытания на соответствие инфраструктур)	Вызов связанного процесса (модель испытания на соответствие приложений)
Version	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.3.2 Ограничения на параметры

9.3.2.1 Исходящим значением **Version** должно быть или корректно представленное целое число (см. 7.4) в интервале от 0 до 255, или пустая строка. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.3.2.2 Возвращаемым значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.3.3 Входные параметры вызова функции

Целое число, представленное исходящим значением **Version**, должно быть присвоено первичному параметру с тем же именем.

9.3.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.3.5 Входные параметры вызова связанного процесса

9.3.5.1 Входящим значением **Version** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.3.5.2 Входящим значением **return** должно быть стандартно представленное целое число, возвращаемое собственной функцией.

9.3.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.4 Функция BioAPI_Terminate

9.4.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

BioAPI_RETURN BioAPI
BioAPI_Terminate (void);

и параметры, указанные в таблице 11.

Таблица 11

Параметр	Вызов функции (модель испытания на соответствие инфраструктур)	Вызов связанного процесса (модель испытания на соответствие приложений)
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.4.2 Ограничения на параметры

Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.4.3 Входные параметры вызова функции

Входные параметры отсутствуют.

9.4.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.4.5 Входные параметры вызова связанного процесса

Входящим значением **return** должно быть стандартно представленное целое число, возвращаемое собственной функцией.

9.4.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.5 Функция BioAPI_GetFrameworkInfo

9.5.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

BioAPI_RETURN BioAPI
BioAPI_GetFrameworkInfo
(BioAPI_FRAMEWORK_SCHEMA *FrameworkSchema);

и параметры, указанные в таблице 12.

Таблица 12

Параметр	Вызов функции (модель испытания на соответствие инфраструктур)	Вызов связанного процесса (модель испытания на соответствие приложений)
no_FrameworkSchema	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема структуры» (см. 9.2.9)	Входной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.5.2 Ограничения на параметры

9.5.2.1 Исходящим значением **no_FrameworkSchema** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.5.2.2 Исходящими значениями группы параметров «Схема структуры» должны быть корректно представленные значения собственного типа **BioAPI_FRAMEWORK_SCHEMA**. Входящими значениями должны быть стандартно представленные значения этого типа или набор пустых строк.

9.5.2.3 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.5.3 Входные параметры вызова функции

Если исходящее значение **no_FrameworkSchema** соответствует значению **"true"**, то собственному параметру **FrameworkSchema** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_FRAMEWORK_SCHEMA**.

9.5.4 Выходные параметры вызова функции

Входящие значения группы параметров «Схема структуры» определяют следующим образом. Если собственный параметр **FrameworkSchema** имеет NULL-значение, то входящими значениями должен быть набор пустых строк. В противном случае входящим значением должно быть стандартно представленное (см. 9.2.9.5) значение типа **BioAPI_FRAMEWORK_SCHEMA**, на которое указывает собственный параметр **FrameworkSchema**.

9.5.5 Входные параметры вызова связанного процесса

Входящим значением **no_FrameworkSchema** должно быть **"true"**, если собственный параметр **FrameworkSchema** имеет NULL-значение, в противном случае — **"false"**.

9.5.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.6 Функция BioAPI_EnumBSPs

9.6.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_EnumBSPs (
   BioAPI_BSP_SCHEMA **BSPSchemaArray,
   uint32_t *NumberOfElements);
```

и параметры, указанные в таблице 13.

Таблица 13

Параметр	Вызов функции (модель испытания на соответствие инфраструктур)	Вызов связанного процесса (модель испытания на соответствие приложений)
no_BSPSchemaArray	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема ПБУ» (см. 9.2.10) с именами, начинающимися с **"BSPSchema_1_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема ПБУ» (см. 9.2.10) с именами, начинающимися с **"BSPSchema_2_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема ПБУ» (см. 9.2.10) с именами, начинающимися с **"BSPSchema_3_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема ПБУ» (см. 9.2.10) с именами, начинающимися с **"BSPSchema_4_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
no_NumberOfElements	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumberOfElements	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.6.2 Ограничения на параметры

9.6.2.1 Исходящими значениями **no_BSPSchemaArray** и **no_NumberOfElements** должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.6.2.2 Исходящими значениями каждой группы параметров «Схема ПБУ» должны быть корректно представленные значения собственного типа **BioAPI_BSP_SCHEMA**. Входящими значениями должны быть стандартно представленные значения этого типа или наборы пустых строк.

9.6.2.3 Исходящим значением **NumberOfElements** должно быть или корректно представленное целое число в интервале от 0 до 4294967295, или пустая строка. Входящим значением должно быть или корректно представленное целое число в том же интервале, или пустая строка.

9.6.2.4 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.6.3 Входные параметры вызова функции

9.6.3.1 Если исходящее значение **no_BSPSchemaArray** соответствует значению **"true"**, то собственному параметру **BSPSchemaArray** должно быть присвоено NULL-значение, в противном случае ему должен быть присвоен адрес переменной типа **BioAPI_BSP_SCHEMA*** (указатель).

9.6.3.2 Если исходящее значение **no_NumberOfElements** соответствует значению **"true"**, то собственному параметру **NumberOfElements** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **uint32_t**.

9.6.4 Выходные параметры вызова функции

9.6.4.1 Входящие значения группы параметров «Схема ПБУ» с префиксом **"BSPSchema_X_"** (где **X** = 1, 2, 3 или 4) определяют следующим образом. Если собственный параметр **BSPSchemaArray** или переменная, на которую указывает этот собственный параметр, или собственный параметр **NumberOfElements** являются пустыми строками, или значение целочисленной переменной, на которую указывает этот собственный параметр, меньше **X**, то входящими значениями должны быть наборы пустых строк. В противном случае входящими значениями должны быть стандартно представленные (см. 9.2.10.5) значения типа **BioAPI_BSP_SCHEMA** в позиции **X** массива, на который указывает переменная, на которую, в свою очередь, указывает собственный параметр **BSPSchemaArray**.

9.6.4.2 Входящее значение **NumberOfElements** определяют следующим образом. Если собственный параметр с тем же именем является пустой строкой, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **uint32_t**, на которую указывает собственный параметр **NumberOfElements**.

9.6.5 Входные параметры вызова связанного процесса

9.6.5.1 Входящим значением **no_BSPSchemaArray** должно быть **"true"**, если собственный параметр **BSPSchemaArray** имеет NULL-значение, в противном случае — **"false"**.

9.6.5.2 Входящим значением **no_NumberOfElements** должно быть **"true"**, если собственный параметр **NumberOfElements** имеет NULL-значение, в противном случае — **"false"**.

9.6.5.3 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.6.5.4 Входящие значения остальных входных параметров определяют в соответствии с 9.6.4.

9.6.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.7 Функция BioAPI_BSPLoad

9.7.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_BSPLoad (
   const BioAPI_UUID *BSPUuid,
   BioAPI_EventHandler AppNotifyCallback,
   void* AppNotifyCallbackCtx);
```

и параметры, указанные в таблице 14.

Таблица 14

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
AppNotifyCallback	Входной параметр, исходящее значение	Входной параметр, входящее значение
AppNotifyCallbackCtx	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.7.2 Ограничения на параметры

9.7.2.1 Исходящим значением **BSPUuid** должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.7.2.2 Исходящими значениями **AppNotifyCallback** и **AppNotifyCallbackCtx** должны быть или «0», или «*». Входящими значениями должны быть стандартно представленные целые числа в интервале от 0 до 4294967295.

9.7.2.3 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.7.3 Входные параметры вызова функции

9.7.3.1 УУИД, представленный исходящим значением **BSPUuid**, должен быть записан в переменную типа **BioAPI_UUID**, адрес которой должен быть присвоен собственному параметру **BSPUuid**.

9.7.3.2 Если исходящее значение **AppNotifyCallBack** равно «0», то собственному параметру **AppNotifyCallback** должно быть присвоено NULL-значение указателя. Если исходящее значение равно «*», то значение указателя (отличающееся от NULL-значения), которое является адресом собственной функции (в рамках компонента тестирования), которая реализует функцию **BioAPI_EventHandler** стандартного интерфейса BioAPI, должно быть присвоено собственному параметру **AppNotifyCallback**.

Примечание — Если исходящее значение не равно «0», то любой из следующих друг за другом входящих вызовов в адрес функции **BioAPI_EventHandler** стандартного интерфейса BioAPI вызывает связанный с функцией процесс, если такая связь существует.

9.7.3.3 Если исходящее значение **AppNotifyCallbackCtx** равно «0», то собственному параметру **AppNotifyCallbackCtx** должно быть присвоено NULL-значение указателя. Если исходящее значение равно «*», то значение указателя (отличающееся от NULL-значения), которое является адресом переменной типа void*, имеющей NULL-значение, должно быть присвоено собственному параметру **AppNotifyCallbackCtx**.

9.7.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.7.5 Входные параметры вызова связанного процесса

9.7.5.1 Входящим значением **BSPUuid** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **BSPUuid**.

9.7.5.2 Входящими значениями **AppNotifyCallback** и **AppNotifyCallbackCtx** должны быть стандартно представленные целые числа в собственном параметре с тем же именем.

9.7.5.3 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.7.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.8 Функция BioAPI_BSPUnload

9.8.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_BSPUnload (
   const BioAPI_UUID *BSPUuid,
   BioAPI_EventHandler AppNotifyCallback,
   void* AppNotifyCallbackCtx);
```

и параметры, указанные в таблице 15.

Таблица 15

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
AppNotifyCallback	Входной параметр, исходящее значение	Входной параметр, входящее значение
AppNotifyCallbackCtx	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.8.2 Ограничения на параметры

9.8.2.1 Исходящим значением **BSPUuid** должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.8.2.2 Исходящими значениями **AppNotifyCallback** и **AppNotifyCallbackCtx** должны быть или «0», или «*». Входящими значениями должны быть стандартно представленные целые числа в интервале от 0 до 4294967295.

9.8.2.3 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.8.3 Входные параметры вызова функции

9.8.3.1 УУИД, представленный исходящим значением **BSPUuid**, должен быть записан в переменную типа **BioAPI_UUID**, адрес которой должен быть присвоен собственному параметру **BSPUuid**.

9.8.3.2 Если исходящее значение **AppNotifyCallback** равно «0», то собственному параметру **AppNotifyCallback** должно быть присвоено NULL-значение указателя. Если исходящее значение равно «*», то значение указателя (отличающееся от NULL-значения), которое является адресом собственной функции (в рамках компонента тестирования), которая реализует функцию **BioAPI_EventHandler** стандартного интерфейса **BioAPI**, должно быть присвоено собственному параметру **AppNotifyCallback**.

9.8.3.3 Если исходящее значение **AppNotifyCallbackCtx** равно «0», то собственному параметру **AppNotifyCallbackCtx** должно быть присвоено NULL-значение указателя. Если исходящее значение равно «*», то значение указателя (отличающееся от NULL-значения), которое является адресом переменной типа **void***, содержащей NULL-значение, должно быть присвоено собственному параметру **AppNotifyCallbackCtx**.

9.8.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.8.5 Выходные параметры вызова связанного процесса

9.8.5.1 Входящим значением **BSPUuid** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **BSPUuid**.

9.8.5.2 Входящими значениями **AppNotifyCallback** и **AppNotifyCallbackCtx** должны быть стандартно представленные целые числа в собственном параметре с тем же именем.

9.8.5.3 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.8.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.9 Функция BioAPI_BSPAttach

9.9.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_BSPAttach (
   const BioAPI_UUID *BSPUuid,
   BioAPI_VERSION Version,
   const BioAPI_UNIT_LIST_ELEMENT *UnitList,
   uint32_t NumUnits,
   BioAPI_HANDLE *NewBSPHandle);
```

и параметры, указанные в таблице 16.

Таблица 16

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
Version	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_1_UnitCategory	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_1_UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_2_UnitCategory	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_2_UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_3_UnitCategory	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_3_UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_4_UnitCategory	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_4_UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumUnit	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_NewBSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
NewBSPHandle	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.9.2 Ограничения на параметры

9.9.2.1 Исходящим значением **BSPUuid** должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.9.2.2 Исходящим значением **Version** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 255. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.9.2.3 Исходящими значениями **Unit_X_UnitCategory** и **Unit_X_UnitID** (где **X** = 1, 2, 3 или 4) должны быть или корректно представленные целые числа в интервале от 0 до 4294967295, или пустые строки. Входящими значениями должны быть или стандартно представленные целые числа в том же интервале, или пустые строки.

9.9.2.4 Исходящим значением **NumUnits** должно быть корректно представленное целое число от 0 до 4. Входящим значением должно быть стандартно представленное целое число в интервале от 0 до 4294967295.

9.9.2.5 Исходящим значением **no_NewBSPHandle** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.9.2.6 Исходящим значением **NewBSPHandle** должно быть или корректно представленное целое число в интервале от 0 до 4294967295, или пустая строка. Входящим значением должно быть или корректно представленное целое число в том же интервале, или пустая строка.

9.9.2.7 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.9.3 Входные параметры вызова функции

9.9.3.1 УУИД, представленный исходящим значением **BSPUuid**, должен быть записан в переменную типа **BioAPI_UUID**, адрес которой должен быть присвоен собственному параметру **BSPUuid**.

9.9.3.2 Целое число, представленное исходящим значением **Version** и **NumUnits**, должно быть присвоено собственному параметру с тем же именем.

9.9.3.3 Целые числа, представленные исходящими значениями **Unit_X_UnitCategory** и **Unit_X_UnitID** (где **X** = 1, 2, 3 или 4), или 0, если исходящее значение является пустой строкой, должны быть записаны в поля **UnitCategory** и **UnitID** (соответственно) элемента в позиции **X** массива четырех элементов типа **BioAPI_UNIT_LIST_ELEMENT**. Адрес этого массива должен быть присвоен собственному параметру **UnitList**.

9.9.3.4 Если исходящее значение **no_NewBSPHandle** имеет значение **"true"**, то собственному параметру **NewBSPHandle** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_HANDLE**.

9.9.4 Выходные параметры вызова функции

Входящее значение **NewBSPHandle** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_HANDLE**, на которую указывает собственный параметр **NewBSPHandle**.

9.9.5 Входные параметры вызова связанного процесса

9.9.5.1 Входящим значением **BSPUuid** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **BSPUuid**.

9.9.5.2 Входящими значениями **Version** и **NumUnits** должны быть стандартно представленные целые числа в собственном параметре с тем же именем.

9.9.5.3 Входящие значения **Unit_X_UnitCategory** и **Unit_X_UnitID** (где **X** = 1, 2, 3 или 4) определяют следующим образом. Если собственный параметр **NumUnits** меньше **X**, то входящими значениями должны быть пустые строки. В противном случае входящими значениями должны быть стандартно представленные целые числа в полях **UnitCategory** и **UnitID** (соответственно) элемента в позиции **X** массива элементов типа **BioAPI_UNIT_LIST_ELEMENT**, на который указывает собственный параметр **UnitList**.

9.9.5.4 Входящим значением **no_NewBSPHandle** должно быть **"true"**, если собственный параметр **NewBSPHandle** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.9.5.5 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.9.5.6 Входящие значения остальных входных параметров определяют в соответствии с 9.9.4.

9.9.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.10 Функция BioAPI_BSPDetach

9.10.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и включает в себя следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_BSPDetach (
   BioAPI_HANDLE BSPHandle);
```

и параметры, указанные в таблице 17.

Таблица 17

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.10.2 Ограничения на параметры

9.10.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.10.2.2 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.10.3 Входные параметры вызова функции

Целое число, представленное исходящим значением **BSPHandle**, должно быть присвоено собственному параметру с тем же именем.

9.10.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.10.5 Входные параметры вызова связанного процесса

9.10.5.1 Входящим значением **BSPHandle** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.10.5.2 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное встроенной функцией.

9.10.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.11 Функция BioAPI_QueryUnits

9.11.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_QueryUnits (
   const BioAPI_UUID *BSPUuid,
   BioAPI_UNIT_SCHEMA **UnitSchemaArray,
   uint32_t *NumberOfElements);
```

и параметры, указанные в таблице 18.

Таблица 18

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_UnitSchemaArray	Входной параметр, исходящее значение	Входной параметр, входящее значение

Окончание таблицы 18

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
Элементы группы параметров «Схема элементов» (см. 9.2.12) с именами, начинающимися с **"UnitSchema_1_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема элементов» (см. 9.2.12) с именами, начинающимися с **"UnitSchema_2_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема элементов» (см. 9.2.12) с именами, начинающимися с **"UnitSchema_3_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема элементов» (см. 9.2.12) с именами, начинающимися с **"UnitSchema_4_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
no_NumberOfElements	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumberOfElements	Выходной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.11.2 Ограничения на параметры

9.11.2.1 Исходящим значением **BSPUuid** должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.11.2.2 Исходящими значениями **no_UnitSchemaArray** и **no_NumberOfElements** должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.11.2.3 Исходящими значениями каждой группы параметров «Схема элементов» должны быть корректно представленные значения собственного типа **BioAPI_UNIT_SCHEMA**. Входящими значениями должны быть или стандартно представленные значения этого типа, или пустые строки.

9.11.2.4 Исходящим значением **NumberOfElements** должно быть или корректно представленное целое число в интервале от 0 до 4294967295, или пустая строка. Входящим значением должно быть или корректно представленное целое число в том же интервале, или пустая строка.

9.11.2.5 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.11.3 Входные параметры вызова функции

9.11.3.1 УУИД, представленный исходящим значением **BSPUuid**, должен быть записан в переменную типа **BioAPI_UUID**, адрес которой должен быть присвоен собственному параметру **BSPUuid**.

9.11.3.2 Если исходящее значение **no_UnitSchemaArray** соответствует значению **"true"**, то собственному параметру **UnitSchemaArray** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_UNIT_SCHEMA*** (указатель).

9.11.3.3 Если исходящее значение **no_NumberOfElements** соответствует значению **"true"**, то собственному параметру **NumberOfElements** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **uint32_t**.

9.11.4 Выходные параметры вызова функции

9.11.4.1 Входящие значения группы параметров «Схема элементов» с префиксом **"UnitSchema_X_"** (где **X** = 1, 2, 3 или 4) определяют следующим образом. Если собственный параметр **UnitSchemaArray** или переменная, на которую указывает этот собственный параметр, имеет NULL-значение, или собственный параметр **NumberOfElements** имеет NULL-значение, или значение целочисленной переменной, на которую указывает этот собственный параметр, меньше **X**, то входящими значениями должен быть набор пустых строк. В противном случае входящими значениями должны быть стандартно представленные (см. 9.2.14.5) значения типа **BioAPI_UNIT_SCHEMA** в позиции **X** массива, на который указывает переменная, на которую, в свою очередь, указывает собственный параметр **UnitSchemaArray**.

9.11.4.2 Входящее значение **NumberOfElements** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **uint32_t**, на которую указывает собственный параметр **NumberOfElements**.

9.11.5 Входные параметры вызова связанного процесса

9.11.5.1 Входящим значением **BSPUuid** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **BSPUuid**.

9.11.5.2 Входящим значением **no_UnitSchemaArray** должно быть **"true"**, если собственный параметр **UnitSchemaArray** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.11.5.3 Входящим значением **no_NumberOfElements** должно быть **"true"**, если собственный параметр **NumberOfElements** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.11.5.4 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.11.5.5 Входящие значения остальных входных параметров определяют в соответствии с 9.11.4.

9.11.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.12 Функция BioAPI_EnumBFPs

9.12.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_EnumBFPs (
  BioAPI_BFP_SCHEMA **UnitSchemaArray,
  uint32_t *NumberOfElements);
```

и параметры, указанные в таблице 19.

Таблица 19

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
no_BFPSchemaArray	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема ПБФ» (см. 9.2.11) с именами, начинающимися с **"BFPSchema_1_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема ПБФ» (см. 9.2.11) с именами, начинающимися с **"BFPSchema_2_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема ПБФ» (см. 9.2.11) с именами, начинающимися с **"BFPSchema_3_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема ПБФ» (см. 9.2.11) с именами, начинающимися с **"BFPSchema_4_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение

Окончание таблицы 19

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
no_NumberOfElements	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumberOfElements	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.12.2 Ограничения на параметры

9.12.2.1 Исходящими значениями **no_BFPSchemaArray** и **no_NumberOfElements** должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.12.2.2 Исходящими значениями каждой группы параметров «Схема ПБФ» должны быть корректно представленные значения собственного типа **BioAPI_BFP_SCHEMA**. Входящими значениями должны быть или стандартно представленные значения этого типа, или наборы пустых строк.

9.12.2.3 Исходящим значением **NumberOfElements** должно быть или корректно представленное целое число в интервале от 0 до 4294967295, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.12.2.4 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.12.3 Входные параметры вызова функции

9.12.3.1 Если исходящее значение **no_BFPSchemaArray** имеет значение **"true"**, то собственному параметру **BFPSchemaArray** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BFP_SCHEMA*** (указатель).

9.12.3.2 Если исходящее значение **no_NumberOfElements** имеет значение **"true"**, то собственному параметру **NumberOfElements** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **uint32_t**.

9.12.4 Выходные параметры вызова функции

9.12.4.1 Входящие значения группы параметров «Схема ПБФ» с префиксом **«BFPSchema_X_»** (где **X** =1, 2, 3 или 4) определяют следующим образом. Если собственный параметр **BFPSchemaArray** или переменная, на которую указывает этот собственный параметр, имеет NULL-значение или собственный параметр **NumberOfElements** имеет NULL-значение, или значение целочисленной переменной, на которую указывает этот собственный параметр, меньше **X**, то входящими значениями должен быть набор пустых строк. В противном случае входящими значениями должны быть стандартные представления (см. 9.2.11.5) значения типа **BioAPI_BFP_SCHEMA** в позиции **X** массива, на который указывает переменная, на которую, в свою очередь, указывает собственный параметр **BFPSchemaArray**.

9.12.4.2 Входящее значение **NumberOfElements** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **uint32_t**, на которую указывает собственный параметр **NumberOfElements**.

9.12.5 Входные параметры вызова связанного процесса

9.12.5.1 Входящим значением **no_BFPSchemaArray** должно быть **"true"**, если собственный параметр **BFPSchemaArray** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.12.5.2 Входящим значением **no_NumberOfElements** должно быть **"true"**, если собственный параметр **NumberOfElements** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.12.5.3 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное встроенной функцией.

9.12.5.4 Входящие значения других входных параметров определяют в соответствии с 9.12.4.

9.12.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.13 Функция BioAPI_QueryBFPs

9.13.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_QueryBFPs (
  const BioAPI_UUID *BSPUuid,
  BioAPI_BFP_LIST_ELEMENT **BFPList,
  uint32_t *NumberOfElements);
```

и параметры, указанные в таблице 20.

Таблица 20

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_BFPList	Входной параметр, исходящее значение	Входной параметр, входящее значение
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_BFPList	Входной параметр, исходящее значение	Входной параметр, входящее значение
BFP_1_BFPCategory	Выходной параметр, входящее значение	Входной параметр, входящее значение
BFP_1_BFPUuid	Выходной параметр, входящее значение	Входной параметр, входящее значение
BFP_2_BFPCategory	Выходной параметр, входящее значение	Входной параметр, входящее значение
BFP_2_BFPUuid	Выходной параметр, входящее значение	Входной параметр, входящее значение
BFP_3_BFPCategory	Выходной параметр, входящее значение	Входной параметр, входящее значение
BFP_3_BFPUuid	Выходной параметр, входящее значение	Входной параметр, входящее значение
BFP_4_BFPCategory	Выходной параметр, входящее значение	Входной параметр, входящее значение
BFP_4_BFPUuid	Выходной параметр, входящее значение	Входной параметр, входящее значение
no_NumberOfElements	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumberOfElements	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.13.2 Ограничения на параметры

9.13.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.13.2.2 Исходящими значениями **no_BFPList** и **no_NumberOfElements** должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.13.2.3 Исходящим значением **NumberOfElements** должно быть или корректно представленное целое число в интервале от 0 до 4294967295, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.13.2.4 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.12.3 Входные параметры вызова функции

9.13.3.1 Целое число, представленное исходящим значением **BSPHandle**, должно быть присвоено собственному параметру с тем же именем.

9.13.3.2 Если исходящее значение **no_BFPList** соответствует значению **"true"**, то собственному параметру **BFPList** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BFP_LIST_ELEMENT*** (указатель).

9.13.3.3 Если исходящее значение **no_NumberOfElements** соответствует значению **"true"**, то собственному параметру **NumberOfElements** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **uint32_t**.

9.13.4 Выходные параметры вызова функции

9.13.4.1 Входящие значения **BFP_X_BFPCategory** и **BFP_X_BFPUuid** (где **X** = 1, 2, 3 или 4) определяют следующим образом. Если собственный параметр **BFPList** имеет NULL-значение или переменная, на которую указывает этот собственный параметр, имеет NULL-значение, или собственный параметр **NumberOfElements** имеет NULL-значение, или значение целочисленной переменной, на которую указывает этот собственный параметр, меньше **X**, то входящими значениями должны быть пустые строки. В противном случае входящими значениями должны быть стандартно представленные целые числа в полях **BFPCategory** и **BFPUuid** (соответственно) элемента в позиции **X** массива элементов типа **BioAPI_BFP_LIST_ELEMENT**, на который указывает переменная, на которую, в свою очередь, указывает собственный параметр **BFPList**.

9.13.4.2 Входящее значение **NumberOfElements** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **uint32_t**, на которую указывает собственный параметр **NumberOfElements**.

9.13.5 Входные параметры вызова связанного процесса

9.13.5.1 Входящим значением **BSPHandle** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.13.5.2 Входящим значением **no_BFPList** должно быть значение **"true"**, если собственный параметр **BFPList** имеет NULL-значение, в противном случае входящим значением должно быть значение **"false"**.

9.13.5.3 Входящим значением **no_NumberOfElements** должно быть значение **"true"**, если собственный параметр **NumberOfElements** имеет NULL-значение, в противном случае входящим значением должно быть значение **"false"**.

9.13.5.4 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.13.5.5 Входящие значения остальных входных параметров определяют в соответствии с 9.13.4.

9.13.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.14 Функция BioAPI_ControlUnit

9.14.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_ControlUnit (
  BioAPI_HANDLE BSPHandle,
  BioAPI_UNIT_ID UnitID,
```

uint32_t ControlCode,
const BioAPI_DATA *InputData,
BioAPI_DATA *OutputData);
и параметры, указанные в таблице 21.

Таблица 21

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
ControlCode	Входной параметр, исходящее значение	Входной параметр, входящее значение
InputData	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_OutputData	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputData	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.14.2 Ограничения на параметры

9.14.2.1 Исходящими значениями **BSPHandle**, **UnitID** и **ControlCode** должны быть корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале.

9.14.2.2 Исходящими значениями **InputData** и **OutputData** должны быть корректно представленные наборы байтов (см. 7.7). Входящими значениями должны быть стандартно представленные наборы байтов (см. 7.7.2).

9.14.2.3 Исходящим значением **no_OutputData** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.14.2.4 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.14.3 Входные параметры вызова функции

9.14.3.1 Целые числа, представленные исходящими значениями **BSPHandle**, **UnitID** и **ControlCode**, должны быть присвоены собственному параметру с тем же именем.

9.14.3.2 Если исходящим значением **InputData** является пустая строка, то собственному параметру с тем же именем должно быть присвоено NULL-значение. В противном случае набор байтов, представленный исходящим значением, должен быть записан в блок памяти достаточного объема. Адрес и длина блока памяти должны быть записаны в поля **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, адрес которой должен быть присвоен собственному параметру **InputData**.

9.14.3.3 Если исходящим значением **no_OutputData** является значение **"true"**, то собственному параметру **OutputData** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_DATA**.

9.14.4 Выходные параметры вызова функции

Входящее значение **OutputData** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном

случае входящим значением должен быть стандартно представленный набор байтов в блоке памяти, адрес и длина которого находятся в полях **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, на которую указывает собственный параметр **OutputData**.

9.14.5 Входные параметры вызова связанного процесса

9.14.5.1 Входящими значениями **BSPHandle**, **UnitID** и **ControlCode** должны быть стандартно представленные целые числа в собственном параметре с тем же именем.

9.14.5.2 Исходящие значения **InputData** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должен быть стандартно представленный (см. 7.7.2) набор байтов в блоке памяти, адрес и длина которого находятся в полях **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, на которую указывает собственный параметр **InputData**.

9.14.5.3 Входящим значением **no_OutputData** должно быть значение **"true"**, если собственный параметр **OutputData** имеет NULL-значение, в противном случае входящим значением должно быть значение **"false"**.

9.14.5.4 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное встроенной функцией.

9.14.5.5 Входящие значения остальных входных параметров определяют в соответствии с 8.14.4.

9.14.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.15 Функция BioAPI_FreeBIRHandle

9.15.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_FreeBIRHandle (
  BioAPI_HANDLE BSPHandle,
  BioAPI_BIR_HANDLE Handle);
```

и параметры, указанные в таблице 22.

Таблица 22

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Handle	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.15.2 Ограничения на параметры

9.15.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.15.2.2 Исходящим значением **Handle** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.15.2.3 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.15.3 Входные параметры вызова функции

Целые числа, представленные исходящими значениями **BSPHandle** и **Handle**, должны быть присвоены собственным параметрам с теми же именами.

9.15.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.15.5 Входные параметры вызова связанного процесса

9.15.5.1 Входящими значениями **BSPHandle** и **Handle** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.15.5.2 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное встроенной функцией.

9.15.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.16 Функция BioAPI_GetBIRFromHandle

9.16.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_GetBIRFromHandle (
  BioAPI_HANDLE BSPHandle,
  BioAPI_BIR_HANDLE Handle,
  BioAPI_BIR *BIR);
```

и параметры, указанные в таблице 23.

Таблица 23

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Handle	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_BIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «ЗБИ» (см. 9.2.14)	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.16.2 Ограничения на параметры

9.16.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.16.2.2 Исходящим значением **Handle** должно быть или корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.16.2.3 Исходящим значением **no_BIR** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.16.2.4 Исходящими значениями группы параметров «ЗБИ» должны быть корректно представленные значения собственного типа **BioAPI_BIR**. Входящими значениями должны быть или стандартно представленные значения этого типа, или наборы пустых строк.

9.16.2.5 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.16.3 Входные параметры вызова функции

9.16.3.1 Целые числа, представленные исходящими значениями **BSPHandle** и **Handle**, должны быть присвоены собственным параметрам с теми же именами.

9.16.3.2 Если исходящим значением **no_BIR** является значение **"true"**, то собственному параметру **BIR** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR**.

9.16.4 Выходные параметры вызова функции

Входящие значения группы параметров «ЗБИ» определяют следующим образом. Если собственный параметр **BIR** имеет NULL-значение, то входящими значениями должны быть наборы пустых строк. В противном случае входящими значениями должны быть стандартно представленные (см. 9.2.14.5) значения типа **BioAPI_BIR**, на которые указывает собственный параметр **BIR**.

9.16.5 Входные параметры вызова связанного процесса

9.16.5.1 Входящими значениями **BSPHandle** и **Handle** должны быть стандартно представленные целые числа в собственных параметрах с тем же именем.

9.16.5.2 Входящим значением **no_BIR** должно быть значение **"true"**, если собственный параметр **BIR** имеет NULL-значение, в противном случае входящим значением должно быть значение **"false"**.

9.16.5.3 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.16.5.4 Входящие значения остальных входных параметров определяют в соответствии с 9.16.4.

9.16.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.17 Функция BioAPI_GetHeaderFromHandle

9.17.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_GetHeaderFromHandle (
  BioAPI_HANDLE BSPHandle,
  BioAPI_BIR_HANDLE Handle,
  BioAPI_BIR_HEADER *Header);
```

и параметры, указанные в таблице 24.

Таблица 24

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Handle	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Header	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Заголовок ЗБИ» (см. 9.2.13)	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.17.2 Ограничения на параметры

9.17.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.17.2.2 Исходящим значением **Handle** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.17.2.3 Исходящим значением **no_Header** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.17.2.4 Исходящими значениями группы параметров «Заголовок ЗБИ» должны быть корректно представленные значения собственного типа **BioAPI_BIR_HEADER**. Входящими значениями должны быть или стандартно представленные значения этого типа, или набор пустых строк.

9.17.2.5 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.17.3 Входные параметры вызова функции

9.17.3.1 Целые числа, представленные исходящими значениями **BSPHandle** и **Handle**, должны быть присвоены собственным параметрам с теми же именами.

9.17.3.2 Если исходящее значение **no_Header** соответствует значению **"true"**, то собственному параметру **Header** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HEADER**.

9.17.4 Выходные параметры вызова функции

Входящие значения параметров группы «Заголовок ЗБИ» определяют следующим образом. Если собственный параметр **Header** имеет NULL-значение, то входящими значениями должны быть наборы пустых строк. В противном случае входящими значениями должны быть стандартно представленные (см. 9.2.13.5) значения типа **BioAPI_BIR_Header**, на которые указывает собственный параметр **Header**.

9.17.5 Входные параметры вызова связанного процесса

9.17.5.1 Входящими значениями **BSPHandle** и **Handle** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.17.5.2 Входящее значение **no_Header** должно иметь значение **"true"**, если собственный параметр **Header** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.17.5.3 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.17.5.4 Входящие значения других входных параметров определяют в соответствии с 9.17.4.

9.17.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.18 Функция BioAPI_EnableEvents

9.18.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_EnableEvents (
   BioAPI_HANDLE BSPHandle,
   BioAPI_EVENT_MASK Events);
```

и параметры, указанные в таблице 25.

Таблица 25

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «События» (см. 9.2.4)	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.18.2 Ограничения на параметры

9.18.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.18.2.2 Исходящими значениями группы параметров «События» должны быть корректно представленные значения собственного типа **BioAPI_EVENT_MASK**. Входящими значениями должны быть стандартно представленные значения этого типа.

9.18.2.3 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.18.3 Входные параметры вызова функции

9.18.3.1 Целое число, представленное исходящим значением **BSPHandle**, должно быть присвоено собственному параметру с тем же именем.

9.18.3.2 Значение типа **BioAPI_EVENT_MASK**, представленное (см. 9.2.4.4) исходящим значением группы параметров «События», должно быть присвоено собственному параметру **Events**.

9.18.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.18.5 Входные параметры вызова связанного процесса

9.18.5.1 Входящим значением **BSPHandle** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.18.5.2 Входящими значениями группы параметров «События» должны быть стандартно представленные (см. 9.2.4.5) значения типа **BioAPI_EVENT_MASK** в собственном параметре **Events**.

9.18.5.3 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.18.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.19 Функция BioAPI_SetGUICallbacks

9.19.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_SetGUICallbacks (
  BioAPI_HANDLE BSPHandle,
  BioAPI_GUI_STREAMING_CALLBACK GuiStreamingCallback,
  void *GuiStreamingCallbackCtx,
  BioAPI_GUI_STATE_CALLBACK GuiStateCallback,
  void *GuiStateCallbackCtx);
```

и параметры, указанные в таблице 26.

Таблица 26

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
GuiStreamingCallback	Входной параметр, исходящее значение	Входной параметр, входящее значение
GuiStreamingCallbackCtx	Входной параметр, исходящее значение	Входной параметр, входящее значение
GuiStateCallback	Входной параметр, исходящее значение	Входной параметр, входящее значение
GuiStateCallbackCtx	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.19.2 Ограничения на параметры

9.19.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.19.2.2 Исходящими значениями **GuiStreamingCallback**, **GuiStreamingCallbackCtx**, **GuiStateCallback** и **GuiStateCallbackCtx** должны быть или «0», или «*». Входящими значениями должны быть стандартно представленные целые числа в интервале от 0 до 4294967295.

9.19.2.3 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.19.3 Входные параметры вызова функции

9.19.3.1 Целое число, представленное исходящим значением **BSPHandle**, должно быть присвоено собственному параметру с тем же именем.

9.19.3.2 Если исходящее значение **GuiStreamingCallback** равно нулю, то собственному параметру GuiStreamingCallback должно быть присвоено NULL-значение указателя. Если исходящее значение равно «*», то значение указателя (отличающееся от NULL-значения), которое является адресом собственной функции (в рамках компонента тестирования), которая реализует функцию **BioSPI_GUI_STREAMING_CALLBACK** стандартного интерфейса BioAPI, должно быть присвоено собственному параметру **GuiStreamingCallback**.

Примечание — Если исходящее значение **GuiStreamingCallback** не равно «0», то любой из следующих друг за другом входящих вызовов в адрес функции **BioSPI_GUI_STREAMING_CALLBACK** стандартного интерфейса BioAPI вызовет связанный с функцией процесс, если такая связь существует.

9.19.3.3 Если исходящее значение **GuiStreamingCallbackCtx** равно «0», то собственному параметру **GuiStreamingCallbackCtx** должно быть присвоено NULL-значение указателя. Если исходящее значение равно «*», то значение указателя (отличающееся от NULL-значения), которое является адресом переменной типа **void***, содержащей NULL-значение, должно быть присвоено собственному параметру **GuiStreamingCallbackCtx**.

9.19.3.4 Если исходящее значение **GuiStateCallback** равно «0», то собственному параметру **GuiStateCallback** должно быть присвоено NULL-значение указателя. Если исходящее значение равно «*», то значение указателя (отличающееся от NULL-значения), которое является адресом встроенной функции (в рамках компонента тестирования), которая реализует функцию **BioSPI_GUI_STATE_CALLBACK** стандартного интерфейса BioAPI, должно быть присвоено собственному параметру **GuiStateCallback**.

Примечание — Если исходящее значение **GuiStateCallback** не равно «0», то любой из следующих друг за другом входящих вызовов в адрес функции **BioSPI_GUI_STATE_CALLBACK** стандартного интерфейса BioAPI вызывает связанный с функцией процесс, если такая связь существует.

9.19.3.5 Если исходящее значение **GuiStateCallbackCtx** равно «0», то собственному параметру **GuiStateCallbackCtx** должно быть присвоено NULL-значение указателя. Если исходящее значение равно «*», то значение указателя (отличающееся от NULL-значения), которое является адресом переменной типа **void***, содержащей NULL-значение, должно быть присвоено собственному параметру **GuiStateCallbackCtx**.

9.19.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.19.5 Входные параметры вызова связанного процесса

9.19.5.1 Входящим значением **BSPHandle** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.19.5.2 Входящими значениями **GuiStreamingCallback**, **GuiStreamingCallbackCtx**, **GuiStateCallback** и **GuiStateCallbackCtx** должны быть стандартно представленные целые числа в собственных параметрах с тем же именем.

9.19.5.3 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное встроенной функцией.

9.19.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.20 Функция BioAPI_Capture

9.20.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_Capture (
  BioAPI_HANDLE BSPHandle,
  BioAPI_BIR_PURPOSE Purpose,
  BioAPI_BIR_SUBTYPE Subtype,
  const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat,
  BioAPI_BIR_HANDLE *CapturedBIR,
  int32_t Timeout,
  BioAPI_BIR_HANDLE *AuditData);
```

и параметры, указанные в таблице 27.

Таблица 27

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Purpose	Входной параметр, исходящее значение	Входной параметр, входящее значение
Subtype	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatOwner	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatType	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_CapturedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
Timeout	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_AuditData	Входной параметр, исходящее значение	Входной параметр, входящее значение
CapturedBIR	Выходной параметр, входящее значение	Входной параметр, входящее значение
AuditData	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.20.2 Ограничения на параметры

9.20.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.20.2.2 Исходящими значениями **Purpose** и **Subtype** должны быть корректно представленные целые числа в интервале от 0 до 255. Входящими значениями должны быть стандартно представленные целые числа в том же интервале.

9.20.2.3 Исходящими значениями **OutputFormatOwner** и **OutputFormatType** должны быть корректно представленные целые числа в интервале от 0 до 65535. Входящими значениями должны быть стандартно представленные целые числа в том же интервале.

9.20.2.4 Исходящими значениями **no_CapturedBIR** и **no_AuditData** должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.20.2.5 Исходящим значением **Timeout** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.20.2.6 Исходящими значениями **CapturedBIR** и **AuditData** должны быть корректно представленные целые числа в интервале от минус 2147483648 до плюс 2147483647 или пустые строки. Входящими значениями должны быть стандартно представленные целые числа в том же интервале или пустые строки.

9.20.2.7 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.20.3 Входные параметры вызова функции

9.20.3.1 Целые числа, представленные исходящими значениями **BSPHandle**, **Purpose**, **Subtype** и **Timeout**, должны быть присвоены собственным параметрам с теми же именами.

9.20.3.2 Целые числа, представленные исходящими значениями **OutputFormatOwner** и **OutputFormatType**, должны быть записаны в поля **FormatOwner** и **FormatType** (соответственно) переменной типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, адрес которой должен быть присвоен собственному параметру **OutputFormat**.

9.20.3.3 Если исходящее значение **no_CapturedBIR** соответствует значению **"true"**, то собственному параметру **CapturedBIR** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HANDLE**.

9.20.3.4 Если исходящее значение **no_AuditData** соответствует значению **"true"**, то собственному параметру **AuditData** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HANDLE**.

9.20.4 Выходные параметры вызова функции

9.20.4.1 Входящее значение **CapturedBIR** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **CapturedBIR**.

9.20.4.2 Входящее значение **AuditData** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **AuditData**.

9.20.5 Входные параметры вызова связанного процесса

9.20.5.1 Входящими значениями **BSPHandle**, **Purpose**, **Subtype** и **Timeout** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.20.5.2 Входящими значениями **OutputFormatOwner** и **OutputFormatType** должны быть стандартно представленные целые числа в полях **FormatOwner** и **FormatType** (соответственно) переменной типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, на которою указывает собственный параметр **OutputFormat**.

9.20.5.3 Входящим значением **no_CapturedBIR** должно быть значение **"true"**, если собственный параметр **CapturedBIR** имеет NULL-значение, в противном случае входящим значением должно быть значение **"false"**.

9.20.5.4 Входящим значением **no_AuditData** должно быть значение **"true"**, если собственный параметр **AuditData** имеет NULL-значение, в противном случае входящим значением должно быть значение **"false"**.

9.20.5.5 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное встроенной функцией.

9.20.5.6 Входящие значения остальных входных параметров определяют в соответствии с 9.20.4.

9.20.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.21 Функция BioAPI_CreateTemplate

9.21.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_CreateTemplate (
  BioAPI_HANDLE BSPHandle,
  const BioAPI_INPUT_BIR *CapturedBIR,
  const BioAPI_INPUT_BIR *ReferenceTemplate,
  const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat,
  BioAPI_BIR_HANDLE *NewTemplate,
  const BioAPI_DATA *Payload,
  BioAPI_UUID *TemplateUUID);
```

и параметры, указанные в таблице 28.

Таблица 28

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"CapturedBIR_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» с именами, начинающимися с **"ReferenceTemplate_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatOwner	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatType	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_NewTemplate	Входной параметр, исходящее значение	Входной параметр, входящее значение
Payload	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_TemplateUUID	Входной параметр, исходящее значение	Входной параметр, входящее значение
NewTemplate	Выходной параметр, входящее значение	Входной параметр, входящее значение
TemplateUUID	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.21.2 Ограничения на параметры

9.21.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.21.2.2 Исходящими значениями каждой группы параметров «Ввод ЗБИ» должны быть корректно представленные значения собственного типа **BioAPI_INPUT_BIR**. Входящими значениями должны быть стандартно представленные значения этого типа.

9.21.2.3 Исходящими значениями **OutputFormatOwner** и **OutputFormatType** должны быть корректно представленные целые числа в интервале от 0 до 65535. Входящими значениями должны быть стандартно представленные целые числа в том же интервале.

9.21.2.4 Исходящими значениями **no_NewTemplate** и **no_TemplateUUID** должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.21.2.5 Исходящим значением **Payload** должен быть корректно представленный набор байтов (см. 7.7). Входящим значением должен быть стандартно представленный набор байтов (см. 7.7.2).

9.21.2.6 Исходящим значением **NewTemplate** должно быть или корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.21.2.7 Исходящим значением **TemplateUUID** должен быть или корректно представленный УУИД (см. 7.6), или пустая строка. Входящим значением должен быть или стандартно представленный УУИД (см. 7.6.3), или пустая строка.

9.21.2.8 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.21.3 Входные параметры вызова функции

9.21.3.1 Целое число, представленное исходящим значением **BSPHandle**, должно быть присвоено собственному параметру с тем же именем.

9.21.3.2 Значение типа **BioAPI_INPUT_BIR**, представленное (см. 9.2.15.4) исходящим значением группы параметров «Ввод ЗБИ» с префиксом **"CapturedBIR_"**, должно быть записано в переменную типа **BioAPI_INPUT_BIR**, адрес которой должен быть присвоен собственному параметру **CapturedBIR**.

9.21.3.3 Значение типа **BioAPI_INPUT_BIR**, представленное (см. 9.2.15.4) исходящим значением группы параметров «Ввод ЗБИ» с префиксом **"ReferenceTemplate_"**, должно быть записано в переменную типа **BioAPI_INPUT_BIR**, адрес которой должен быть присвоен собственному параметру **ReferenceTemplate**.

9.21.3.4 Целые числа, представленные исходящими значениями **OutputFormatOwner** и **OutputFormatType**, должны быть записаны в поля **FormatOwner** и **FormatType** (соответственно) переменной типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, адрес которой должен быть присвоен собственному параметру **OutputFormat**.

9.21.3.5 Если исходящим значением **no_NewTemplate** является значение **"true"**, то собственному параметру **NewTemplate** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HANDLE**.

9.21.3.6 Если исходящим значением **Payload** является пустая строка, то собственному параметру **Payload** должно быть присвоено NULL-значение. В противном случае набор байтов, представленный исходящим значением, должен быть записан в блок памяти достаточного объема. Адрес и длина блока памяти должны быть записаны в поля **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, адрес которой должен быть присвоен собственному параметру **Payload**.

9.21.3.7 Если исходящим значение **no_TemplateUUID** является значение **"true"**, то собственному параметру **TemplateUUID** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_UUID**.

9.21.4 Выходные параметры вызова функции

9.21.4.1 Входящее значение **NewTemplate** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **NewTemplate**.

9.21.4.2 Входящее значение **TemplateUUID** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **TemplateUUID**.

9.21.5 Входные параметры вызова связанного процесса

9.21.5.1 Входящим значением **BSPHandle** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.21.5.2 Входящими значениями группы параметров «Ввод ЗБИ» с префиксом **"CapturedBIR_"** должны быть стандартно представленные (см. 9.2.15.5) значения типа **BioAPI_INPUT_BIR** в переменной, на которую указывает собственный параметр **CapturedBIR**.

9.21.5.3 Входящими значениями группы параметров «Ввод ЗБИ» с префиксом **"ReferenceTemplate_"** должны быть стандартно представленные (см. 9.2.15.5) значения типа **BioAPI_INPUT_BIR** в переменной, на которую указывает собственный параметр **ReferenceTemplate**.

9.21.5.4 Входящими значениями **OutputFormatOwner** и **OutputFormatType** должны быть стандартно представленные целые числа в полях **FormatOwner** и **FormatType** (соответственно) переменной типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, на которую указывает собственный параметр **OutputFormat**.

9.21.5.5 Входящим значением **no_NewTemplate** должно быть значение **"true"**, если собственный параметр **NewTemplate** имеет NULL-значение, в противном случае входящим значением должно быть значение **"false"**.

9.21.5.6 Входящее значение **Payload** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном

случае входящим значением должен быть стандартно представленный (см. 7.7.2) набор байтов в блоке памяти, адрес и длина которого находятся в полях **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, на которую указывает собственный параметр **Payload**.

9.21.5.7 Входящим значением **no_TemplateUUID** должно быть значению **"true"**, если собственный параметр **TemplateUUID** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.21.5.8 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное встроенной функцией.

9.21.5.9 Входящие значения остальных входных параметров определяют в соответствии с 9.21.4.

9.21.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.22 Функция BioAPI_Process

9.22.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_Process (
  BioAPI_HANDLE BSPHandle,
  const BioAPI_INPUT_BIR *CapturedBIR,
  const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat,
  BioAPI_BIR_HANDLE *ProcessedBIR);
```

и параметры, указанные в таблице 29.

Таблица 29

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"CapturedBIR_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatOwner	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatType	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_ProcessedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
ProcessedBIR	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.22.2 Ограничения на параметры

9.22.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.22.2.2 Исходящими значениями группы параметров «Ввод ЗБИ» должны быть корректно представленные значения собственного типа **BioAPI_INPUT_BIR**. Входящими значениями должны быть стандартно представленные значения этого типа.

9.22.2.3 Исходящими значениями **OutputFormatOwner** и **OutputFormatType** должны быть корректно представленные целые числа в интервале от 0 до 65535. Входящими значениями должны быть стандартно представленные целые числа в том же интервале.

9.22.2.4 Исходящим значением **no_ProcessedBIR** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.22.2.5 Исходящим значением **ProcessedBIR** должно быть или корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.22.2.6 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.22.3 Входные параметры вызова функции

9.22.3.1 Целое число, представленное исходящим значением **BSPHandle**, должно быть присвоено собственному параметру с тем же именем.

9.22.3.2 Значение типа **BioAPI_INPUT_BIR**, представленное (см. 9.2.15.4) исходящим значением группы параметров «Ввод ЗБИ», должно быть записано в переменную типа **BioAPI_INPUT_BIR**, адрес которой должен быть присвоен собственному параметру **CapturedBIR**.

9.22.3.3 Целые числа, представленные исходящими значениями **OutputFormatOwner** и **OutputFormatType**, должны быть записаны в поля **FormatOwner** и **FormatType** (соответственно) переменной типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, адрес которой должен быть присвоен собственному параметру **OutputFormat**.

9.22.3.4 Если исходящим значением **no_ProcessedBIR** является значение **"true"**, то собственному параметру **ProcessedBIR** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HANDLE**.

9.22.4 Выходные параметры вызова функции

Входящее значение **ProcessedBIR** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **ProcessedBIR**.

9.22.5 Входные параметры вызова связанного процесса

9.22.5.1 Входящим значением **BSPHandle** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.22.5.2 Входящими значениями группы параметров «Ввод ЗБИ» должны быть стандартно представленные (см. 9.2.15.5) значения типа **BioAPI_INPUT_BIR** в переменной, на которую указывает собственный параметр **CapturedBIR**.

9.22.5.3 Входящими значениями **OutputFormatOwner** и **OutputFormatType** должны быть стандартно представленные целые числа в полях **FormatOwner** и **FormatType** (соответственно) переменной типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, на которую указывает собственный параметр **OutputFormat**.

9.22.5.4 Входящим значением **no_ProcessedBIR** должно быть значение **"true"**, если собственный параметр **ProcessedBIR** содержит NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.22.5.5 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.22.5.6 Входящие значения других входных параметров определяют в соответствии с 9.22.4.

9.22.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.23 Функция BioAPI_ProcessWithAuxBIR

9.23.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_ProcessWithAuxBIR (
  BioAPI_HANDLE BSPHandle,
  const BioAPI_INPUT_BIR *CapturedBIR,
  const BioAPI_INPUT_BIR *AuxiliaryData,
  const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat,
  BioAPI_HANDLE *ProcessedBIR);
```

и параметры, указанные в таблице 30.

Таблица 30

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"CapturedBIR_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» с именами, начинающимися с **"AuxiliaryData_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatOwner	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatType	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_ProcessedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
ProcessedBIR	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.23.2 Ограничения на параметры

9.23.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.23.2.2 Исходящими значениями каждой группы параметров «Ввод ЗБИ» должны быть корректно представленные значения собственного типа **BioAPI_INPUT_BIR**. Входящими значениями должны быть стандартно представленные значения этого типа.

9.23.2.3 Исходящими значениями **OutputFormatOwner** и **OutputFormatType** должны быть корректно представленные целые числа в интервале от 0 до 65535. Входящими значениями должны быть стандартно представленные целые числа в том же интервале.

9.23.2.4 Исходящим значением **no_ProcessedBIR** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.23.2.5 Исходящим значением **ProcessedBIR** должно быть или корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.23.2.6 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.23.3 Входные параметры вызова функции

9.23.3.1 Целое число, представленное исходящим значением **BSPHandle**, должно быть присвоено собственному параметру с тем же именем.

9.23.3.2 Значение типа **BioAPI_INPUT_BIR**, представленное (см. 9.2.15.4) исходящим значением группы параметров «Ввод ЗБИ» с префиксом **"CapturedBIR_"**, должно быть записано в переменную типа **BioAPI_INPUT_BIR**, адрес которой должен быть присвоен собственному параметру **CapturedBIR**.

9.23.3.3 Значение типа **BioAPI_INPUT_BIR**, представленное (см. 9.2.15.4) исходящим значением группы параметров «Ввод ЗБИ» с префиксом **"AuxiliaryData_"**, должно быть записано в переменную типа **BioAPI_INPUT_BIR**, адрес которой должен быть присвоен собственному параметру **AuxiliaryData**.

9.23.3.4 Целые числа, представленные исходящими значениями **OutputFormatOwner** и **OutputFormatType** должны быть записаны в поля **FormatOwner** и **FormatType** (соответственно) переменной типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, адрес которой должен быть присвоен собственному параметру **OutputFormat**.

9.23.3.5 Если исходящим значением **no_ProcessedBIR** является значение **"true"**, то собственному параметру **ProcessedBIR** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HANDLE**.

9.23.4 Выходные параметры вызова функции

Входящее значение **ProcessedBIR** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **ProcessedBIR**.

9.23.5 Входные параметры вызова связанного процесса

9.23.5.1 Входящим значением **BSPHandle** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.23.5.2 Входящими значениями группы параметров «Ввод ЗБИ» с префиксом **"CapturedBIR_"** должны быть стандартно представленные (см. 9.2.15.5) значения типа **BioAPI_INPUT_BIR** в переменной, на которую указывает собственный параметр **CapturedBIR**.

9.23.5.3 Входящими значениями группы параметров «Ввод ЗБИ» с префиксом **"AuxiliaryData_"** должны быть стандартно представленные (см. 9.2.15.5) значения типа **BioAPI_INPUT_BIR** в переменной, на которую указывает собственный параметр **AuxiliaryData**.

9.23.5.4 Входящими значениями **OutputFormatOwner** и **OutputFormatType** должны быть стандартно представленные целые числа в полях **FormatOwner** и **FormatType** (соответственно) переменной типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, на которую указывает собственный параметр **OutputFormat**.

9.23.5.5 Входящим значением **no_ProcessedBIR** должно быть значение **"true"**, если собственный параметр **ProcessedBIR** иметт NULL-значение, в противном случае входящее значение должно быть **"false"**.

9.23.5.6 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное встроенной функцией.

9.23.5.7 Входящие значения остальных входных параметров определяют в соответствии с 9.23.4.

9.23.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.24 Функция BioAPI_VerifyMatch

9.24.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_VerifyMatch (
  BioAPI_HANDLE BSPHandle,
  BioAPI_FMR MaxFMRRequested,
  const BioAPI_INPUT_BIR *ProcessedBIR,
  const BioAPI_INPUT_BIR *ReferenceTemplate,
  BioAPI_BIR_HANDLE *AdaptedBIR,
  BioAPI_BOOL *Result,
  BioAPI_FMR *FMRAchieved,
  BioAPI_DATA *Payload);
```

и параметры, указанные в таблице 31.

Таблица 31

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение

Окончание таблицы 31

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
MaxFMRRequested	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"ProcessedBIR_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» с именами, начинающимися с **"ReferenceTemplate_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_AdaptedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Result	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_FMRAchieved	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Payload	Входной параметр, исходящее значение	Входной параметр, входящее значение
AdaptedBIR	Выходной параметр, входящее значение	Входной параметр, входящее значение
Result	Выходной параметр, входящее значение	Входной параметр, входящее значение
FMRAchieved	Выходной параметр, входящее значение	Входной параметр, входящее значение
Payload	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.24.2 Ограничения на параметры

9.24.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.24.2.2 Исходящим значением **MaxFMRRequested** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.24.2.3 Исходящими значениями каждой группы параметров «Ввод ЗБИ» должны быть корректно представленные значения собственного типа **BioAPI_INPUT_BIR**. Входящими значениями должны быть стандартно представленные значения этого типа.

9.24.2.4 Исходящим значением **no_Result** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.24.2.5 Исходящими значениями **AdaptedBIR** и **FMRAchieved** должны быть или корректно представленные целые числа в интервале от минус 2147483648 до плюс 2147483647, или пустые строки. Входящими значениями должны быть или стандартно представленные целые числа в том же интервале, или пустые строки.

9.24.2.6 Исходящим значением **Result** должно быть или корректно представленное логическое значение, или пустая строка. Входящим значением должно быть или стандартно представленное логическое значение, или пустая строка.

9.24.2.7 Исходящим значением **Payload** должен быть корректно представленный набор байтов. Входящим значением должен быть стандартно представленный набор байтов.

9.24.2.8 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.24.3 Входные параметры вызова функции

9.24.3.1 Целые числа, представленные исходящими значениями **BSPHandle** и **MaxFMRRequested**, должны быть присвоены собственным параметрам с тем же именем.

9.24.3.2 Значение типа **BioAPI_INPUT_BIR**, представленное (см. 9.2.15.4) исходящим значением группы параметров «Ввод ЗБИ» с префиксом **"ProcessedBIR_"**, должно быть записано в переменную типа **BioAPI_INPUT_BIR**, адрес которой должен быть присвоен собственному параметру **ProcessedBIR**.

9.24.3.3 Значение типа **BioAPI_INPUT_BIR**, представленное (см. 9.2.15.4) исходящим значением группы параметров «Ввод ЗБИ» с префиксом **"ReferenceTemplate_"**, должно быть записано в переменную типа **BioAPI_INPUT_BIR**, адрес которой должен быть присвоен собственному параметру **ReferenceTemplate**.

9.24.3.4 Если исходящим значением **no_AdaptedBIR** является значение **"true"**, то собственному параметру **AdaptedBIR** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HANDLE**.

9.24.3.5 Если исходящим значением **no_Result** является значение **"true"**, то собственному параметру **Result** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BOOL**.

9.24.3.6 Если исходящим значением **no_FMRAchieved** является значение **"true"**, то собственному параметру **FMRAchieved** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_FMR**.

9.24.3.7 Если исходящим значением **no_Payload** является значение **"true"**, то собственному параметр **Payload** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_DATA**.

9.24.4 Выходные параметры вызова функции

9.24.4.1 Входящее значение **AdaptedBIR** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **AdaptedBIR**.

9.24.4.2 Входящее значение **FMRAchieved** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_FMR**, на которую указывает собственный параметр **FMRAchieved**.

9.24.4.3 Входящее значение **Result** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное логическое значение в переменной типа **BioAPI_BOOL**, на которую указывает собственный параметр **Result**.

9.24.4.4 Входящее значение **Payload** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должен быть стандартно представленный набор байтов в блоке памяти, адрес и длина которого находятся в полях **Data** и **Length** (соответственно) в переменной типа **BioAPI_DATA**, на которую указывает собственный параметр **Payload**.

9.24.5 Входные параметры вызова связанного процесса

9.24.5.1 Входящими значениями **BSPHandle** и **MaxFMRRequested** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.24.5.2 Входящим значением группы параметров «Ввод ЗБИ» с префиксом **"ProcessedBIR_"** должно быть стандартно представленное (см. 9.2.15.5) значение типа **BioAPI_INPUT_BIR** в переменной, на которую указывает собственный параметр **ProcessedBIR**.

9.24.5.3 Входящим значением группы параметров «Ввод ЗБИ» с префиксом **"ReferenceTemplate_"** должно быть стандартно представленное (см. 9.2.15.5) значение типа **BioAPI_INPUT_BIR** в переменной, на которую указывает собственный параметр **ReferenceTemplate**.

9.24.5.4 Входящим значением **no_AdaptedBIR** должно быть значение **"true"**, если собственный параметр **AdaptedBIR** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.24.5.5 Входящим значением **no_Result** должно быть значение **"true"**, если собственный параметр **Result** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.24.5.6 Входящим значением **no_FMRAchieved** должно быть значение **"true"**, если собственный параметр **FMRAchieved** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.24.5.7 Входящим значением **no_Payload** должно быть значение **"true"**, если собственный параметр **Payload** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.24.5.8 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.24.5.9 Входящие значения остальных входных параметров определяют в соответствии с 9.24.4.

9.24.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.25 Функция BioAPI_IdentifyMatch

9.25.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_IdentifyMatch (
  BioAPI_HANDLE BSPHandle,
  BioAPI_FMR MaxFMRRequested,
  const BioAPI_INPUT_BIR *ProcessedBIR,
  const BioAPI_IDENTIFY_POPULATION *Population,
  uint32_t TotalNumberOfTemplates,
  BioAPI_BOOL Binning,
  uint32_t MaxNumberOfResults,
  uint32_t *NumberOfResults,
  BioAPI_CANDIDATE **Candidates,
  int32_t Timeout);
```

и параметры, указанные в таблице 32.

Таблица 32

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
MaxFMRRequested	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"ProcessedBIR_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Определение заполнения» (см. 9.2.16)	Входной параметр, исходящее значение	Входной параметр, входящее значение
TotalNumberOf Templates	Входной параметр, исходящее значение	Входной параметр, входящее значение
Binning	Входной параметр, исходящее значение	Входной параметр, входящее значение
MaxNumberOfResults	Входной параметр, исходящее значение	Входной параметр, входящее значение

Окончание таблицы 32

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
no_NumberOfResults	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Candidates	Входной параметр, исходящее значение	Входной параметр, входящее значение
Timeout	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumberOfResult	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Кандидат» (см. 9.2.17) с именами, начинающимися с **"Candidate_1_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Кандидат» с именами, начинающимися с **"Candidate_2_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Кандидат» с именами, начинающимися с **"Candidate_3_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Кандидат» с именами, начинающимися с **"Candidate_4_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.25.2 Ограничения на параметры

9.25.2.1 Исходящими значениями **BSPHandle**, **TotalNumberOfTemplates** и **MaxNumberOfResults** должны быть корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале.

9.25.2.2 Исходящим значением **MaxFMRRequested** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.25.2.3 Исходящим значением группы параметров «Ввод ЗБИ» должно быть корректно представленное значение собственного типа **BioAPI_INPUT_BIR**. Входящим значением должно быть стандартно представленное значение этого типа.

9.25.2.4 Исходящим значением группы параметров «Определение заполнения» должно быть корректно представленное значение собственного типа **BioAPI_IDENTIFY_POPULATION**. Входящим значением должно быть стандартно представленное значение этого типа.

9.25.2.5 Исходящим значением **Binning** должно быть корректно представленное логическое значение. Входящим значением должно быть стандартно представленное логическое значение.

9.25.2.6 Исходящими значениями **no_NumberOfResults** и **no_Candidates** должны быть или корректно представленные логические значения, или пустые строки. Входящими значениями должны быть стандартно представленные логические значения.

9.25.2.7 Исходящим значением **Timeout** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.25.2.8 Исходящим значением **NumberOfResults** должно быть или корректно представленное целое число в интервале от 0 до 4294967295, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.25.2.9 Исходящим значением каждой группы параметров «Кандидат» должно быть корректно представленное значение собственного типа **BioAPI_CANDIDATE**. Входящим значением должно быть или стандартно представленное значение этого типа, или набор пустых строк.

9.25.2.10 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.25.3 Входные параметры вызова функции

9.25.3.1 Целые числа, представленные исходящими значениями **BSPHandle**, **MaxFMRRequested**, **TotalNumberOfTemplates**, **MaxNumberOfResults** и **Timeout**, должны быть присвоены собственным параметрам с тем же именем.

9.25.3.2 Логическое значение, представленное исходящим значением **Binning**, должно быть присвоено собственному параметру с тем же именем.

9.25.3.3 Значение типа **BioAPI_INPUT_BIR**, представленное (см. 9.2.15.4) исходящим значением группы параметров «Ввод ЗБИ», должно быть записано в переменную типа **BioAPI_INPUT_BIR**, адрес которой должен быть присвоен собственному параметру **ProcessedBIR**.

9.25.3.4 Значение типа **BioAPI_IDENTIFY_POPULATION**, представленное (см. 9.2.16.4) исходящим значением группы параметров «Определение заполнения», должно быть записано в переменную типа **BioAPI_ IDENTIFY_POPULATION**, адрес которой должен быть присвоен собственному параметру **Population**.

9.25.3.5 Если исходящим значением **no_NumberOfResults** является **"true"**, то собственному параметру **NumberOfResults** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **uint32_t**.

9.25.3.6 Если исходящим значением **no_Candidates** является **"true"**, то собственному параметру **Candidates** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_Candidate*** (указатель).

9.25.4 Выходные параметры вызова функции

9.25.4.1 Входящее значение **NumberOfResults** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **uint32_t**, на которую указывает собственный параметр **NumberOfResults**.

9.25.4.2 Входящее значение группы параметров «Кандидат» с префиксом **"Candidate_X_"** (где **X** = 1, 2, 3 или 4) определяют следующим образом. Если собственный параметр **Candidates** имеет NULL-значение или переменная, на которую указывает этот собственный параметр, имеет NULL значение, или собственный параметр **NumberOfResults** имеет NULL-значение, или значение целочисленной переменной, на которую указывает этот собственный параметр, меньше **X**, то входящим значением должен быть набор пустых строк. В противном случае входящим значением должно быть стандартно представленное (см. 9.2.14.5) значение типа **BioAPI_CANDIDATE** в позиции **X** массива, на который указывает переменная, на которую, в свою очередь, указывает собственный параметр **Candidates**.

9.25.5 Входные параметры вызова связанного процесса

9.25.5.1 Входящими значениями **BSPHandle**, **MaxFMRRequested**, **TotalNumberOfTemplates**, **MaxNumberOfResults** и **Timeout** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.25.5.2 Входящим значением **Binning** должно быть стандартно представленное логическое значение в собственном параметре с тем же именем.

9.25.5.3 Входящим значением группы параметров «Ввод ЗБИ» должно быть стандартно представленное (см. 9.2.15.5) значение типа **BioAPI_INPUT_BIR** в переменной, на которую указывает собственный параметр **ProcessedBIR**.

9.25.5.4 Входящим значением группы параметров «Определение заполнения» должно быть стандартно представленное (см. 9.2.16.5) значение типа **BioAPI_IDENTIFY_POPULATION** в переменной, на которую указывает собственный параметр **Population**.

9.25.5.5 Входящим значением **no_NumberOfResults** должно быть **"true"**, если собственный параметр **NumberOfResults** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.25.5.6 Входящим значением **no_Candidates** должно быть **"true"**, если собственный параметр **Candidates** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.25.5.7 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.25.5.8 Входящие значения остальных входных параметров определяют в соответствии с 9.25.4.

9.25.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.26 Функция BioAPI_Enroll

9.26.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_Enroll (
  BioAPI_HANDLE BSPHandle,
  BioAPI_BIR_PURPOSE Purpose,
  BioAPI_BIR_SUBTYPE Subtype,
  const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat,
  const BioAPI_INPUT_BIR *ReferenceTemplate,
  BioAPI_BIR_HANDLE *NewTemplate,
  const BioAPI_DATA *Payload,
  int32_t Timeout,
  BioAPI_BIR_HANDLE *AuditData,
  BioAPI_UUID *TemplateUUID);
```

и параметры, указанные в таблице 33.

Таблица 33

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Purpose	Входной параметр, исходящее значение	Входной параметр, входящее значение
Subtype	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatOwner	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatType	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"ReferenceTemplate_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_NewTemplate	Входной параметр, исходящее значение	Входной параметр, входящее значение
Payload	Входной параметр, исходящее значение	Входной параметр, входящее значение
Timeout	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_AuditData	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_TemplateUUID	Входной параметр, исходящее значение	Входной параметр, входящее значение
NewTemplate	Выходной параметр, входящее значение	Входной параметр, входящее значение

Окончание таблицы 33

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
AuditData	Выходной параметр, входящее значение	Входной параметр, входящее значение
TemplateUUID	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.26.2 Ограничения на параметры

9.26.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.26.2.2 Исходящими значением **Purpose** и **Subtype** должны быть корректно представленные целые числа в интервале от 0 до 255. Входящими значениями должны быть стандартно представленные целые числа в том же интервале.

9.26.2.3 Исходящими значениями **OutputFormatOwner** и **OutputFormatType** должны быть корректно представленные целые числа в интервале от 0 до 65535. Входящими значениями должны быть стандартно представленные целые числа в том же интервале.

9.26.2.4 Исходящим значением группы параметров «Ввод ЗБИ» должно быть корректно представленное значение собственного типа **BioAPI_INPUT_BIR**. Входящим значением должно быть стандартно представленное значение этого типа.

9.26.2.5 Исходящими значениями **no_NewTemplate**, **no_AuditData** и **no_TemplateUUID** должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.26.2.6 Исходящим значением **Payload** должен быть корректно представленный набор байтов (см. 7.7). Входящим значением должен быть стандартно представленный набор байтов (см. 7.7.2).

9.26.2.7 Исходящим значением **Timeout** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.26.2.8 Исходящими значениями **NewTemplate** и **AuditData** должны быть корректно представленные целые числа в интервале от минус 2147483648 до плюс 2147483647. Входящими значениями должны быть или стандартно представленные целые числа в том же интервале, или пустые строки.

9.26.2.9 Исходящим значением **TemplateUUID** должен быть или корректно представленный УУИД (см. 7.6), или пустая строка. Входящим значением должен быть или стандартно представленный УУИД (см. 7.6.3), или пустая строка.

9.26.2.10 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.26.3 Входные параметры вызова функции

9.26.3.1 Целые числа, представленные исходящими значениями **BSPHandle**, **Purpose**, **Subtype** и **Timeout**, должны быть присвоены собственному параметру с тем же именем.

9.26.3.2 Целые числа, представленные исходящими значениями **OutputFormatOwner** и **OutputFormatType**, должны быть записаны в поля **FormatOwner** и **FormatType** (соответственно) переменной типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, адрес которой должен быть присвоен собственному параметру **OutputFormat**.

9.26.3.3 Значение типа **BioAPI_INPUT_BIR**, представленное (см. 9.2.15.4) исходящим значением группы параметров «Ввод ЗБИ», должно быть записано в переменную типа **BioAPI_INPUT_BIR**, адрес которой должен быть присвоен собственному параметру **ReferenceTemplate**.

9.26.3.4 Если исходящим значением **no_NewTemplate** является **"true"**, то собственному параметру **NewTemplate** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HANDLE**.

9.26.3.5 Если исходящим значением **Payload** является пустая строка, то собственному параметру с тем же именем должно быть присвоено NULL-значение. В противном случае набор байтов, представленный исходящим значением, должен быть записан в блок памяти достаточного объема. Адрес и длина блока памяти должны быть записаны в поля **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, адрес которой должен быть присвоен собственному параметру **Payload**.

9.26.3.6 Если исходящим значением **no_AuditData** является **"true"**, то собственному параметру **AuditData** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HANDLE**.

9.26.3.7 Если исходящим значением **no_TemplateUUID** является значение **"true"**, то собственному параметру **TemplateUUID** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_UUID**.

9.26.4 Выходные параметры вызова функции

9.26.4.1 Входящее значение **NewTemplate** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **NewTemplate**.

9.26.4.2 Входящее значение **AuditData** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **AuditData**.

9.26.4.3 Входящее значение **TemplateUUID** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **TemplateUUID**.

9.26.5 Входные параметры вызова связанного процесса

9.26.5.1 Входящими значениями **BSPHandle**, **Purpose**, **Subtype** и **Timeout** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.26.5.2 Входящими значениями **OutputFormatOwner** и **OutputFormatType** должны быть стандартно представленные целые числа в полях **FormatOwner** и **FormatType** (соответственно) переменной типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, на которую указывает собственный параметр **OutputFormat**.

9.26.5.3 Входящим значением группы параметров «Ввод ЗБИ» с префиксом **"ReferenceTemplate_"** должно быть стандартно представленное (см. 9.2.15.5) значение типа **BioAPI_INPUT_BIR** в переменной, на которую указывает собственный параметр **ReferenceTemplate**.

9.26.5.4 Входящим значением **no_NewTemplate** должно быть **"true"**, если собственный параметр **NewTemplate** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.26.5.5 Входящее значение **Payload** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должен быть стандартно представленный (см. 7.7.2) набор байтов в блоке памяти, адрес и длина которого находятся в полях **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, на которую указывает собственный параметр **Payload**.

9.26.5.6 Входящим значением **no_AuditData** должно быть **"true"**, если собственный параметр **AuditData** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.26.5.7 Входящим значением **no_TemplateUUID** должно быть **"true"**, если собственный параметр **TemplateUUID** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.26.5.8 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное встроенной функцией.

9.26.5.9 Входящие значения остальных входных параметров определяют в соответствии с 9.26.4.

9.26.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.27 Функция BioAPI_Verify

9.27.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_Verify (
  BioAPI_HANDLE BSPHandle,
  BioAPI_FMR MaxFMRRequested,
```

const BioAPI_INPUT_BIR *ReferenceTemplate,
BioAPI_BIR_SUBTYPE Subtype,
BioAPI_BIR_HANDLE *AdaptedBIR,
BioAPI_BOOL *Result,
BioAPI_FMR *FMRAchieved,
BioAPI_DATA *Payload,
int32_t Timeout,
BioAPI_BIR_HANDLE *AuditData);

и параметры, указанные в таблице 34.

Таблица 34

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
MaxFMRRequested	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"ReferenceTemplate_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
Subtype	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_AdaptedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Result	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_FMRAchieved	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Payload	Входной параметр, исходящее значение	Входной параметр, входящее значение
Timeout	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_AuditData	Входной параметр, исходящее значение	Входной параметр, входящее значение
AdaptedBIR	Выходной параметр, входящее значение	Входной параметр, входящее значение
Result	Выходной параметр, входящее значение	Входной параметр, входящее значение
FMRAchieved	Выходной параметр, входящее значение	Входной параметр, входящее значение
Payload	Выходной параметр, входящее значение	Входной параметр, входящее значение
AuditData	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.27.2 Ограничения на параметры

9.27.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.27.2.2 Исходящим значением **MaxFMRRequested** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.27.2.3 Исходящим значением группы параметров «Ввод ЗБИ» должно быть корректно представленное значение собственного типа **BioAPI_INPUT_BIR**. Входящим значением должно быть стандартно представленное значение этого типа.

9.27.2.4 Исходящими значениями **no_AdaptedBIR**, **no_Result**, **no_FMRAchieved**, **no_Payload** и **no_AuditData** должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.27.2.5 Исходящим значением **Subtype** должно быть корректно представленное целое число в интервале от 0 до 255. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.27.2.6 Исходящим значение **Timeout** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.27.2.7 Исходящими значениями **AdaptedBIR**, **FMRAchieved** и **AuditData** должны быть или корректно представленные целые числа в интервале от минус 2147483648 до плюс 2147483647, или пустые строки. Входящими значениями должны быть или стандартно представленные целые числа в том же интервале, или пустые строки.

9.27.2.8 Исходящим значением **Result** должно быть или корректно представленное логическое значение, или пустая строка. Входящим значением должно быть или стандартно представленное логическое значение, или пустая строка.

9.27.2.9 Исходящим значением **Payload** должен быть корректно представленный набор байтов. Входящим значением должен быть стандартно представленный набор байтов.

9.27.2.10 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.27.3 Входные параметры вызова функции

9.27.3.1 Целые числа, представленные исходящими значениями **BSPHandle**, **MaxFMRRequested**, **Subtype** и **Timeout**, должны быть присвоены собственным параметрам с теми же именами.

9.27.3.2 Значение типа **BioAPI_INPUT_BIR**, представленное (см. 9.2.15.4) исходящим значением группы параметров «Ввод ЗБИ», должно быть записано в переменную типа **BioAPI_INPUT_BIR**, адрес которой должен быть присвоен собственному параметру **ReferenceTemplate**.

9.27.3.3 Если исходящее значение **no_AdaptedBIR** соответствует значению **"true"**, то собственному параметру **AdaptedBIR** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HANDLE**.

9.27.3.4 Если исходящим значением **no_Result** является **"true"**, то собственному параметру **Result** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BOOL**.

9.27.3.5 Если исходящим значением **no_FMRAchieved** является **"true"**, то собственному параметру **FMRAchieved** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_FMR**.

9.27.3.6 Если исходящим значением **no_Payload** является **"true"**, то собственному параметру **Payload** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_DATA.**

9.27.3.7 Если исходящим значением **no_AuditData** является **"true"**, то собственному параметру **AuditData** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HANDLE**.

9.27.4 Выходные параметры вызова функции

9.27.4.1 Входящее значение **AdaptedBIR** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **AdaptedBIR**.

9.27.4.2 Входящее значение **FMRAchieved** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_FMR**, на которую указывает собственный параметр **FMRAchieved**.

9.27.4.3 Входящее значение **Result** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное логическое значение в переменной типа **BioAPI_BOOL**, на которую указывает собственный параметр **Result**.

9.27.4.4 Входящее значение **Payload** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должен быть стандартно представленный набор байтов в блоке памяти, адрес и длина которого находятся в полях **Data** и **Length** (соответственно) в переменной типа **BioAPI_DATA**, на которую указывает собственный параметр **Payload**.

9.27.4.5 Входящее значение **AuditData** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **AuditData**.

9.27.5 Входные параметры вызова связанного процесса

9.27.5.1 Входящими значениями **BSPHandle**, **MaxFMRRequesied**, **Subtype** и **Timeout** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.27.5.2 Входящими значениями группы параметров «Ввод ЗБИ» с префиксом **"ReferenceTemplate_"** должны быть стандартно представленные (см. 9.2.15.5) значения типа **BioAPI_INPUT_BIR** в переменной, на которую указывает собственный параметр **ReferenceTemplate**.

9.27.5.3 Входящим значением **no_AdaptedBIR** должно быть **"true"**, если собственный параметр **AdaptedBIR** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.27.5.4 Входящим значением **no_Result** должно быть **"true"**, если собственный параметр **Result** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.27.5.5 Входящим значением **no_FMRAchieved** должно быть **"true"**, если собственный параметр **FMRAchieved** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.27.5.6 Входящим значением **no_Payload** должно быть **"true"**, если собственный параметр **Payload** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.27.5.7 Входящим значением **no_AuditData** должно быть **"true"**, если собственный параметр **AuditData** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.27.5.8 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное встроенной функцией.

9.27.5.9 Входящие значения остальных входных параметров определяют в соответствии с 9.27.4.

9.27.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.28 Функция BioAPI_Identify

9.28.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_Identify (
  BioAPI_HANDLE BSPHandle,
  BioAPI_FMR MaxFMRRequested,
  BioAPI_BIR_SUBTYPE Subtype,
  const BioAPI_IDENTIFY_POPULATION *Population,
  uint32_t TotalNumberOfTemplates,
  BioAPI_BOOL Binning,
  uint32_t MaxNumberOfResults,
  uint32_t *NumberOfResults,
  BioAPI_CANDIDATE **Candidates,
  int32_t Timeout,
  BioAPI_BIR_HANDLE *AuditData);
```

и параметры, указанные в таблице 35.

Таблица 35

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
MaxFMRRequested	Входной параметр, исходящее значение	Входной параметр, входящее значение
Subtype	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы «Определение заполнения» (см. 9.2.16)	Входной параметр, исходящее значение	Входной параметр, входящее значение
TotalNumberOfTemplates	Входной параметр, исходящее значение	Входной параметр, входящее значение
Binning	Входной параметр, исходящее значение	Входной параметр, входящее значение
MaxNumberOfResults	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_NumberOfResults	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Candidates	Входной параметр, исходящее значение	Входной параметр, входящее значение
Timeout	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_AuditData	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumberOfResults	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Кандидат» (см. 9.2.17) с именами, начинающимися с **"Candidate_1_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Кандидат» с именами, начинающимися с **"Candidate_2_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Кандидат» с именами, начинающимися с **"Candidate_3_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Кандидат» с именами, начинающимися с **"Candidate_4_"**	Выходной параметр, входящее значение	Входной параметр, входящее значение

Окончание таблицы 35

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
AuditData	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.28.2 Ограничения на параметры

9.28.2.1 Исходящими значениями **BSPHandle**, **TotalNumberOfTemplates** и **MaxNumberOfResults** должны быть корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале.

9.28.2.2 Исходящим значением **MaxFMRRequested** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.28.2.3 Исходящим значением **Subtype** должно быть корректно представленное целое число в интервале от 0 до 255. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.28.2.4 Исходящим значением группы параметров «Определение заполнения» должно быть корректно представленное значение собственного типа **BioAPI_IDENTIFY_POPULATION**. Входящим значением должно быть стандартно представленное значение этого типа.

9.28.2.5 Исходящим значением **Binning** должно быть корректно представленное логическое значение. Входящим значением должно быть стандартно представленное логическое значение.

9.28.2.6 Исходящими значениями **no_NumberOfResults**, **no_Candidates** и **no_AuditData** должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.28.2.7 Исходящим значением **Timeout** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.28.2.8 Исходящим значением **NumberOfResults** должно быть или корректно представленное целое число в интервале от 0 до 4294967295, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.28.2.9 Исходящим значением **AuditData** должно быть или корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.28.2.10 Исходящими значениями каждой группы параметров «Кандидат» должны быть корректно представленные значения собственного типа **BioAPI_CANDIDATE**. Входящими значениями должно быть или стандартно представленные значения этого типа, или наборы пустых строк.

9.28.2.11 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.28.3 Входные параметры вызова функции

9.28.3.1 Целые числа, представленные исходящими значениями **BSPHandle**, **MaxFMRRequested, Subtype**, **TotalNumberOfTemplates**, **MaxNumberOfResults** и **Timeout**, должны быть присвоены собственным параметрам с теми же именами.

9.28.3.2 Значение типа **BioAPI_IDENTIFY_POPULATION**, представленное (см. 9.2.16.4) исходящим значением группы параметров «Определение заполнения», должно быть записано в переменную типа **BioAPI_IDENTIFY_POPULATION**, адрес которой должен быть присвоен собственному параметру **Population**.

9.28.3.3 Логическое значение, представленное исходящим значением **Binning**, должно быть присвоено собственному параметру с тем же именем.

9.28.3.4 Если исходящим значением **no_NumberOfResults** является **"true"**, то собственному параметру **NumberOfResults** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **uint32_t**.

9.28.3.5 Если исходящим значением **no_Candidates** является **"true"**, то собственному параметру **Candidates** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_Candidate*** (указатель).

9.28.3.6 Если исходящим значением **no_AuditData** является **"true"**, то собственному параметру **AuditData** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HANDLE**.

9.28.4 Выходные параметры вызова функции

9.28.4.1 Входящее значение **NumberOfResults** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **uint32_t**, на которую указывает собственный параметр **NumberOfResults**.

9.28.4.2 Входящие значения группы параметров «Кандидат» с префиксом **"Candidate_X_"** (где **X** =1, 2, 3 или 4) определяют следующим образом. Если собственный параметр **Candidates** имеет NULL-значение или переменная, на которую указывает собственный параметр, имеет NULL-значение, или собственный параметр **NumberOfResults** имеет NULL-значение, или значение целочисленной переменной, на которую указывает этот собственный параметр, меньше **X**, то входящим значением должен быть набор пустых строк. В противном случае входящим значением должно быть стандартно представленное (см. 9.2.14.5) значение типа **BioAPI_CANDIDATE** в позиции **X** массива, на который указывает переменная, на которую, в свою очередь, указывает собственный параметр **Candidates**.

9.28.4.3 Входящее значение **AuditData** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **AuditData**.

9.28.5 Входные параметры вызова связанного процесса

9.28.5.1 Входящими значениями **BSPHandle**, **MaxFMRRequested**, **Subtype**, **TotalNumberOfTemplates**, **MaxNumberOfResults** и **Timeout** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.28.5.2 Входящим значением группы параметров «Определение заполнения» должно быть стандартно представленное (см. 9.2.16.5) значение типа **BioAPI_IDENTIFY_POPULATION** в переменной, на которую указывает собственный параметр **Population**.

9.28.5.3 Входящим значением **Binning** должно быть стандартно представленное логическое значение в собственном параметре с тем же именем.

9.28.5.4 Входящим значением **no_NumberOfResults** должно быть **"true"**, если собственный параметр **NumberOfResults** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.28.5.5 Входящим значением **no_Candidates** должно быть **"true"**, если собственный параметр **Candidates** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.28.5.6 Входящим значением **no_AuditData** должно быть **"true"**, если собственный параметр **AuditData** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.28.5.7 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.28.5.8 Входящие значения остальных входных параметров определяют в соответствии с 9.28.4.

9.28.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.29 Функция BioAPI_Import

9.29.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_Import (
  BioAPI_HANDLE BSPHandle,
  const BioAPI_DATA *InputData,
  const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *InputFormat,
  const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat,
  BioAPI_BIR_PURPOSE Purpose,
  BioAPI_BIR_HANDLE *ConstructedBIR);
```

и параметры, указанные в таблице 36.

Таблица 36

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
InputData	Входной параметр, исходящее значение	Входной параметр, входящее значение
InputFormatOwner	Входной параметр, исходящее значение	Входной параметр, входящее значение
InputFormatType	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatOwner	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatType	Входной параметр, исходящее значение	Входной параметр, входящее значение
Purpose	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_ConstructedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
ConstructedBIR	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.29.2 Ограничения на параметры

9.29.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.29.2.2 Исходящим значением **InputData** должен быть корректно представленный набор байтов (см. 7.7). Входящим значением должен быть стандартно представленный набор байтов (см. 7.7.2).

9.29.2.3 Исходящими значениями **InputFormatOwner**, **InputFormatType**, **OutputFormatOwner** и **OutputFormatType** должны быть корректно представленные целые числа в интервале от 0 до 65535. Входящими значениями должны быть стандартно представленные целые числа в том же интервале.

9.29.2.4 Исходящим значением **Purpose** должно быть корректно представленное целое число в интервале от 0 до 255. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.29.2.5 Исходящим значением **no_ConstructedBIR** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.29.2.6 Исходящим значением **ConstructedBIR** должно быть или корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.29.2.7 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.29.3 Входные параметры вызова функции

9.29.3.1 Целые числа, представленные исходящими значениями **BSPHandle** и **Purpose**, должны быть присвоены собственным параметрам с теми же именами.

9.29.3.2 Если исходящее значение **InputData** является пустой строкой, то собственному параметру с тем же именем должно быть присвоено NULL-значение. В противном случае набор байтов, представленный исходящим значением, должен быть записан в блок памяти достаточного объема. Адрес и длина блока памяти должны быть записаны в поля **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, адрес которой должен быть присвоен собственному параметру **InputData**.

9.29.3.3 Целые числа, представленные исходящими значениями **InputFormatOwner** и **InputFormatType**, должны быть записаны в поля **FormatOwner** и **FormatType** (соответственно) переменной типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, адрес которой должен быть присвоен собственному параметру **InputFormat**.

9.29.3.4 Целые числа, представленные исходящими значениями **OutputFormatOwner** и **OutputFormatType**, должны быть записаны в поля **FormatOwner** и **FormatType** (соответственно) переменной типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, адрес которой должен быть присвоен собственному параметру **OutputFormat**.

9.29.3.5 Если исходящим значением **no_ConstructedBIR** является **"true"**, то собственному параметру **ConstructedBIR** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HANDLE**.

9.29.4 Выходные параметры вызова функции

Входящее значение **ConstructedBIR** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в интервале от минус 2147483648 до плюс 2147483647 (см. 7.4.3). Целое число должно быть считано из переменной типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **ConstructedBIR**.

9.29.5 Входные параметры вызова связанного процесса

9.29.5.1 Входящими значениями **BSPHandle** и **Purpose** должны быть стандартно представленные целые числа в собственных параметрах с тем же именем.

9.29.5.2 Входящее значение **InputData** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должен быть стандартно представленный (см 7.7.2) набор байтов в блоке памяти, адрес и длина которого находятся в полях **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, на которую указывает собственный параметр **InputData**.

9.29.5.3 Входящими значениями **InputFormatOwner** и **InputFormatType** должны быть стандартно представленные целые числа в полях **FormatOwner** и **FormatType** (соответственно) переменной типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, на которую указывает собственный параметр **InputFormat**.

9.29.5.4 Входящими значениями **OutputFormatOwner** и **OutputFormatType** должны быть стандартно представленные целые числа в полях **FormatOwner** и **FormatType** (соответственно) переменной типа **BioAPI_BIR_BIOMETRIC_DATA_FORMAT**, на которую указывает собственный параметр **OutputFormat**.

9.29.5.5 Входящим значением **no_ConstructedBIR** должно быть **"true"**, если собственный параметр **ConstructedBIR** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.29.5.6 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.29.5.7 Входящие значения остальных входных параметров определяют в соответствии с 9.29.4.

9.29.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.30 Функция BioAPI_PresetIdentifyPopulation

9.30.1 Схема вызова функции

Эта функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_PresetIdentifyPopulation (
  BioAPI_HANDLE BSPHandle,
  const BioAPI_IDENTIFY_POPULATION *Population);
```

и параметры, указанные в таблице 37.

Таблица 37

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Определение заполнения» (см. 9.2.16)	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.30.2 Ограничения на параметры

9.30.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.30.2.2 Исходящим значением «Определение заполнения» должно быть корректно представленное значение собственного типа **BioAPI_IDENTIFY_POPULATION**. Входящим значением должно быть стандартно представленное значение этого типа.

9.30.2.3 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.30.3 Входные параметры вызова функции

9.30.3.1 Целое число, представленное исходящим значением **BSPHandle**, должно быть присвоено собственному параметру с тем же именем.

9.30.3.2 Значение типа **BioAPI_IDENTIFY_POPULATION**, представленное (см. 9.2.16.4) исходящим значением группы параметров «Определение заполнения», должно быть записано в переменную типа **BioAPI_IDENTIFY_POPULATION**, адрес которой должен быть присвоен собственному параметру **Population**.

9.30.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.30.5 Входные параметры вызова связанного процесса

9.30.5.1 Входящим значением **BSPHandle** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.30.5.2 Входящим значением группы параметров «Определение заполнения» должно быть стандартно представленное (см. 9.2.16.5) значение типа **BioAPI_IDENTIFY_POPULATION** в переменной, на которую указывает собственный параметр **Population**.

9.30.5.3 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное встроенной функцией.

9.30.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.31 Функция BioAPI_DbOpen

9.31.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_DbOpen (
  BioAPI_HANDLE BSPHandle,
  const BioAPI_UUID *DbUuid,
  BioAPI_DB_ACCESS_TYPE AccessRequest,
  BioAPI_DB_HANDLE *DbHandle,
  BioAPI_DB_MARKER_HANDLE *MarkerHandle);
```

и параметры, указанные в таблице 38.

Таблица 38

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Тип доступа» (см. 9.2.19)	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_DbHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_MarkerHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbHandle	Выходной параметр, входящее значение	Входной параметр, входящее значение
MarkerHandle	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.31.2 Ограничения на параметры

9.31.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.31.2.2 Исходящим значением **DbUuid** должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.31.2.3 Исходящим значением группы параметров «Тип доступа» должно быть корректно представленное значение собственного типа **BioAPI_DB_ACCESS_TYPE**. Входящим значением должно быть стандартно представленное значение этого типа.

9.31.2.4 Исходящими значениями **no_DbHandle** и **no_MarkerHandle** должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.31.2.5 Исходящим значением **DbHandle** должно быть или корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.31.2.6 Исходящим значением **MarkerHandle** должно быть или корректно представленное целое число в интервале от 0 до 4294967295, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.31.2.7 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.31.3 Входные параметры вызова функции

9.31.3.1 Целое число, представленное исходящим значением **BSPHandle**, должно быть присвоено собственному параметру с тем же именем.

9.31.3.2 УУИД, представленный исходящим значением **DbUuid**, должен быть записан в переменную типа **BioAPI_UUID**, адрес которой должен быть присвоен собственному параметру **DbUuid**.

9.31.3.3 Значение типа **BioAPI_Db_ACCESS_TYPE**, представленное (см. 9.2.19.4) исходящим параметром группы параметров «Тип доступа», должно быть присвоено собственному параметру **AccessRequest**.

9.31.3.4 Если исходящим значением **no_DbHandle** является **"true"**, то собственному параметру **DbHandle** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_DB_HANDLE**.

9.31.3.5 Если исходящим значением **no_MarkerHandle** является **"true"**, то собственному параметру **MarkerHandle** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_DB_MARKER_HANDLE**.

9.31.4 Выходные параметры вызова функции

9.31.4.1 Входящее значение **DbHandle** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значение должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_DB_HANDLE**, на которую указывает собственный параметр **DbHandle**.

9.31.4.2 Входящее значение **MarkerHandle** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_DB_MARKER_HANDLE**, на которую указывает собственный параметр **MarkerHandle**.

9.31.5 Входные параметры вызова связанного процесса

9.31.5.1 Входящим значением **BSPHandle** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.31.5.2 Входящим значением **DbUuid** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **DbUuid**.

9.31.5.3 Входящим значением группы параметров «Тип доступа» должно быть стандартно представленное (см. 9.2.19.5) значение типа **BioAPI_DB_ACCESS_TYPE** в собственном параметре **AccessRequest.**

9.31.5.4 Входящим значением **no_DbHandle** должно быть **"true"**, если собственный параметр **DbHandle** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.31.5.5 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.31.5.6 Входящие значения остальных входных параметров определяют в соответствии с 9.31.4.

9.31.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.32 Функция BioAPI_DbClose

9.32.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_DbClose (
   BioAPI_HANDLE BSPHandle,
   BioAPI_DB_HANDLE DbHandle);
```

и параметры, указанные в таблице 39.

Таблица 39

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.32.2 Ограничения на параметры

9.32.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.32.2.2 Исходящим значением **DbHandle** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.32.2.3 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.32.3 Входные параметры вызова функции

Целое число, представленное исходящими значениями **BSPHandle** и **DbHandle**, должно быть присвоено собственному параметру с тем же именем.

9.32.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.32.5 Входные параметры вызова связанного процесса

9.32.5.1 Входящим значением **BSPHandle** и **DbHandle** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.32.5.2 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное встроенной функцией.

9.32.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.33 Функция BioAPI_DbCreate

9.33.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_DbCreate (
   BioAPI_HANDLE BSPHandle,
   const BioAPI_UUID *DbUuid,
   uint32_t NumberOfRecords,
   BioAPI_DB_ACCESS_TYPE AccessRequest,
   BioAPI_DB_HANDLE *DbHandle);
```

и параметры, указанные в таблице 40.

Таблица 40

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumberOfRecords	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Тип доступа» (см. 9.2.19)	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_DbHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbHandle	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.33.2 Ограничения на параметры

9.33.2.1 Исходящими значениями **BSPHandle** и **NumberOfRecords** должны быть корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.33.2.2 Исходящим значением **DbUuid** должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.33.2.3 Исходящим значением группы параметров «Тип доступа» должно быть корректно представленное значение собственного типа **BioAPI_DB_ACCESS_TYPE**. Входящим значением должно быть стандартно представленное значение этого типа.

9.33.2.4 Исходящим значением **no_DbHandle** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значения (см. 7.5.2).

9.33.2.5 Исходящим значением **DbHandle** должно быть или корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.33.2.6 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.33.3 Входные параметры вызова функции

9.33.3.1 Целые числа, представленные исходящими значениями **BSPHandle** и **NumberOfRecords,** должны быть присвоены собственным параметрам с теми же именами.

9.33.3.2 УУИД, представленный исходящим значением **DbUuid**, должен быть записан в переменную типа **BioAPI_UUID**, адрес которой должен быть присвоен собственному параметру **DbUuid**.

9.33.3.3 Значение типа **BioAPI_Db_ACCESS_TYPE**, представленное (см. 9.2.19.4) исходящим параметром группы параметров «Тип доступа», должно быть присвоено собственному параметру **AccessRequest**.

9.33.3.4 Если исходящим значением **no_DbHandle** является **"true"**, то собственному параметру **DbHandle** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_DB_HANDLE**.

9.33.4 Выходные параметры вызова функции

Входящее значение **DbHandle** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_DB_HANDLE**, на которую указывает собственный параметр **DbHandle**.

9.33.5 Входные параметры вызова связанного процесса

9.33.5.1 Входящими значениями **BSPHandle** и **NumberOfRecords** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.33.5.2 Входящим значением **DbUuid** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **DbUuid**.

9.33.5.3 Входящим значением группы параметров «Тип доступа» должно быть стандартно представленное (см. 9.2.19.5) значение типа **BioAPI_DB_ACCESS_TYPE** в собственном параметре **AccessRequest**.

9.33.5.4 Входящим значением **no_DbHandle** должно быть **"true"**, если собственный параметр **DbHandle** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.33.5.5 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное встроенной функцией.

9.33.5.6 Входящие значения остальных входных параметров определяют в соответствии с 9.33.4.

9.33.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.34 Функция BioAPI_DbDelete

9.34.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_DbDelete (
   BioAPI_HANDLE BSPHandle,
   const BioAPI_UUID *DbUuid);
```

и параметры, указанные в таблице 41.

Таблица 41

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.34.2 Ограничения на параметры

9.34.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.34.2.2 Исходящим значением **DbUuid** должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.34.2.3 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.34.3 Входные параметры вызова функции

9.34.3.1 Целые числа, представленные исходящими значениями **BSPHandle** и **DbHandle**, должны быть присвоены собственным параметрам с теми же именами.

9.34.3.2 УУИД, представленный исходящим значением **DbUuid**, должен быть записан в переменную типа **BioAPI_UUID**, адрес которой должен быть присвоен собственному параметру **DbUuid**.

9.34.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.34.5 Входные параметры вызова связанного процесса

9.34.5.1 Входящим значением **BSPHandle** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.34.5.2 Входящим значением **DbUuid** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **DbUuid**.

9.34.5.3 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.34.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.35 Функция BioAPI_DbSetMarker

9.35.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_DbSetMarker (
   BioAPI_HANDLE BSPHandle,
   BioAPI_DB_HANDLE DbHandle,
   const BioAPI_UUID *KeyValue,
   BioAPI_DB_MARKER_HANDLE MarkerHandle);
```

и параметры, указанные в таблице 42.

Таблица 42

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
KeyValue	Входной параметр, исходящее значение	Входной параметр, входящее значение
MarkerHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.35.2 Ограничения на параметры

9.35.2.1 Исходящими значениями **BSPHandle** и **MarkerHandle** должны быть корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.35.2.2 Исходящим значением **DbHandle** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.35.2.3 Исходящим значением **KeyValue** должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.35.2.4 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.35.3 Входные параметры вызова функции

9.35.3.1 Целые числа, представленные исходящими значениями **BSPHandle**, **DbHandle** и **MarkerHandle**, должны быть присвоены собственным параметрам с теми же именами.

9.35.3.2 УУИД, представленный исходящим значением **KeyValue**, должен быть записан в переменную типа **BioAPI_UUID**, адрес которой должен быть присвоен собственному параметру **KeyValue**.

9.35.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.35.5 Входные параметры вызова связанного процесса

9.35.5.1 Входящими значениями **BSPHandle**, **DbHandle** и **MarkerHandle** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.35.5.2 Входящим значением **KeyValue** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **KeyValue**.

9.35.5.3 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.35.5.4 Входящие значения остальных входных параметров определяют в соответствии с 9.35.4.

9.35.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.36 Функция BioAPI_DbFreeMarker

9.36.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_DbFreeMarker (
  BioAPI_HANDLE BSPHandle,
  BioAPI_DB_MARKER_HANDLE MarkerHandle);
```

и параметры, указанные в таблице 43.

Таблица 43

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
MarkerHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.36.2 Ограничения на параметры

9.36.2.1 Исходящими значениями **BSPHandle** и **MarkerHandle** должны быть корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале.

9.36.2.2 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.36.3 Входные параметры вызова функции

Целые числа, представленные исходящими значениями **BSPHandle** и **MarkerHandle**, должны быть присвоены собственным параметрам с теми же именами.

9.36.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.36.5 Входные параметры вызова связанного процесса

9.36.5.1 Входящими значениями **BSPHandle** и **MarkerHandle** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.36.5.2 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.36.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.37 Функция BioAPI_DbStoreBIR

9.37.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_DbStoreBIR (
   BioAPI_HANDLE BSPHandle,
   const BioAPI_INPUT_BIR *BIRToStore,
   BioAPI_DB_HANDLE DbHandle,
   BioAPI_UUID *BirUuid);
```

и параметры, указанные в таблице 44.

Таблица 44

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"BIRToStore_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
DBHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_BirUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
BirUuid	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.37.2 Ограничения на параметры

9.37.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.37.2.2 Исходящим значением группы параметров «Ввод ЗБИ» должно быть корректно представленное значение собственного типа **BioAPI_INPUT_BIR**. Входящим значением должно быть стандартно представленное значение этого типа.

9.37.2.3 Исходящим значением **DbHandle** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.37.2.4 Исходящим значением **no_BirUuid** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.37.2.5 Исходящим значением **BirUuid** должен быть или корректно представленный УУИД (см. 7.6), или пустая строка. Входящим значением должен быть или стандартно представленный УУИД (см. 7.6.3), или пустая строка.

9.37.2.6 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.37.3 Входные параметры вызова функции

9.37.3.1 Целые числа, представленные исходящими значениями **BSPHandle** и **DbHandle**, должны быть присвоены собственным параметрам с теми же именами.

9.37.3.2 Значение типа **BioAPI_INPUT_BIR**, представленное (см. 9.2.15.4) исходящим значением группы параметров «Ввод ЗБИ», должно быть записано в переменную типа **BioAPI_INPUT_BIR**, адрес которой должен быть присвоен собственному параметру **BirToStore**.

9.37.3.3 Если исходящим значением **no_BirUuid** является **"true"**, то собственному параметру **BirUuid** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_UUID**.

9.37.4 Выходные параметры вызова функции

Входящее значение **BirUuid** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **BirUuid**.

9.37.5 Входные параметры вызова связанного процесса

9.37.5.1 Входящими значениями **BSPHandle** и **DbHandle** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.37.5.2 Входящими значениями группы параметров «Ввод ЗБИ» должно быть стандартно представленное (см. 9.2.15.5) значение типа **BioAPI_INPUT_BIR** в собственном параметре **BirToStore**.

9.37.5.3 Входящим значением **no_BirUuid** должно быть **"true"**, если собственный параметр **BirUuid** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.37.5.4 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.37.5.5 Входящие значения других входных параметров определяют в соответствии с 9.37.4.

9.37.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.38 Функция BioAPI_DbGetBIR

9.38.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_DbGetBIR (
   BioAPI_HANDLE BSPHandle,
   BioAPI_DB_HANDLE DbHandle,
   const BioAPI_UUID *KeyValue,
   BioAPI_BIR_HANDLE *RetrievedBIR,
   BioAPI_DB_MARKER_HANDLE *MarkerHandle);
```

и параметры, указанные в таблице 45.

Таблица 45

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DBHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
KeyValue	Входной параметр, исходящее значение	Входной параметр, входящее значение

Окончание таблицы 45

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
no_RetrievedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_MarkerHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
RetrievedBIR	Выходной параметр, входящее значение	Входной параметр, входящее значение
MarkerHandle	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.38.2 Ограничения на параметры

9.38.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.38.2.2 Исходящим значением **DbHandle** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.38.2.3 Исходящим значением **KeyValue** должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.38.2.4 Исходящими значениями **no_RetrievedBIR** и **no_MarkerHandle** должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.38.2.5 Исходящим значением **RetrievedBIR** должно быть или корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.38.2.6 Исходящим значением **MarkerHandle** должно быть или корректно представленное целое число в интервале от 0 до 4294967295, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.38.2.7 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.38.3 Входные параметры вызова функции

9.38.3.1 Целые числа, представленные исходящими значениями **BSPHandle** и **DbHandle**, должны быть присвоены собственным параметрам с теми же именами.

9.38.3.2 УУИД, представленный исходящим значением **KeyValue**, должен быть записан в переменную типа **BioAPI_UUID**, адрес которой должен быть присвоен собственному параметру **KeyValue**.

9.38.3.3 Если исходящим значением **no_RetrievedBIR** является **"true"**, то собственному параметру **RetrievedBIR** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HANDLE**.

9.38.3.4 Если исходящим значением **no_MarkerHandle** является **"true"**, то собственному параметру **MarkerHandle** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_DB_MARKER_HANDLE**.

9.38.4 Выходные параметры вызова функции

9.38.4.1 Входящее значение **RetrievedBIR** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **RetrievedBIR**.

9.38.4.2 Входящее значение **MarkerHandle** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_DB_MARKER_HANDLE**, на которую указывает собственный параметр **MarkerHandle**.

9.38.5 Входные параметры вызова связанного процесса

9.38.5.1 Входящими значениями **BSPHandle** и **DbHandle** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.38.5.2 Входящим значением **KeyValue** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **KeyValue**.

9.38.5.3 Входящим значением **no_RetrievedBIR** должно быть **"true"**, если собственный параметр **RetrievedBIR** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.38.5.4 Входящим значением **no_MarkerHandle** должно быть **"true"**, если собственный параметр **MarkerHandle** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.38.5.5 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.38.5.6 Входящие значения остальных входных параметров определяют соответствии с 9.38.4.

9.38.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.39 Функция BioAPI_DbGetNextBIR

9.39.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_DbGetNextBIR (
   BioAPI_HANDLE BSPHandle,
   BioAPI_DB_HANDLE DbHandle,
   BioAPI_DB_MARKER_HANDLE MarkerHandle,
   BioAPI_BIR_HANDLE *RetrievedBIR,
   BioAPI_UUID *BirUuid);
```

и параметры, указанные в таблице 46.

Таблица 46

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
MarkerHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_RetrievedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_BirUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
RetrievedBIR	Выходной параметр, входящее значение	Входной параметр, входящее значение
BirUuid	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.39.2 Ограничения на параметры

9.39.2.1 Исходящими значениями **BSPHandle** и **MarkerHandle** должны быть корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале.

9.39.2.2 Исходящим значением **DbHandle** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.39.2.3 Исходящими значениями **no_RetrievedBIR** и **no_BirUuid** должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.39.2.4 Исходящим значением **RetrievedBIR** должно быть или корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.39.2.5 Исходящим значением **BirUuid** должен быть или корректно представленный УУИД (см. 7.6), или пустая строка. Входящим значением должен быть или стандартно представленный УУИД (см. 7.6.3), или пустая строка.

9.39.2.6 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.39.3 Входные параметры вызова функции

9.39.3.1 Целые числа, представленные исходящими значениями **BSPHandle**, **DbHandle** и **MarkerHandle**, должны быть присвоены собственным параметрам с тем же именем.

9.39.3.2 Если исходящим значением **no_RetrievedBIR** является **"true"**, то собственному параметру **RetrievedBIR** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BIR_HANDLE**.

9.39.3.3 Если исходящим значением **no_BirUuid** является **"true"**, то собственному параметру **BirUuid** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_UUID**.

9.39.4 Выходные параметры вызова функции

9.39.4.1 Входящее значение **RetrievedBIR** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **RetrievedBIR**.

9.39.4.2 Входящее значение **BirUuid** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **BirUuid**.

9.39.5 Входные параметры вызова связанного процесса

9.39.5.1 Входящими значениями **BSPHandle**, **DbHandle** и **MarkerHandle** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.39.5.2 Входящим значением **no_RetrievedBIR** должно быть **"true"**, если собственный параметр **RetrievedBIR** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.39.5.3 Входящее значение **no_BirUuid** должно быть **"true"**, если собственный параметр **BirUuid** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.39.5.4 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.39.5.5 Входящие значения остальных входных параметров определяют в соответствии с 9.39.4.

9.39.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.40 Функция BioAPI_DbDeleteBIR

9.40.1 Схема вызова функции

Эта функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_DbDeleteBIR (
  BioAPI_HANDLE BSPHandle,
  BioAPI_DB_HANDLE DbHandle,
  const BioAPI_UUID *KeyValue);
```

и параметры, указанные в таблице 47.

Таблица 47

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
KeyValue	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.40.2 Ограничения на параметры

9.40.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.40.2.2 Исходящим значением **DbHandle** должно быть корректно представленное целое число в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.40.2.3 Исходящим значением **KeyValue** должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.40.2.4 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.40.3 Входные параметры вызова функции

9.40.3.1 Целые числа, представленные исходящими значениями **BSPHandle** и **DbHandle**, должны быть присвоены собственным параметрам с теми же именами.

9.40.3.2 УУИД, представленный исходящим значением **KeyValue**, должен быть записан в переменную типа **BioAPI_UUID**, адрес которой должен быть присвоен собственному параметру **KeyValue**.

9.40.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.40.5 Входные параметры вызова связанного процесса

9.40.5.1 Входящими значениями **BSPHandle** и **DbHandle** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.40.5.2 Входящим значением **KeyValue** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **KeyValue**.

9.40.5.3 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.40.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.41 Функция BioAPI_SetPowerMode

9.41.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_SetPowerMode (
  BioAPI_HANDLE BSPHandle,
  BioAPI_UNIT_ID UnitID,
  BioAPI_POWER_MODE PowerMode);
```

и параметры, указанные в таблице 48.

Таблица 48

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
PowerMode	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.41.2 Ограничения на параметры

9.41.2.1 Исходящими значениями **BSPHandle**, **UnitID** и **PowerMode** должны быть корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале.

9.41.2.2 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.41.3 Входные параметры вызова функции

Целые числа, представленные исходящими значениями **BSPHandle_UnitID** и **PowerMode**, должны быть присвоены собственным параметрам с теми же именами.

9.41.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.41.5 Входные параметры вызова связанного процесса

9.41.5.1 Входящими значениями **BSPHandle_UnitID** и **PowerMode** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.41.5.2 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.41.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.42 Функция BioAPI_SetIndicatorStatus

9.42.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_SetIndicatorStatus (
   BioAPI_HANDLE BSPHandle,
   BioAPI_UNIT_ID UnitID,
   BioAPI_INDICATOR_STATUS IndicatorStatus);
```

и параметры, указанные в таблице 49.

Таблица 49

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
IndicatorStatus	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.42.2 Ограничения на параметры

9.42.2.1 Исходящими значениями **BSPHandle** и **UnitID** должны быть корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295. Входящими значениями должны быть стандартно представленное целые числа (см. 7.4.3) в том же интервале.

9.42.2.2 Исходящим значением **IndicatorStatus** должно быть корректно представленное целое число в интервале от 0 до 255. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.42.2.3 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.42.3 Входные параметры вызова функции

Целые числа, представленные исходящими значениями **BSPHandle**, **UnitID** и **IndicatorStatus**, должны быть присвоены собственным параметрам с теми же именами.

9.42.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.42.5 Входные параметры вызова связанного процесса

9.42.5.1 Входящими значениями **BSPHandle**, **UnitID** и **IndicatorStatus** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.42.5.2 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.42.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.43 Функция BioAPI_GetIndicatorStatus

9.43.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_GetIndicatorStatus (
   BioAPI_HANDLE BSPHandle,
   BioAPI_UNIT_ID UnitID,
   BioAPI_INDICATOR_STATUS *IndicatorStatus);
```

и параметры, указанные в таблице 50.

Таблица 50

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_IndicatorStatus	Входной параметр, исходящее значение	Входной параметр, входящее значение
IndicatorStatus	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.43.2 Ограничения на параметры

9.43.2.1 Исходящими значениями **BSPHandle** и **UnitID** должны быть корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295. Входящими значениями должны быть стандартно представленное целые числа (см. 7.4.3) в том же интервале.

9.43.2.2 Исходящим значением **no_IndicatorStatus** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.43.2.3 Исходящим значением **IndicatorStatus** должно быть или корректно представленное целое число в интервале от 0 до 255, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.43.2.4 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.43.3 Входные параметры вызова функции

9.43.3.1 Целые числа, представленные исходящими значениями **BSPHandle** и **UnitID**, должны быть присвоены собственным параметрам с теми же именами.

9.43.3.2 Если исходящим значением **no_IndicatorStatus** является **"true"**, то собственному параметру **IndicatorStatus** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_INDICATOR_STATUS**.

9.43.4 Выходные параметры вызова функции

Входящее значение **IndicatorStatus** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_INDICATOR_STATUS**, на которую указывает собственный параметр **IndicatorStatus**.

9.43.5 Входные параметры вызова связанного процесса

9.43.5.1 Входящими значениями **BSPHandle** и **UnitID** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.43.5.2 Входящим значением **no_IndicatorStatus** должно быть **"true"**, если собственный параметр **IndicatorStatus** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.43.5.3 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.43.5.4 Входящие значения остальных входных параметров определяют в соответствии с 9.43.4.

9.43.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.44 Функция BioAPI_CalibrateSensor

9.44.1 Схема вызова функции

Эта функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_CalibrateSensor (
   BioAPI_HANDLE BSPHandle,
   int32_t Timeout);
```

и параметры, указанные в таблице 51.

Таблица 51

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Timeout	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.44.2 Ограничения на параметры

9.44.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.44.2.2 Исходящим значением **Timeout** должно быть корректно представленное целое число (см. 7.4) в интервале от минус 2147483648 до плюс 2147483647. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.44.2.3 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.44.3 Входные параметры вызова функции

Целые числа, представленные исходящими значениями **BSPHandle** и **Timeout**, должны быть присвоены собственным параметрам с теми же именами.

9.44.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.44.5 Входные параметры вызова связанного процесса

9.44.5.1 Входящими значениями **BSPHandle** и **Timeout** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.44.5.2 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.44.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.45 Функция BioAPI_Cancel

9.45.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_Cancel (
  BioAPI_HANDLE BSPHandle);
```

и параметры, указанные в таблице 52.

Таблица 52

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.45.2 Ограничения на параметры

9.45.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.45.2.2 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.45.3 Входные параметры вызова функции

Целое число, представленное исходящим значением **BSPHandle**, должно быть присвоено собственному параметру с тем же именем.

9.45.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.45.5 Входные параметры вызова связанного процесса

9.45.5.1 Входящим значением **BSPHandle** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.45.5.2 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.45.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.46 Функция BioAPI_Free

9.46.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
  BioAPI_Free (
  void* Ptr);
```

и параметры, указанные в таблице 53.

Таблица 53

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
Ptr	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.46.2 Ограничения на параметры

9.46.2.1 Исходящим значением **Ptr** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.46.2.2 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.46.3 Входные параметры вызова функции

Целое число, представленное исходящим значением **Ptr,** должно быть присвоено собственному параметру с тем же именем.

9.46.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.46.5 Входные параметры вызова связанного процесса

9.46.5.1 Входящим значением **Ptr** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.46.5.2 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.46.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.47 Функция BioAPI_Util_InstallBSP

9.47.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_Util_InstallBSP (
  BioAPI_INSTALL_ACTION Action,
  BioAPI_INSTALL_ERROR *Error,
  const BioAPI_BSP_SCHEMA *BSPSchema);
```

и параметры, указанные в таблице 54.

Таблица 54

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
Action	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Error	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема ПБУ» (см. 9.2.10)	Входной параметр, исходящее значение	Входной параметр, входящее значение
ErrorCode	Выходной параметр, входящее значение	Входной параметр, входящее значение
ErrorString	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.47.2 Ограничения на параметры

9.47.2.1 Исходящим значением **Action** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.47.2.2 Исходящим значением **no_Error** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.47.2.3 Исходящим значением группы параметров «Схема ПБУ» должно быть корректно представленное значение собственного типа **BioAPI_BSP_SCHEMA**. Входящим значением должно быть стандартно представленное значение этого типа или набор пустых строк.

9.47.2.4 Исходящим значением **ErrorCode** должно быть или корректно представленное целое число в интервале от 0 до 4294967295, или пустая строка. Входящим значением должно быть или корректно представленное целое число в том же интервале, или пустая строка.

9.47.2.5 Исходящим или входящим значением **ErrorString** должна быть строка символов, кодировка UTF-8 которой не должна быть более 268 байтов, не содержащая символов NUL (0).

9.47.2.6 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.47.3 Входные параметры вызова функции

9.47.3.1 Целое число, представленное исходящим значением **Action**, должно быть присвоено собственному параметру с тем же именем.

9.47.3.2 Если исходящим значением **no_Error** является **"true"**, то собственному параметру **Error** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_INSTALL_ERROR**.

9.47.3.3 Значение типа **BioAPI_BSP_SCHEMA**, представленное (см. 9.2.10.4) исходящим значением группы параметров «Схема ПБУ», должно быть записано в переменную типа **BioAPI_BSP_SCHEMA**, адрес которой должен быть присвоен собственному параметру **BSPSchema**.

9.47.4 Выходные параметры вызова функции

Входящие значения **ErrorCode** и **ErrorString** определяют следующим образом. Если собственный параметр **Error** имеет NULL-значение, то входящими значениями должны быть пустые строки. В противном случае входящим значением **ErrorCode** должно быть стандартно представленное целое число в поле **ErrorCode** переменной типа **BioAPI_INSTALL_ERROR**, на которую указывает собственный параметр **Error**, а входящим значением **ErrorString** должна быть строка символов в поле **ErrorString** этой переменной.

9.47.5 Входные параметры вызова связанного процесса

9.47.5.1 Входящим значением **Action** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.47.5.2 Входящим значением **no_Error** должно быть **"true"**, если собственный параметр **Error** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.47.5.3 Входящим значением группы параметров «Схема ПБУ» должно быть стандартно представленное (см. 9.2.10.5) значение типа **BioAPI_BSP_SCHEMA** в переменной, на которую указывает собственный параметр **BSPSchema**.

9.47.5.4 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.47.5.5 Входящие значения остальных входных параметров определяют в соответствии с 9.47.4.

9.47.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.48 Функция BioAPI_Util_InstallBFP

9.48.1 Схема вызова функции

Данная функция принадлежит BioAPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioAPI_Util_InstallBFP (
  BioAPI_INSTALL_ACTION Action,
  BioAPI_INSTALL_ERROR *Error,
  const BioAPI_BFP_SCHEMA *BFPSchema);
```

и параметры, указанные в таблице 55.

Таблица 55

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие приложений)
Action	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Error	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема ПБФ» (см. 9.2.11)	Входной параметр, исходящее значение	Входной параметр, входящее значение
ErrorCode	Выходной параметр, входящее значение	Входной параметр, входящее значение
ErrorString	Выходной параметр, входящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, входящее значение

9.48.2 Ограничения на параметры

9.48.2.1 Исходящим значением **Action** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.48.2.2 Исходящим значением **no_Error** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.48.2.3 Исходящим значением группы параметров «Схема ПБФ» должно быть корректно представленное значение собственного типа **BioAPI_BFP_SCHEMA**. Входящим значением должно быть стандартно представленное значение этого типа или набор пустых строк.

9.48.2.4 Исходящим значением **ErrorCode** должно быть или корректно представленное целое число в интервале от 0 до 4294967295, или пустая строка. Входящим значением должно быть или корректно представленное целое число в том же интервале, или пустая строка.

9.48.2.5 Исходящим или входящим значением **ErrorString** должна быть строка символов, кодировка UTF-8 которой не должна превышать 268 байтов, не содержащая символов NUL (0).

9.48.2.6 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.48.3 Входные параметры вызова функции

9.48.3.1 Целое число, представленное исходящим значением **Action**, должно быть присвоено собственному параметру с тем же именем.

9.48.3.2 Если исходящим значением **no_Error** является **"true"**, то собственному параметру **Error** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_INSTALL_ERROR**.

9.48.3.3 Значение типа **BioAPI_BFP_SCHEMA**, представленное (см. 9.2.10.4) исходящим значением группы параметров «Схема ПБФ», должно быть записано в переменную типа **BioAPI_BFP_SCHEMA**, адрес которой должен быть присвоен собственному параметру **BFPSchema**.

9.48.4 Выходные параметры вызова функции

Входящие значения **ErrorCode** и **ErrorString** определяют следующим образом. Если собственный параметр **Error** имеет NULL-значение, то входящими значениями должны быть пустые строки. В противном случае входящим значением **ErrorCode** должно быть стандартно представленное целое число в поле **ErrorCode** переменной типа **BioAPI_INSTALL_ERROR**, на которую указывает собственный параметр **Error**, а входящим значением **ErrorString** должна быть строка символов в поле **ErrorString** этой переменной.

9.48.5 Входные параметры вызова связанного процесса

9.48.5.1 Входящим значением **Action** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.48.5.2 Входящим значением **no_Error** должно быть **"true"**, если собственный параметр **Error** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.48.5.3 Входящим значением группы параметров «Схема ПБФ» должно быть стандартно представленное (см. 9.2.11.5) значение типа **BioAPI_BFP_SCHEMA** в переменной, на которую указывает собственный параметр **BFPSchema**.

9.48.5.4 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.48.5.5 Входящие значения остальных входных параметров определяют в соответствии с 9.48.4.

9.48.6 Выходные параметры вызова связанного процесса

Выходные параметры отсутствуют.

9.49 Функция BioSPI_BSPLoad

9.49.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_BSPLoad (
   const BioAPI_UUID *BSPUuid,
   BioSPI_EventHandler BioAPINotifyCallback,
   BioSPI_BFP_ENUMERATION_HANDLER BFPEnumerationHandler,
   BioSPI_MEMORY_FREE_HANDLER MemoryFreeHandler);
```

и параметры, указанные в таблице 56.

Таблица 56

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
BioAPINotifyCallback	Входной параметр, исходящее значение	Входной параметр, входящее значение
BFPEnumerationHandler	Входной параметр, исходящее значение	Входной параметр, входящее значение
MemoryFreeHandler	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

9.49.2 Ограничения на параметры

9.49.2.1 Исходящим значением **BSPUuid** должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.49.2.2 Исходящими значениями **BioAPINotifyCallback**, **BFPEnumerationHandler** и **MemoryFreeHandler** должны быть или «0», или «*». Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в интервале от 0 до 4294967295.

9.49.2.3 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.49.3 Входные параметры вызова функции

9.49.3.1 УУИД, представленный исходящим значением **BSPUuid**, должен быть записан в переменную типа **BioAPI_UUID**, адрес которой должен быть присвоен собственному параметру **BSPUuid**.

9.49.3.2 Если исходящее значение **BioAPINotifyCallback** равно нулю, то NULL-значение указателя должно быть присвоено собственному параметру **BioAPINotifyCallback**. Если исходящее значение

BioAPINotifyCallback равно «*», то значение указателя (отличающееся от NULL-значения), которое является адресом собственной функции (в рамках компонента тестирования), реализующей функцию **BioSPI_EventHandler** стандартного интерфейса **BioAPI**, должно быть присвоено собственному параметру **BioAPINotifyCallback**.

Примечание — Если исходящее значение **BioAPINotifyCallback** не равно «0», то любой из следующих друг за другом входящих вызовов в адрес функции **BioSPI_EventHandler** стандартного интерфейса **BioAPI** вызывает связанный с этой функцией процесс, если такая связь существует.

9.49.3.3 Если исходящее значение **BFPEnumerationHandler** равно «0», то NULL-значение указателя должно быть присвоено собственному параметру **BFPEnumerationHandler**. Если исходящее значение **BFPEnumerationHandler** равно «*», то значение указателя (отличающееся от NULL-значения), которое является адресом собственной функции (в рамках компонента тестирования), реализующей функцию **BioSPI_BFP_ENUMERATION_HANDLER** стандартного интерфейса **BioAPI**, должно быть присвоено собственному параметру **BFPEnumerationHandler**.

Примечание — Если исходящее значение **BFPEnumerationHandler** не равно «0», то любой из следующих друг за другом входящих вызовов в адрес функции **BioSPI_BFP_ENUMERATION_HANDLER** стандартного интерфейса **BioAPI** вызывает связанный с этой функцией процесс, если такая связь существует.

9.49.3.4 Если исходящее значение **MemoryFreeHandler** равно «0», то NULL-значение указателя должно быть присвоено собственному параметру **MemoryFreeHandler**. Если исходящее значение **MemoryFreeHandler** равно «*», то значение указателя (отличающееся от NULL-значения), которое является адресом собственной функции (в рамках компонента тестирования), реализующей функцию **BioSPI_MEMORY_FREE_HANDLER** стандартного интерфейса **BioAPI**, должно быть присвоено собственному параметру **MemoryFreeHandler**.

Примечание — Если исходящее значение **MemoryFreeHandler** не равно «0», то любой из следующих друг за другом входящих вызовов в адрес функции **BioSPI_MEMORY_FREE_HANDLER** стандартного интерфейса BioAPI вызывает связанный с функцией процесс, если такая связь существует.

9.49.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.49.5 Входные параметры вызова связанного процесса

9.49.5.1 Входящим значением **BSPUuid** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **BSPUuid**.

9.49.5.2 Входящими значениями **BioAPINotifyCallback**, **BFPEnumerationHandler** и **MemoryFreeHandler** должны быть стандартно представленные целые числа в собственном параметре с теми же именами.

9.49.6 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.49.7 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то встроенная функция должна возвратить ноль.

9.50 Функция BioSPI_BSPUnload

9.50.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

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

и параметры, указанные в таблице 57.

Таблица 57

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

9.50.2 Ограничения на параметры

9.50.2.1 Исходящим значением **BSPUuid**, должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.50.2.2 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.50.3 Входные параметры вызова функции

УУИД, представленный исходящим значением **BSPUuid**, должен быть записан в переменную типа **BioAPI_UUID**, адрес которой должен быть присвоен собственному параметру **BSPUuid**.

9.50.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.50.5 Входные параметры вызова связанного процесса

Входящим значением **BSPUuid** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **BSPUuid**.

9.50.6 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.50.7 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то встроенная функция должна возвратить ноль.

9.51 Функция BioSPI_BSPAttach

9.51.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_BSPAttach (
   const BioAPI_UUID *BSPUuid,
   BioAPI_VERSION Version,
   const BioAPI_UNIT_LIST_ELEMENT *UnitList,
   uint32_t NumUnits,
   BioAPI_HANDLE BSPHandle);
```

и параметры, указанные в таблице 58.

Таблица 58

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
Version	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_1_UnitCategory	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_1_UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_2_UnitCategory	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_2_UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_3_UnitCategory	Входной параметр, исходящее значение	Входной параметр, входящее значение

Окончание таблицы 58

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
Unit_3_UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_4_UnitCategory	Входной параметр, исходящее значение	Входной параметр, входящее значение
Unit_4_UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumUnits	Входной параметр, исходящее значение	Входной параметр, входящее значение
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

9.51.2 Ограничения на параметры

9.51.2.1 Исходящим значением **BSPUuid** должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.51.2.2 Исходящим значением **Version** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 255. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.51.2.3 Исходящими значениями **Unit_X_UnitCategory** и **Unit_X_UnitID** (где **X** = 1, 2, 3 или 4) должны быть или корректно представлены целые числа в интервале от 0 до 4294967295, или пустые строки.

Входящими значениями должны быть или стандартно представленные целые числа (см. 7.4.3) в том же интервале, или пустые строки.

9.51.2.4 Исходящим значением **NumUnits** должно быть корректно представленное целое число в интервале от 0 до 4. Входящим значением должно быть стандартно представленное целое число в интервале от 0 до 4294967295.

9.51.2.5 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.51.2.6 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.51.3 Входные параметры вызова функции

9.51.3.1 УУИД, представленный исходящим значением **BSPUuid**, должен быть записан в переменную типа **BioAPI_UUID**, адрес которой должен быть присвоен собственному параметру **BSPUuid**.

9.51.3.2 Целые числа, представленные исходящими значениями **Version**, **NumUnits** и **BSPHandle**, должны быть присвоены собственным параметрам с теми же именами.

9.51.3.3 Целые числа, представленные исходящими значениями **Unit_X_UnitCategory** и **Unit_X_UnitID** (где **X** = 1, 2, 3 или 4), или 0, если исходящим значением является пустая строка, должны быть записаны в поля **UnitCategory** и **UnitID** (соответственно) элемента в позиции **X** массива четырех элементов типа **BioAPI_UNIT_LIST_ELEMENT**. Адрес этого массива должен быть присвоен собственному параметру **UnitList**.

9.51.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.51.5 Входные параметры вызова связанного процесса

9.51.5.1 Входящим значением **BSPUuid** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **BSPUuid**.

9.51.5.2 Входящими значениями **Version**, **NumUnits** и **BSPHandle** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.51.5.3 Входящие значения **Unit_X_UnitCategory** и **Unit_X_UnitID** (где **X** = 1, 2, 3 или 4) определяют следующим образом. Если собственный параметр **NumUnits** меньше **X**, то входящим значением должна быть пустая строка. В противном случае входящими значениями должны быть стандартно представленные целые числа в полях **UnitCategory** и **UnitID** (соответственно) элемента в позиции **X** массива элементов типа **BioAPI_UNIT_LIST_ELEMENT**, на который указывает собственный параметр **UnitList**.

9.51.6 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.51.7 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить ноль.

9.52 Функция BioSPI_BSPDetach

9.52.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_BSPDetach (
  BioAPI_HANDLE BSPHandle);
```

и параметры, указанные в таблице 59.

Таблица 59

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

9.52.2 Ограничения на параметры

9.52.2.1 Исходящим значением **BSPHandle** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.52.2.2 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.52.3 Входные параметры вызова функции

Целое число, представленное исходящим значением **BSPHandle**, должно быть присвоено собственному параметру с тем же именем.

9.52.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.52.5 Входные параметры вызова связанного процесса

Входящим значением **BSPHandle** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.52.6 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.52.7 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то встроенная функция должна возвратить ноль.

9.53 Функция BioSPI_QueryUnits

9.53.1 Схема вызова функции

Эта функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_QueryUnits (
  const BioAPI_UUID *BSPUuid,
  BioAPI_UNIT_SCHEMA **UnitSchemaArray,
  uint32_t *NumberOfElements);
```

и параметры, указанные в таблице 60.

Таблица 60

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_UnitSchemaArray	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема элементов» (см. 9.2.12) с именами, начинающимися с **"UnitSchema_1_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Элементы группы параметров «Схема элементов» с именами, начинающимися с **"UnitSchema_2_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Элементы группы параметров «Схема элементов» с именами, начинающимися с **"UnitSchema_3_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Элементы группы параметров «Схема элементов» с именами, начинающимися с **"UnitSchema_4_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
no_NumberOfElements	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumberOfElement	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_QueryUnits** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_QueryUnits**.

9.53.2 Выходные параметры вызова связанного процесса

9.53.2.1 Если собственный параметр **UnitSchemaArray** имеет значение, отличающееся от NULL, и целое число (например, **N**), представленное исходящим значением **NumberOfElements**, больше нуля, то для хранения массива **N** элементов типа **BioAPI_UNIT_SCHEMA** должен быть выделен блок памяти достаточного размера, а его адрес должен быть записан в переменную, на которую указывает собственный параметр **UnitSchemaArray**.

9.53.2.2 Исходящее значение группы параметров «Схема элементов» с префиксом **UnitSchema_X_** (где **X** = 1, 2, 3 или 4) должно быть обработано следующим образом. Если собственный параметр **UnitSchemaArray** имеет NULL-значение, или целое число, представленное исходящим значением **NumberOfElements**, меньше **X**, то исходящее значение группы параметров должно быть проигнорировано. В противном случае значение типа **BioAPI_UNIT_SCHEMA**, представленное исходящим значением группы параметров, должно быть записано в элемент позиции **X** массива элементов типа **BioAPI_UNIT_SCHEMA**, на который указывает переменная, на которую, в свою очередь, указывает собственный параметр **UnitSchemaArray**.

9.53.2.3 Исходящее значение **NumberOfElements** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **uint32_t**, на которую указывает собственный параметр **NumberOfElements**.

9.53.2.4 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.53.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **NumberOfElements**, должно быть присвоено значение, равное нулю;

b) переменной, на которую указывает собственный параметр **UnitSchemaArray**, должно быть присвоено NULL-значение;

c) собственная функция должна возвратить значение, равное нулю.

9.54 Функция BioSPI_QueryBFPs

9.54.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_QueryBFPs (
   const BioAPI_UUID *BSPUuid,
   BioAPI_BFP_LIST_ELEMENT **BFPList,
   uint32_t *NumberOfElements);
```

и параметры, указанные в таблице 61.

Таблица 61

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_BFPList	Входной параметр, исходящее значение	Входной параметр, входящее значение
BFP_1_BFPCategory	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
BFP_1_BFPUuid	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
BFP_2_BFPCategory	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
BFP_2_BFPUuid	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
BFP_3_BFPCategory	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
BFP_3_BFPUuid	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
BFP_4_BFPCategory	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
BFP_4_BFPUuid	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
no_NumberOfElements	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumberOfElements	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_QueryBFPs** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_QueryBFPs**.

9.54.2 Выходные параметры вызова связанного процесса

9.54.2.1 Если собственный параметр **BFPList** имеет значение, отличающееся от NULL-значения, и целое число (например, **N**), представленное исходящим значением **NumberOfElements**, больше нуля, то для хранения массива **N** элементов типа **BioAPI_BFP_LIST_ELEMENT** должен быть выделен блок памяти достаточного размера, а его адрес должен быть записан в переменную, на которую указывает собственное значение параметра **BFPList**.

9.54.2.2 Исходящие значения **BFP_X_BFPCategory** и **BFP_X_BFPUuid** (где **X** = 1, 2, 3 или 4) должны быть обработаны следующим образом. Если собственный параметр **BFPList** имеет NULL-значение или целое число, представленное исходящим значением **NumberOfElements**, меньше **X**, то исходящие значения **BFP_X_BFPCategory** и **BFP_X_BFPUuid** должны быть проигнорированы. В противном случае целое число, представленное исходящим значением **BFP_X_BFPCategory**, и УУИД, представленный исходящим значением **BFP_X_BFPUuid**, должны быть записаны в поля **BFPCategory** и **BFPUuid** (соответственно) элемента в позиции **X** массива элементов типа **BioAPI_BFP_LIST_ELEMENT**, на который указывает переменная, на которую, в свою очередь, указывает собственный параметр **BFPList**.

9.54.2.3 Исходящее значение **NumberOfElements** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **uint32_t**, на которую указывает собственный параметр **NumberOfElements**.

9.54.2.4 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.54.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **NumberOfElements**, должно быть присвоено значение, равное нулю;

b) переменной, на которую указывает собственный параметр **BFPList**, должно быть присвоено NULL-значение;

c) встроенная функция должна возвратить значение, равное нулю.

9.55 Функция BioSPI_ControlUnit

9.55.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_ControlUnit (
  BioAPI_HANDLE BSPHandle,
  BioAPI_UNIT_ID UnitID,
  uint32_t ControlCode,
  const BioAPI_DATA *InputData,
  BioAPI_DATA *OutputData);
```

и параметры, указанные в таблице 62.

Таблица 62

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
ControlCode	Входной параметр, исходящее значение	Входной параметр, входящее значение
InputData	Входной параметр, исходящее значение	Входной параметр, входящее значение

Окончание таблицы 62

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
no_OutputData	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputData	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_ControlUnit** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_ControlUnit**.

9.55.2 Выходные параметры вызова связанного процесса

9.55.2.1 Исходящее значение **OutputData** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае набор байтов, представленный исходящим значением, должен быть записан в блок памяти, адрес и размер которого должны быть записаны в поля **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, на которую указывает собственный параметр **OutputData**.

9.55.2.2 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.55.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) в полях **Data** и **Length** переменной, указанной собственным параметром **OutputData**, должно быть записано значение, равное нулю;

b) собственная функция должна возвратить значение, равное нулю.

9.56 Функция BioSPI_FreeBIRHandle

9.56.1 Схема вызова функции

Эта функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_FreeBIRHandle (
  BioAPI_HANDLE BSPHandle,
  BioAPI_BIR_HANDLE Handle);
```

и параметры, указанные в таблице 63.

Таблица 63

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Handle	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_FreeBIRHandle** аналогичны описаниям функции стандартного интерфейса **BioSPI_FreeBIRHandle**.

9.56.2 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением return, должно быть возвращено собственной функцией.

9.56.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то встроенная функция должна возвратить ноль.

9.57 Функция BioSPI_GetBIRFromHandle

9.57.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_GetBIRFromHandle (
  BioAPI_HANDLE BSPHandle,
  BioAPI_BIR_HANDLE Handle,
  BioAPI_BIR *BIR);
```

и параметры, указанные в таблице 64.

Таблица 64

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Handle	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_BIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «ЗБИ» (см. 9.2.14)	Выходной параметр, исходящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_GetBIRFromHandle** аналогичны описаниям функции стандартного интерфейса **BioSPI_GetBIRFromHandle**.

9.57.2 Выходные параметры вызова связанного процесса

9.57.2.1 Исходящее значение группы параметров «ЗБИ» должно быть обработано следующим образом. Если собственный параметр **BIR** имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае значение типа **BioAPI_BIR,** представленное исходящим значением, должно быть записано в переменную, на которую указывает собственный параметр **BIR**.

9.57.2.2 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.57.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) в переменную, на которую указывает собственный параметр **BIR,** во все поля должны быть записаны значения, равные нулю;

b) встроенная функция должна возвратить значение, равное нулю.

9.58 Функция BioSPI_GetHeaderFromHandle

9.58.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_GetHeaderFromHandle (
  BioAPI_HANDLE BSPHandle,
  BioAPI_BIR_HANDLE Handle,
  BioAPI_BIR_HEADER *Header);
```

и параметры, указанные в таблице 65.

Таблица 65

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Handle	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Header	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «ЗБИ» (см. 9.2.13)	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_GetHeaderFromHandle** аналогичны описаниям функции стандартного интерфейса **BioSPI_GetHeaderFromHandle**.

9.58.2 Выходные параметры вызова связанного процесса

9.58.2.1 Исходящее значение группы параметров «Заголовок ЗБИ» должно быть обработано следующим образом. Если собственный параметр **Header** имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае значение типа **BioAPI_BIR_HEADER**, представленное исходящим значением, должно быть записано в переменную, на которую указывает собственный параметр **Header**.

9.58.2.2 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.58.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) в переменную, на которую указывает собственный параметр **Header**, во все поля должны быть записаны значения, равные нулю;

b) встроенная функция должна возвратить значение, равное нулю.

9.59 Функция BioSPI_EnableEvents

9.59.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_EnableEvents (
  BioAPI_HANDLE BSPHandle,
  BioAPI_EVENT_MASK Events);
```

и параметры, указанные в таблице 66.

Таблица 66

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «События» (см. 9.2.4)	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_EnableEvents** аналогичны описаниям функции стандартного интерфейса **BioSPI_EnableEvents**.

9.59.2 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.59.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то встроенная функция должна возвратить значение, равное нулю.

9.60 Функция BioSPI_SetGUICallback

9.60.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_SetGUICallbacks (
  BioAPI_HANDLE BSPHandle,
  BioAPI_GUI_STREAMING_CALLBACK GuiStreamingCallback,
  void *GuiStreamingCallbackCtx,
  BioAPI_GUI_STATE_CALLBACK GuiStateCallback,
  void *GuiStateCallbackCtx);
```

и параметры, указанные в таблице 67.

Таблица 67

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
GuiStreamingCallback	Входной параметр, исходящее значение	Входной параметр, входящее значение
GuiStreaming CallbackCtx	Входной параметр, исходящее значение	Входной параметр, входящее значение
GuiStateCallback	Входной параметр, исходящее значение	Входной параметр, входящее значение
GuiStateCallbackCtx	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_SetGUICallbacks** аналогичны описаниям функции стандартного интерфейса **BioSPI_SetGUICallbacks**.

9.60.2 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.60.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.61 Функция BioSPI_Capture

9.61.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_Capture (
  BioAPI_HANDLE BSPHandle,
```

BioAPI_BIR_PURPOSE Purpose,
BioAPI_BIR_SUBTYPE Subtype,
const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat,
BioAPI_BIR_HANDLE *CapturedBIR,
int32_t Timeout,
BioAPI_BIR_HANDLE *AuditData);
и параметры, указанные в таблице 68.

Таблица 68

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Purpose	Входной параметр, исходящее значение	Входной параметр, входящее значение
Subtype	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatOwner	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatType	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_CapturedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
Timeout	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_AuditData	Входной параметр, исходящее значение	Входной параметр, входящее значение
CapturedBIR	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
AuditData	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_Capture** аналогичны описаниям функции стандартного интерфейса **BioSPI_Capture**.

9.61.2 Выходные параметры вызова связанного процесса

9.61.2.1 Исходящее значение **CapturedBIR** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **CapturedBIR**.

9.61.2.2 Исходящее значение **AuditData** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **AuditData**.

9.61.2.3 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.61.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **CapturedBIR**, должно быть присвоено значение, равное нулю;

b) переменной, на которую указывает собственный параметр **AuditData**, должно быть присвоено значение, равное нулю;

c) собственная функция должна возвратить значение, равное нулю.

9.62 Функция BioSPI_CreateTemplate

9.62.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_CreateTemplate (
   BioAPI_HANDLE BSPHandle,
   const BioAPI_INPUT_BIR *CapturedBIR,
   const BioAPI_INPUT_BIR *ReferenceTemplate,
   const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat,
   BioAPI_BIR_HANDLE *NewTemplate,
   const BioAPI_DATA *Payload,
   BioAPI_UUID *TemplateUUID);
```

и параметры, указанные в таблице 69.

Таблица 69

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"CapturedBIR_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» с именами, начинающимися с **"ReferenceTemplate_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatOwner	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatType	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_NewTemplate	Входной параметр, исходящее значение	Входной параметр, входящее значение
Payload	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_TemplateUUID	Входной параметр, исходящее значение	Входной параметр, входящее значение
NewTemplate	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
TemplateUUID	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_CreateTemplate** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_CreateTemplate.**

9.62.2 Выходные параметры вызова связанного процесса

9.62.2.1 Исходящее значение **NewTemplate** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **NewTemplate**.

9.62.2.2 Исходящее значение **TemplateUUID** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае УУИД, представленный исходящим значением, должен быть записан в переменную типа **BioAPI_UUID**, на которую указывает собственный параметр **TemplateUUID**.

9.62.2.3 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.62.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **NewTemplate**, должно быть присвоено значение, равное нулю;

b) переменной, на которую указывает собственный параметр **TemplateUUID**, должен быть присвоен УУИД «00000000-0000-0000-0000-000000000000»;

c) собственная функция должна возвратить значение, равное нулю.

9.63 Функция BioSPI_Process

9.63.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_Process (
  BioAPI_HANDLE BSPHandle,
  const BioAPI_INPUT_BIR *CapturedBIR,
  const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat,
  BioAPI_BIR_HANDLE *ProcessedBIR);
```

и параметры, указанные в таблице 70.

Таблица 70

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"CapturedBIR_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatOwner	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatType	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_ProcessedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
ProcessedBIR	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_Process** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_Process**.

9.63.2 Выходные параметры вызова связанного процесса

9.63.2.1 Исходящее значение **ProcessedBIR** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **ProcessedBIR**.

9.63.2.2 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.63.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **ProcessedBIR**, должно быть присвоено значение, равное нулю.

b) встроенная функция должна возвратить значение, равное нулю.

9.64 Функция BioSPI_ProcessWithAuxBIR

9.64.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_ProcessWithAuxBIR (
   BioAPI_HANDLE BSPHandle,
   const BioAPI_INPUT_BIR *CapturedBIR,
   const BioAPI_INPUT_BIR *AuxiliaryData,
   const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat,
   BioAPI_HANDLE *ProcessedBIR);
```

и параметры, указанные в таблице 71.

Таблица 71

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"CapturedBIR_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» с именами, начинающимися с **"AuxiliaryData_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatOwner	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatType	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_ProcessedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
ProcessedBIR	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_ProcessWithAuxBIR** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_ProcessWithAuxBIR**.

9.64.2 Выходные параметры вызова связанного процесса

9.64.2.1 Исходящее значение **ProcessedBIR** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **ProcessedBIR**.

9.64.2.2 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.64.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **ProcessedBIR**, должно быть присвоено значение, равное нулю;

b) встроенная функция должна возвратить значение, равное нулю.

9.65 Функция BioSPI_VerifyMatch

9.65.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_VerifyMatch (
  BioAPI_HANDLE BSPHandle,
  BioAPI_FMR MaxFMRRequested,
  const BioAPI_INPUT_BIR *ProcessedBIR,
  const BioAPI_INPUT_BIR *ReferenceTemplate,
  BioAPI_BIR_HANDLE *AdaptedBIR,
  BioAPI_BOOL *Result,
  BioAPI_FMR *FMRAchieved,
  BioAPI_DATA *Payload);
```

и параметры, указанные в таблице 72.

Таблица 72

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
MaxFMRRequested	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"ProcessedBIR_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"ReferenceTemplate"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_AdaptedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Result	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_FMRAchieved	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Payload	Входной параметр, исходящее значение	Входной параметр, входящее значение

Окончание таблицы 72

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
AdaptedBIR	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Result	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
FMRAchieved	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Payload	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_VerifyMatch** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_VerifyMatch**.

9.65.2 Выходные параметры вызова связанного процесса

9.65.2.1 Исходящее значение **AdaptedBIR** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **AdaptedBIR**.

9.65.2.2 Исходящее значение **Result** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае логическое значение, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BOOL**, на которую указывает собственный параметр **Result**.

9.65.2.3 Исходящее значение **FMRAchieved** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_FMR**, на которую указывает собственный параметр **FMRAchieved**.

9.65.2.4 Исходящее значение **Payload** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае набор байтов, представленный исходящим значением, должен быть записан в блок памяти достаточного объема, адрес и размер которого должны быть записаны в поля **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA,** на которую указывает собственный параметр **Payload**.

9.65.2.5 Целое число, представленное исходящим значением **return**, должно быть возвращено встроенной функцией.

9.65.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **AdaptedBIR**, должно быть присвоено значение, равное нулю;

b) переменной, на которую указывает собственный параметр **Result**, должно быть присвоено значение, равное нулю;

c) переменной, на которую указывает собственный параметр **FMRAchieved**, должно быть присвоено значение, равное нулю;

d) в поля **Data** и **Length** переменной, на которую указывает собственный параметр **Payload**, должны быть записаны значения, равные нулю;

e) собственная функция должна возвратить значение, равное нулю.

9.66 Функция BioSPI_IdentifyMatch

9.66.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_IdentifyMatch (
    BioAPI_HANDLE BSPHandle,
    BioAPI_FMR MaxFMRRequested,
    const BioAPI_INPUT_BIR *ProcessedBIR,
    const BioAPI_IDENTIFY_POPULATION *Population,
    uint32_t TotalNumberOfTemplates,
    BioAPI_BOOL Binning,
    uint32_t MaxNumberOfResults,
    uint32_t *NumberOfResults,
    BioAPI_CANDIDATE_ARRAY **Candidates,
    int32_t Timeout);
```

и параметры, указанные в таблице 73.

Таблица 73

Параметр	Вызов функции (модель испытаний на соответствие для ПБУ)	Вызов связанного процесса (модель испытаний на соответствие для инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
MaxFMRRequested	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"ProcessedBIR_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Определение заполнения» (см. 9.2.16)	Входной параметр, исходящее значение	Входной параметр, входящее значение
TotalNumberOfTemplates	Входной параметр, исходящее значение	Входной параметр, входящее значение
Binning	Входной параметр, исходящее значение	Входной параметр, входящее значение
MaxNumberOfResults	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_NumberOfResults	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Candidates	Входной параметр, исходящее значение	Входной параметр, входящее значение
Timeout	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumberOfResults	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Элементы группы параметров «Кандидат» (см. 9.2.17) с именами, начинающимися с **"Candidate_1_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Элементы группы параметров «Кандидат» с именами, начинающимися с **"Candidate_2_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение

Окончание таблицы 73

Параметр	Вызов функции (модель испытаний на соответствие для ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
Элементы группы параметров «Кандидат» с именами, начинающимися с **"Candidate_3_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Элементы группы параметров «Кандидат» с именами, начинающимися с **"Candidate_4_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_IdentifyMatch** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_IdentifyMatch**.

9.66.2 Выходные параметры вызова связанного процесса

9.66.2.1 Если собственный параметр **Candidates** имеет значение, отличающееся от NULL-значения, и целое число (например, **N**), представленное исходящим значением **NumberOfResults**, больше нуля, то для хранения массива **N** элементов типа **BioAPI_CANDIDATE** должен быть назначен блок памяти достаточного размера, а его адрес должен быть записан в переменную, на которую указывает собственный параметр **Candidates**.

9.66.2.2 Исходящее значение **NumberOfResults** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **uint32_t**, на которую указывает собственный параметр **NumberOfResults**.

9.66.2.3 Исходящее значение группы параметров «Кандидат» с префиксом **"Candidate_X_"** (где **X** = 1, 2, 3 или 4) должно быть обработано следующим образом. Если собственный параметр **Candidates** имеет NULL-значение или целое число, представленное исходящим значением **NumberOfResults**, меньше **X**, то исходящее значение должно быть проигнорировано. В противном случае значение типа **BioAPI_CANDIDATE**, представленное исходящим значением группы параметров, должно быть записано в элемент позиции **X** массива элементов типа **BioAPI_CANDIDATE**, на который указывает переменная, на которую, в свою очередь, указывает собственный параметр **Candidates**.

9.66.2.4 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.66.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **NumberOfResults**, должно быть присвоено значение, равное нулю;

b) переменной, на которую указывает собственный параметр **Candidates,** должно быть присвоено NULL-значение;

c) встроенная функция должна возвратить значение, равное нулю.

9.67 Функция BioSPI_Enroll

9.67.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_Enroll (
  BioAPI_HANDLE BSPHandle,
  BioAPI_BIR_PURPOSE Purpose,
  BioAPI_BIR_SUBTYPE Subtype,
  const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat,
  const BioAPI_INPUT_BIR *ReferenceTemplate,
  BioAPI_BIR_HANDLE *NewTemplate,
  const BioAPI_DATA *Payload,
```

int32_t Timeout,
BioAPI_BIR_HANDLE *AuditData,
BioAPI_UUID *TemplateUUID);

и параметры, указанные в таблице 74.

Таблица 74

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Purpose	Входной параметр, исходящее значение	Входной параметр, входящее значение
Subtype	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatOwner	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatType	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"ReferenceTemplate_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_NewTemplate	Входной параметр, исходящее значение	Входной параметр, входящее значение
Payload	Входной параметр, исходящее значение	Входной параметр, входящее значение
Timeout	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_AuditData	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_TemplateUUID	Входной параметр, исходящее значение	Входной параметр, входящее значение
NewTemplate	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
AuditData	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
TemplateUUID	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_Enroll** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_Enroll**.

9.67.2 Выходные параметры вызова связанного процесса

9.67.2.1 Исходящее значение **NewTemplate** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть про-

игнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **NewTemplate**.

9.67.2.2 Исходящее значение **AuditData** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое значение, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **AuditData**.

9.67.2.3 Исходящее значение **TemplateUUID** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае УУИД, представленный исходящим значением, должен быть записан в переменную типа **BioAPI_UUID,** на которую указывает собственный параметр **TemplateUUID**.

9.67.2.4 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.67.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **NewTemplate**, должно быть присвоено значение, равное нулю;

b) переменной, на которую указывает собственный параметр **AuditData**, должно быть присвоено значение, равное нулю;

c) переменной, на которую указывает собственный параметр **TemplateUUID**, должен быть присвоен УУИД «00000000-0000-0000-0000-000000000000»;

d) встроенная функция должна возвратить значение, равное нулю.

9.68 Функция BioSPI_Verify

9.68.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_Verify (
  BioAPI_HANDLE BSPHandle,
  BioAPI_FMR MaxFMRRequested,
  const BioAPI_INPUT_BIR *ReferenceTemplate,
  BioAPI_BIR_SUBTYPE Subtype,
  BioAPI_BIR_HANDLE *AdaptedBIR,
  BioAPI_BOOL *Result,
  BioAPI_FMR *FMRAchieved,
  BioAPI_DATA *Payload,
  int32_t Timeout,
  BioAPI_BIR_HANDLE *AuditData);
```

и параметры, указанные в таблице 75.

Таблица 75

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
MaxFMRRequested	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"ReferencedTemplate_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
Subtype	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_AdaptedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение

Окончание таблицы 75

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
no_Result	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_FMRAchieved	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Payload	Входной параметр, исходящее значение	Входной параметр, входящее значение
Timeout	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_AuditData	Входной параметр, исходящее значение	Входной параметр, входящее значение
AdaptedBIR	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Result	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
FMRAchieved	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Payload	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
AuditData	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_Verify** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_Verify**.

9.68.2 Выходные параметры вызова связанного процесса

9.68.2.1 Исходящее значение **AdaptedBIR** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **AdaptedBIR**.

9.68.2.2 Исходящее значение **Result** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае логическое значение, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BOOL**, на которую указывает собственный параметр **Result**.

9.68.2.3 Исходящее значение **FMRAchieved** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_FMR**, на которую указывает собственный параметр **FMRAchieved**.

9.68.2.4 Исходящее значение **Payload** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае набор байтов, представленный исходящим значением, должен быть записан в блок памяти достаточного объема, адрес и размер которого должны быть записаны в поля **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, на которую указывает собственный параметр **Payload**.

9.68.2.5 Исходящее значение **AuditData** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **AuditData**.

9.68.2.6 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.68.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **AdaptedBIR**, должно быть присвоено значение, равное нулю;

b) переменной, на которую указывает собственный параметр **Result**, должно быть присвоено значение, равное нулю;

c) переменной, на которую указывает собственный параметр **FMRAchieved**, должно быть присвоено значение, равное нулю;

d) в поля **Data** и **Length** переменной, на которую указывает собственный параметр **Payload**, должны быть записаны значения, равные нулю;

e) переменной, на которую указывает собственный параметр **AuditData**, должно быть присвоено значение, равное нулю;

f) собственная функция должна возвратить значение, равное нулю.

9.69 Функция BioSPI_Identify

9.69.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_Identify (
  BioAPI_HANDLE BSPHandle,
  BioAPI_FMR MaxFMRRequested,
  BioAPI_BIR_SUBTYPE Subtype,
  const BioAPI_IDENTIFY_POPULATION *Population,
  uint32_t TotalNumberOfTemplates,
  BioAPI_BOOL Binning,
  uint32_t MaxNumberOfResults,
  uint32_t *NumberOfResults,
  BioAPI_CANDIDATE **Candidates,
  int32_t Timeout,
  BioAPI_BIR_HANDLE *AuditData);
```

и параметры, указанные в таблице 76.

Таблица 76

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
MaxFMRRequested	Входной параметр, исходящее значение	Входной параметр, входящее значение
Subtype	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Определение заполнения» (см. 9.2.16)	Входной параметр, исходящее значение	Входной параметр, входящее значение
TotalNumberOfTemplates	Входной параметр, исходящее значение	Входной параметр, входящее значение
Binning	Входной параметр, исходящее значение	Входной параметр, входящее значение

Окончание таблицы 76

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
MaxNumberOfResults	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_NumberOfResults	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Candidates	Входной параметр, исходящее значение	Входной параметр, входящее значение
Timeout	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_AuditData	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumberOfResults	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Элементы группы параметров «Кандидат» (см. 9.2.17) с именами, начинающимися с **"Candidate_1_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Элементы группы параметров «Кандидат» с именами, начинающимися с **"Candidate_2_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Элементы группы параметров «Кандидат» с именами, начинающимися с **"Candidate_3_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Элементы группы параметров «Кандидат» с именами, начинающимися с **"Candidate_4_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
AuditData	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_Identify** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_Identify**.

9.69.2 Выходные параметры вызова связанного процесса

9.69.2.1 Если собственный параметр **Candidates** имеет значение, отличающееся от NULL-значения, и целое число (например, **N**), представленное исходящим значением **NumberOfResults**, больше нуля, то для хранения массива **N** элементов типа **BioAPI_CANDIDATE** должен быть выделен блок памяти достаточного размера, а его адрес должен быть записан в переменную, на которую указывает собственный параметр **Candidates**.

9.69.2.2 Исходящее значение **NumberOfResults** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **uint32_t**, на которую указывает собственный параметр **NumberOfResults**.

9.69.2.3 Исходящее значение группы параметров «Кандидат» с префиксом **«Candidate_X_»** (где **X** = 1, 2, 3 или 4) должно быть обработано следующим образом. Если собственный параметр **Candidates**

имеет NULL-значение или целое число, представленное исходящим значением **NumberOfResults**, меньше **X**, то исходящее значение должно быть проигнорировано. В противном случае значение типа **BioAPI_CANDIDATE**, представленное исходящим значением группы параметров, должно быть записано в элемент в позиции **X** массива элементов типа **BioAPI_CANDIDATE**, на который указывает переменная, на которую, в свою очередь, указывает собственный параметр **Candidates**.

9.69.2.4 Исходящее значение **AuditData** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **AuditData**.

9.69.2.5 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.69.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **NumberOfResults**, должно быть присвоено значение, равное нулю;

b) переменной, на которую указывает собственный параметр **Candidates**, должно быть присвоено NULL-значение;

c) переменной, на которую указывает собственный параметр **AuditData**, должно быть присвоено значение, равное нулю;

d) собственная функция должна возвратить значение, равное нулю.

9.70 Функция BioSPI_Import

9.70.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_Import (
  BioAPI_HANDLE BSPHandle,
  const BioAPI_DATA *InputData,
  const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *InputFormat,
  const BioAPI_BIR_BIOMETRIC_DATA_FORMAT *OutputFormat,
  BioAPI_BIR_PURPOSE Purpose,
  BioAPI_BIR_HANDLE *ConstructedBIR);
```

и параметры, указанные в таблице 77.

Таблица 77

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
InputData	Входной параметр, исходящее значение	Входной параметр, входящее значение
InputFormatOwner	Входной параметр, исходящее значение	Входной параметр, входящее значение
InputFormatType	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatOwner	Входной параметр, исходящее значение	Входной параметр, входящее значение
OutputFormatType	Входной параметр, исходящее значение	Входной параметр, входящее значение
Purpose	Входной параметр, исходящее значение	Входной параметр, входящее значение

Окончание таблицы 77

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
no_ConstructedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
ConstructedBIR	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_Import** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_Import**.

9.70.2 Выходные параметры вызова связанного процесса

9.70.2.1 Исходящее значение **ConstructedBIR** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **ConstructedBIR**.

9.70.2.2 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.70.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **ConstructedBIR**, должно быть присвоено значение, равное нулю;

b) собственная функция должна возвратить значение, равное нулю.

9.71 Функция BioSPI_PresetIdentifyPopulation

9.71.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_PresetIdentifyPopulation (
  BioAPI_HANDLE BSPHandle,
  const BioAPI_IDENTIFY_POPULATION *Population);
```

и параметры, указанные в таблице 78.

Таблица 78

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Определение заполнения» (см. 9.2.16)	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_PresetIdentifyPopulation** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_PresetIdentifyPopulation**.

9.71.2 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.71.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.72 Функция BioSPI_DbOpen

9.72.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_DbOpen (
  BioAPI_HANDLE BSPHandle,
  const BioAPI_UUID *DbUuid,
  BioAPI_DB_ACCESS_TYPE AccessRequest,
  BioAPI_DB_HANDLE *DbHandle,
  BioAPI_DB_MARKER_HANDLE *MarkerHandle);
```

и параметры, указанные в таблице 79.

Таблица 79

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Тип доступа» (см. 9.2.19)	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_DbHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_MarkerHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbHandle	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
MarkerHandle	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_DbOpen** аналогичны подобным описаниям для данной функции стандартного интерфейса **BioSPI_DbOpen**.

9.72.2 Выходные параметры вызова связанного процесса

9.72.2.1 Исходящее значение **DbHandle** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_DB_HANDLE**, на которую указывает собственный параметр **DbHandle**.

9.72.2.2 Исходящее значение **MarkerHandle** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_DB_MARKER_HANDLE**, на которую указывает собственный параметр **MarkerHandle**.

9.72.2.3 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.72.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **DbHandle**, должно быть присвоено значение, равное нулю;

b) переменной, на которую указывает собственный параметр **MarkerHandle**, должно быть присвоено значение, равное нулю;

c) собственная функция должна возвратить значение, равное нулю.

9.73 Функция BioSPI_DbClose

9.73.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_DbClose (
  BioAPI_HANDLE BSPHandle,
  BioAPI_DB_HANDLE DbHandle);
```

и параметры, указанные в таблице 80.

Таблица 80

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_DbClose** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_DbClose**.

9.73.2 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.73.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.74 Функция BioSPI_DbCreate

9.74.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_DbCreate (
  BioAPI_HANDLE BSPHandle,
  const BioAPI_UUID *DbUuid,
  uint32_t NumberOfRecords,
  BioAPI_DB_ACCESS_TYPE AccessRequest,
  BioAPI_DB_HANDLE *DbHandle);
```

и параметры, указанные в таблице 81.

Таблица 81

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumberOfRecords	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Тип доступа» (см. 9.2.19)	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_DbHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbHandle	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_DbCreate** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_DbCreate**.

9.74.2 Выходные параметры вызова связанного процесса

9.74.2.1 Исходящее значение **DbHandle** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_DB_HANDLE**, на которую указывает собственный параметр **DbHandle**.

9.74.2.2 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.74.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **DbHandle**, должно быть присвоено значение, равное нулю;

b) собственная функция должна возвратить значение, равное нулю.

9.75 Функция BioSPI_DbDelete

9.75.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_DbDelete (
  BioAPI_HANDLE BSPHandle,
  const BioAPI_UUID *DbUuid);
```

и параметры, указанные в таблице 82.

Таблица 82

Параметр	Вызов функции (модель испытаний на соответствие для ПБУ)	Вызов связанного процесса (модель испытаний на соответствие для инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Входной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_DbDelete** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_DbDelete**.

9.75.2 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.75.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.76 Функция BioSPI_DbSetMarker

9.76.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_DbSetMarker (
  BioAPI_HANDLE BSPHandle,
  BioAPI_DB_HANDLE DbHandle,
  const BioAPI_UUID *KeyValue,
  BioAPI_DB_MARKER_HANDLE MarkerHandle);
```

и параметры, указанные в таблице 83.

Таблица 83

Параметр	Вызов функции (модель испытаний на соответствие для ПБУ)	Вызов связанного процесса (модель испытаний на соответствие для инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
KeyValue	Входной параметр, исходящее значение	Входной параметр, входящее значение
MarkerHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_DbSetMarker** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_DbSetMarker**.

9.76.2 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.76.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.77 Функция BioSPI_DbFreeMarker

9.77.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_DbFreeMarker (
  BioAPI_HANDLE BSPHandle,
  BioAPI_DB_MARKER_HANDLE MarkerHandle);
```

и параметры, указанные в таблице 84.

Таблица 84

Параметр	Вызов функции (модель испытаний на соответствие для ПБУ)	Вызов связанного процесса (модель испытаний на соответствие для инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
MarkerHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_DbFreeMarker** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_DbFreeMarker**.

9.77.2 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.77.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.78 Функция BioSPI_DbStoreBIR

9.78.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_DbStoreBIR (
  BioAPI_HANDLE BSPHandle,
  const BioAPI_INPUT_BIR *BIRToStore,
  BioAPI_DB_HANDLE DbHandle,
  BioAPI_UUID *BirUuid);
```

и параметры, указанные в таблице 85.

Таблица 85

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Ввод ЗБИ» (см. 9.2.15) с именами, начинающимися с **"BIRToStore_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
DBHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_BirUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
BirUuid	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_DbStoreBIR** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_DbStoreBIR**.

9.78.2 Выходные параметры вызова связанного процесса

9.78.2.1 Исходящее значение **BirUuid** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае УУИД, представленный исходящим значением, должен быть записан в переменную типа **BioAPI_UUID**, на которую указывает собственный параметр **BirUuid**.

9.78.2.2 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.78.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **BIRUuid**, должен быть присвоен УУИД «00000000-0000-0000-0000-000000000000»;

b) собственная функция должна возвратить значение, равное нулю.

9.79 Функция BioSPI_DbGetBIR

9.79.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_DbGetBIR (
   BioAPI_HANDLE BSPHandle,
   BioAPI_DB_HANDLE DbHandle,
   const BioAPI_UUID *KeyValue,
   BioAPI_BIR_HANDLE *RetrievedBIR,
   BioAPI_DB_MARKER_HANDLE *MarkerHandle);
```

и параметры, указанные в таблице 86.

Таблица 86

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
KeyValue	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_RetrievedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_MarkerHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
RetrievedBIR	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
MarkerHandle	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_DbGetBIR** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_DbGetBIR**.

9.79.2 Выходные параметры вызова связанного процесса

9.79.2.1 Исходящее значение **RetrievedBIR** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BIR_HANDLE,** на которую указывает собственный параметр **RetrievedBIR**.

9.79.2.2 Исходящее значение **MarkerHandle** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_DB_MARKER_HANDLE**, на которую указывает собственный параметр **MarkerHandle**.

9.79.2.3 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.79.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **RetrievedBIR**, должно быть присвоено значение, равное нулю;

b) переменной, на которую указывает собственный параметр **MarkerHandle**, должно быть присвоено значение, равное нулю;

c) собственная функция должна возвратить значение, равное нулю.

9.80 Функция BioSPI_DbGetNextBIR

9.80.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_DbGetNextBIR (
  BioAPI_HANDLE BSPHandle,
  BioAPI_DB_HANDLE DbHandle,
  BioAPI_DB_MARKER_HANDLE MarkerHandle,
  BioAPI_BIR_HANDLE *RetrievedBIR,
  BioAPI_UUID *BirUuid);
```

и параметры, указанные в таблице 87.

Таблица 87

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
MarkerHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_RetrievedBIR	Входной параметр, исходящее значение	Входной параметр, входящее значение
No_BirUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
RetrievedBIR	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
BirUuid	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_DbGetNextBIR** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_DbGetNextBIR**.

9.80.2 Выходные параметры вызова связанного процесса

9.80.2.1 Исходящее значение **RetrievedBIR** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_BIR_HANDLE**, на которую указывает собственный параметр **RetrievedBIR**.

9.80.2.2 Исходящее значение **BirUuid** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае УУИД, представленный исходящим значением, должен быть записан в переменную типа **BioAPI_UUID**, на которую указывает собственный параметр **BirUuid**.

9.80.2.3 Целое число, представленное исходящим значением **return**, должно быть возвращено встроенной функцией.

9.80.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **RetrievedBIR**, должно быть присвоено значение, равное нулю;

b) переменной, на которую указывает собственный параметр **BirUuid**, должен быть присвоен УУИД «00000000-0000-0000-0000-000000000000»;

c) собственная функция должна возвратить значение, равное нулю.

9.81 Функция BioSPI_DbDeleteBIR

9.81.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_DbDeleteBIR (
  BioAPI_HANDLE BSPHandle,
  BioAPI_DB_HANDLE DbHandle,
  const BioAPI_UUID *KeyValue);
```

и параметры, указанные в таблице 88.

Таблица 88

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
DbHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
KeyValue	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_DbDeleteBIR** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_DbDeleteBIR**.

9.81.2 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.81.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.82 Функция BioSPI_SetPowerMode

9.82.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_SetPowerMode (
  BioAPI_HANDLE BSPHandle,
  BioAPI_UNIT_ID UnitID,
  BioAPI_POWER_MODE PowerMode);
```

и параметры, указанные в таблице 89.

Таблица 89

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
PowerMode	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_SetPowerMode** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_SetPowerMode**.

9.82.2 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.82.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.83 Функция BioSPI_SetIndicatorStatus

9.83.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_SetIndicatorStatus (
  BioAPI_HANDLE BSPHandle,
  BioAPI_UNIT_ID UnitID,
  BioAPI_INDICATOR_STATUS IndicatorStatus);
```

и параметры, указанные в таблице 90.

Таблица 90

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
IndicatorStatus	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_SetIndicatorStatus** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_SetIndicatorStatus**.

9.83.2 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.83.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.84 Функция BioSPI_GetIndicatorStatus

9.84.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_GetIndicatorStatus (
  BioAPI_HANDLE BSPHandle,
  BioAPI_UNIT_ID UnitID,
  BioAPI_INDICATOR_STATUS *IndicatorStatus);
```

и параметры, указанные в таблице 91.

Таблица 91

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_IndicatorStatus	Входной параметр, исходящее значение	Входной параметр, входящее значение
IndicatorStatus	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_GetIndicatorStatus** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_GetIndicatorStatus**.

9.84.2 Выходные параметры вызова связанного процесса

9.84.2.1 Исходящее значение **IndicatorStatus** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_INDICATOR_STATUS**, на которую указывает собственный параметр **IndicatorStatus**.

9.84.2.2 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.84.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **IndicatorStatus**, должно быть присвоено значение, равное нулю;

b) собственная функция должна возвратить значение, равное нулю.

9.85 Функция BioSPI_CalibrateSensor

9.85.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_CalibrateSensor (
  BioAPI_HANDLE BSPHandle,
  int32_t Timeout);
```

и параметры, указанные в таблице 92.

Таблица 92

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
Timeout	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_CalibrateSensor** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_CalibrateSensor**.

9.85.2 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.85.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.86 Функция BioSPI_Cancel

9.86.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_Cancel (
  BioAPI_HANDLE BSPHandle);
```

и параметры, указанные в таблице 93.

Таблица 93

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPHandle	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_Cancel** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_Cancel**.

9.86.2 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.86.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.87 Функция BioSPI_Free

9.87.1 Схема вызова функции

Данная функция принадлежит BioSPI интерфейсу и имеет следующий основной синтаксис вызова:

```
BioAPI_RETURN BioAPI
BioSPI_Free (
  void* Ptr);
```

и параметры, указанные в таблице 94.

Таблица 94

Параметр	Вызов функции (модель испытаний на соответствие ПБУ)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
Ptr	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова функции, выходных параметров вызова функции и входных параметров вызова связанного процесса функции **BioAPI_Free** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_Free**.

9.87.2 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.87.3 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.88 Функция BioAPI_EventHandler

9.88.1 Схема вызова функции

Данная функция принадлежит интерфейсу обратного вызова приложения и имеет следующий основной синтаксис вызова:

```
typedef BioAPI_RETURN (*BioAPI_EventHandler) (
  const BioAPI_UUID *BSPUuid,
  BioAPI_UNIT_ID UnitID,
  void* AppNotifyCallbackCtx,
  const BioAPI_UNIT_SCHEMA *UnitSchema,
  BioAPI_EVENT EventType);
```

и параметры, указанные в таблице 95.

Таблица 95

Параметр	Вызов связанного процесса (модель испытаний на соответствие приложений)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BSPUuid	Входной параметр, входящее значение	Входной параметр, входящее значение
UnitID	Входной параметр, входящее значение	Входной параметр, входящее значение
AppNotifyCallbackCtx	Входной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема элементов» (см. 9.2.12) с именами, начинающимися с **"UnitSchema_"**	Входной параметр, входящее значение	Входной параметр, входящее значение

Окончание таблицы 95

Параметр	Вызов связанного процесса (модель испытаний на соответствие приложений)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
EventType	Входной параметр, входящее значение	Входной параметр, входящее значение
return	Входной параметр, входящее значение	Выходной параметр, исходящее значение

9.88.2 Ограничения на параметры

9.88.2.1 Исходящим значением **BSPUuid** должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.88.2.2 Исходящими значениями **UnitID**, **AppNotifyCallbackCtx** и **EventType** должны быть корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале.

9.88.2.3 Исходящим значением группы параметров «Схема элементов» должно быть корректно представленное значение собственного типа **BioAPI_UNIT_SCHEMA**. Входящим значением должно быть или стандартно представленное значение этого типа, или набор пустых строк.

9.88.2.4 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.88.3 Входные параметры вызова связанного процесса

9.88.3.1 Входящим значением **BSPUuid** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **BSPUuid**.

9.88.3.2 Входящими значениями **UnitID**, **AppNotifyCallbackCtx** и **EventType** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.88.3.3 Входящим значением группы параметров «Схема элементов» должно быть стандартно представленное (см. 9.2.12.5) значение типа **BioAPI_UNIT_SCHEMA** в переменной, на которую указывает собственный параметр **UnitSchema**.

9.88.3.4 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.88.4 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.88.5 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.89 Функция BioAPI_GUI_STATE_CALLBACK

9.89.1 Схема вызова функции

Данная функция принадлежит интерфейсу обратного вызова приложения и имеет следующий основной синтаксис вызова:

```
typedef BioAPI_RETURN
(BioAPI *BioAPI_GUI_STATE_CALLBACK) (
  void *GuiStateCallbackCtx,
  BioAPI_GUI_STATE GuiState,
  BioAPI_GUI_RESPONSE *Response,
  BioAPI_GUI_MESSAGE Message,
  BioAPI_GUI_PROGRESS Progress,
  const BioAPI_GUI_BITMAP *SampleBuffer);
```

и параметры, указанные в таблице 96.

Таблица 96

Параметр	Вызов связанного процесса (модель испытаний на соответствие приложений)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
GuiStateCallbackCtx	Входной параметр, входящее значение	Входной параметр, входящее значение
Элементы группы параметров «Состояние ГИП» (см. 9.2.18)	Входной параметр, входящее значение	Входной параметр, входящее значение
no_Response	Входной параметр, входящее значение	Входной параметр, входящее значение
Message	Входной параметр, входящее значение	Входной параметр, входящее значение
Progress	Входной параметр, входящее значение	Входной параметр, входящее значение
BitmapWidth	Входной параметр, входящее значение	Входной параметр, входящее значение
BitmapHeight	Входной параметр, входящее значение	Входной параметр, входящее значение
Bitmap	Входной параметр, входящее значение	Входной параметр, входящее значение
Response	Входной параметр, входящее значение	Выходной параметр, исходящее значение
return	Входной параметр, входящее значение	Выходной параметр, исходящее значение

9.89.2 Ограничения на параметры

9.89.2.1 Исходящими значениями **GuiStateCallbackCtx** и **Message** должны быть корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале.

9.89.2.2 Исходящим значением группы параметров «Состояние ГИП» должно быть корректно представленное значение собственного типа **BioAPI_GUI_STATE**. Входящим значением должно быть стандартно представленное значение этого типа.

9.89.2.3 Исходящим значением **no_Response** должно быть или корректно представленное логическое значение (см. 7.5), или пустая строка. Входящим значением должно быть стандартно представленное логическое значение (см. 7.5.2).

9.89.2.4 Исходящим значением **Progress** должно быть корректно представленное целое число в интервале от 0 до 255. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.89.2.5 Исходящими значениями **BitmapWidth** и **BitmapHeight** должны быть или корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295, или пустые строки. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале.

9.89.2.6 Исходящим значением **Bitmap** должен быть корректно представленный набор байтов (см. 7.7). Входящим значением должен быть стандартно представленный набор байтов (см. 7.7.2).

9.89.2.7 Исходящим значением **Response** должно быть или корректно представленное целое число в интервале от 0 до 255, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.89.2.8 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.89.3 Входные параметры вызова связанного процесса

9.89.3.1 Входящими значениями **GuiStateCallbackCtx**, **Message** и **Progress** должны быть стандартно представленные целые числа в собственных параметрах с теми же именами.

9.89.3.2 Входящим значением группы параметров «Состояние ГИП» должно быть стандартно представленное (см. 9.2.18.5) целое число в собственном параметре **GuiState**.

9.89.3.3 Входящим значением **no_Response** должно быть **"true"**, если собственный параметр **Response** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.89.3.4 Входящее значение **Bitmap** определяют следующим образом. Если собственный параметр **SampleBuffer** имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должен быть стандартно представленный (см. 7.7.2) набор байтов в блоке памяти, адрес и длина которого находятся в полях **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, на которую указывает поле **Bitmap** переменной типа **BioAPI_GUI_BITMAP**, на которую, в свою очередь, указывает собственный параметр **SampleBuffer**.

9.89.3.5 Входящие значения **BitmapWidth** и **BitmapHeight** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящие значения должны быть равны «0». В противном случае входящими значениями должны быть стандартно представленные (см. 7.4.3) целые числа в полях **Width** и **Height** (соответственно) переменной типа **BioAPI_GUI_BITMAP**, на которую указывает собственный параметр **SampleBuffer**.

9.89.3.6 Входящее значение **Response** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное значение в переменной типа **BioAPI_GUI_RESPONSE**, на которую указывает собственный параметр **Response**.

9.89.3.7 Исходящее значение **return** должно быть стандартно представленным целым числом, возвращенным собственной функцией.

9.89.4 Выходные параметры вызова связанного процесса

9.89.4.1 Исходящее значение **Response** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **BioAPI_GUI_RESPONSE**, на которую указывает собственный параметр **Response**.

9.89.4.2 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.89.5 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **Response**, должно быть присвоено значение, равное нулю;

b) собственная функция должна возвратить значение, равное нулю.

9.90 Функция BioAPI_GUI_STREAMING_CALLBACK

9.90.1 Схема вызова функции

Данная функция принадлежит интерфейсу обратного вызова приложения и имеет следующий основной синтаксис вызова:

```
typedef BioAPI_RETURN (BioAPI *BioAPI_GUI_STREAMING_CALLBACK) (
  void *GuiStreamingCallbackCtx,
  const BioAPI_GUI_BITMAP *Bitmap);
```

и параметры, указанные в таблице 97.

Таблица 97

Параметр	Вызов связанного процесса (модель испытаний на соответствие приложений)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
GuiStreamingCallbackCtx	Входной параметр, входящее значение	Входной параметр, входящее значение
BitmapWidth	Входной параметр, входящее значение	Входной параметр, входящее значение

Окончание таблицы 97

Параметр	Вызов связанного процесса (модель испытаний на соответствие приложений)	Вызов связанного процесса (модель испытаний на соответствие инфраструктур)
BitmapHeight	Входной параметр, входящее значение	Входной параметр, входящее значение
Bitmap	Входной параметр, входящее значение	Входной параметр, входящее значение
return	Входной параметр, входящее значение	Входной параметр, входящее значение

9.90.2 Ограничения на параметры

9.90.2.1 Исходящим значением **GuiStreamingCallbackCtx** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.90.2.2 Исходящими значениями **BitmapWidth** и **BitmapHeight** должны быть или корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295, или пустые строки. Входящими значениями должны быть стандартно представленные целые числа в том же интервале.

9.90.2.3 Исходящим значением **Bitmap** должен быть корректно представленный набор байтов (см. 7.7). Входящим значением должен быть стандартно представленный набор байтов (см. 7.7.2).

9.90.2.4 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.90.3 Входные параметры вызова связанного процесса

9.90.3.1 Входящим значением **GuiStreamingCallbackCtx** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.90.3.2 Входящее значение **Bitmap** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должен быть стандартно представленный (см. 7.7.2) набор байтов в блоке памяти, адрес и длина которого находятся в полях **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**, на которую указывает поле **Bitmap** переменной типа **BioAPI_GUI_BITMAP**, на которую, в свою очередь, указывает собственный параметр **Bitmap**.

9.90.3.3 Входящие значения **BitmapWidth** и **BitmapHeight** определяют следующим образом. Если собственный параметр **Bitmap** имеет NULL-значение, то входящие значения должны быть равны «0». В противном случае входящими значениями должны быть стандартно представленные (см. 7.4.3) целые числа в полях **Width** и **Height** (соответственно) переменной типа **BioAPI_GUI_BITMAP**, на которую указывает собственный параметр **Bitmap**.

9.90.3.4 Входящим значением **return** должно быть стандартно представленное целое число, возвращенное собственной функцией.

9.90.4 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.90.5 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.91 Функция BioSPI_EventHandler

9.91.1 Схема вызова функции

Данная функция принадлежит интерфейсу обратного вызова инфраструктуры и имеет следующий основной синтаксис вызова:

```
typedef BioAPI_RETURN (*BioSPI_EventHandler) (
  const BioAPI_UUID *BSPUuid,
  BioAPI_UNIT_ID UnitID,
  const BioAPI_UNIT_SCHEMA *UnitSchema,
  BioAPI_EVENT EventType);
```

и параметры, указанные в таблице 98.

Таблица 98

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие ПБУ)
BSPUuid	Входной параметр, исходящее значение	Входной параметр, входящее значение
UnitID	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема элементов» (см. 9.2.12) с именами, начинающимися с **"UnitSchema_"**	Входной параметр, исходящее значение	Входной параметр, входящее значение
EventType	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

9.91.2 Ограничения на параметры

9.91.2.1 Исходящим значением **BSPUuid** должен быть корректно представленный УУИД (см. 7.6). Входящим значением должен быть стандартно представленный УУИД (см. 7.6.3).

9.91.2.2 Исходящими значениями **UnitID** и **EventType** должны быть корректно представленные целые числа (см. 7.4) в интервале от 0 до 4294967295. Входящими значениями должны быть стандартно представленные целые числа (см. 7.4.3) в том же интервале.

9.91.2.3 Исходящим значением группы параметров «Схема элементов» должно быть или корректно представленное значение собственного типа **BioAPI_UNIT_SCHEMA**, или набор пустых строк. Входящим значением должно быть или стандартно представленное значение этого типа, или набор пустых строк.

9.91.2.4 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.91.3 Входные параметры вызова функции

9.91.3.1 УУИД, представленный исходящим значением **BSPUuid**, должен быть записан в переменную типа **BioAPI_UUID**, адрес которой должен быть присвоен собственному параметру **BSPUuid**.

9.91.3.2 Целые числа, представленные исходящими значениями **UnitID** и **EventType**, должны быть присвоены собственным параметрам с тем же именем.

9.91.3.3 Если исходящим значением группы параметров «Схема элементов» является набор пустых строк, то собственному параметру **UnitSchema** должно быть присвоено NULL-значение. В противном случае значение типа **BioAPI_UNIT_SCHEMA**, представленное (см. 9.2.12.4) исходящим значением группы параметров «Схема элементов», должно быть записано в переменную типа **BioAPI_UNIT_SCHEMA**, адрес которой должен быть сохранен в собственном параметре **UnitSchema**.

9.91.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.91.5 Входные параметры вызова связанного процесса

9.91.5.1 Входящим значением **BSPUuid** должен быть стандартно представленный УУИД в переменной типа **BioAPI_UUID**, на которую указывает собственный параметр **BSPUuid**.

9.91.5.2 Входящими значениями **UnitID** и **EventType** должны быть стандартно представленные целые числа в собственных параметрах с тем же именем.

9.91.5.3 Если собственный параметр **UnitSchema** имеет NULL-значение, то входящим значением группы параметров «Схема элементов» должен быть набор пустых строк. В противном случае входящим значением должно быть стандартно представленное (см. 9.2.12.5) значение типа **BioAPI_UNIT_SCHEMA** в переменной, на которую указывает собственный параметр **UnitSchema**.

9.91.6 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.91.7 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то собственная функция должна возвратить значение, равное нулю.

9.92 Функция BioSPI_GUI_STATE_CALLBACK

9.92.1 Схема вызова функции

Данная функция принадлежит интерфейсу обратного вызова инфраструктуры и имеет следующий основной синтаксис вызова:

```
typedef BioAPI_RETURN (BioAPI *BioAPI_GUI_STATE_CALLBACK) (
  void *GuiStateCallbackCtx,
  BioAPI_GUI_STATE GuiState,
  BioAPI_GUI_RESPONSE *Response,
  BioAPI_GUI_MESSAGE Message,
  BioAPI_GUI_PROGRESS Progress,
  const BioAPI_GUI_BITMAP *SampleBuffer);
```

и параметры, указанные в таблице 99.

Таблица 99

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие ПБУ)
GuiStateCallbackCtx	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Состояние ГИП» (см. 9.2.18)	Входной параметр, исходящее значение	Входной параметр, входящее значение
no_Response	Входной параметр, исходящее значение	Входной параметр, входящее значение
Message	Входной параметр, исходящее значение	Входной параметр, входящее значение
Progress	Входной параметр, исходящее значение	Входной параметр, входящее значение
BitmapWidth	Входной параметр, исходящее значение	Входной параметр, входящее значение
BitmapHeight	Входной параметр, исходящее значение	Входной параметр, входящее значение
Bitmap	Входной параметр, исходящее значение	Входной параметр, входящее значение
Response	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова связанного процесса, выходных параметров вызова связанного процесса и возвращаемого значения по умолчанию для функции **BioAPI_GUI_STATE_CALLBACK** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_GUI_STATE_CALLBACK**.

9.92.2 Входные параметры вызова функции

9.92.2.1 Целые числа, представленные исходящими значениями **GuiStateCallbackCtx**, **Message** и **Progress**, должны быть присвоены собственным параметрам с теми же именами.

9.92.2.2 Целое число, представленное (см. 9.2.18.4) исходящим значением группы параметров «Состояние ГИП», должно быть присвоено собственному параметру **GuiState**.

9.92.2.3 Если исходящим значением **no_Response** является **"true"**, то собственному параметру **Response** должно быть присвоено NULL-значение, в противном случае собственному параметру должно быть присвоено значение адреса переменной типа **BioAPI_GUI_Response**.

9.92.2.4 Если значение **Bitmap** является пустой строкой, то собственному параметру **SampleBuffer** должно быть присвоено NULL-значение. В противном случае:

a) целые числа, представленные исходящими значениями **BitmapWidth** и **BitmapHeight** (или ноль, если исходящими значениями являются пустые строки), должны быть записаны в поля **Width** и **Height** (соответственно) переменной типа **BioAPI_GUI_BITMAP**, адрес которой должен быть присвоен собственному параметру **SampleBuffer**;

b) набор байтов, представленный исходящим значением **Bitmap**, должен быть записан в блок памяти достаточного объема;

c) адрес и длина блока памяти должны быть записаны в поля **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**;

d) адрес этой переменной должен быть записан в поле **Bitmap** переменной типа **BioAPI_GUI_BITMAP**, указанной в перечислении a).

9.92.3 Выходные параметры вызова функции

Входящее значение **Response** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **BioAPI_GUI_RESPONSE**, на которую указывает собственный параметр **Response**.

9.93 Функция BioSPI_GUI_STREAMING_CALLBACK

9.93.1 Схема вызова функции

Данная функция принадлежит интерфейсу обратного вызова инфраструктуры и имеет следующий основной синтаксис вызова:

```
typedef BioAPI_RETURN (BioAPI *BioAPI_GUI_STREAMING_CALLBACK) (
  void *GuiStreamingCallbackCtx,
  const BioAPI_GUI_BITMAP *Bitmap);
```

и параметры, указанные в таблице 100.

Таблица 100

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие ПБУ)
GuiStreamingCallbackCtx	Входной параметр, исходящее значение	Входной параметр, входящее значение
BitmapWidth	Входной параметр, исходящее значение	Входной параметр, входящее значение
BitmapHeight	Входной параметр, исходящее значение	Входной параметр, входящее значение
Bitmap	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

Описания ограничений на параметры, входных параметров вызова связанного процесса, выходных параметров вызова связанного процесса и возвращаемого значения по умолчанию для функции **BioAPI_GUI_STREAMING_CALLBACK** аналогичны описаниям данной функции стандартного интерфейса **BioSPI_GUI_STREAMING_CALLBACK**.

9.93.2 Входные параметры вызова функции

9.93.2.1 Целое число, представленное исходящим значением **GuiStreamingCallbackCtx**, должно быть присвоено собственному параметру с тем же именем.

9.93.2.2 Если значение **Bitmap** является пустой строкой, то собственному параметру **Bitmap** должно быть присвоено NULL-значение. В противном случае:

a) целые числа, представленные исходящими значениями **BitmapWidth** и **BitmapHeight** (или ноль, если исходящими значениями являются пустые строки), должны быть записаны в поля **Width** и **Height** (соответственно) переменной типа **BioAPI_GUI_BITMAP**, адрес которой должен быть присвоен собственному параметру **Bitmap**;

b) набор байтов, представленный исходящим значением **Bitmap**, должен быть записан в блок памяти достаточного объема;

c) адрес и длина блока памяти должны быть записаны в поля **Data** и **Length** (соответственно) переменной типа **BioAPI_DATA**;

d) адрес этой переменной должен быть записан в поле **Bitmap** переменной типа **BioAPI_GUI_BITMAP**, указанной в перечислении a).

9.93.3 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.94 Функция BioSPI_BFP_ENUMERATION_HANDLER

9.94.1 Схема вызова функции

Данная функция принадлежит интерфейсу обратного вызова приложения и имеет следующий основной синтаксис вызова:

```
typedef BioAPI_RETURN (BioAPI *BioSPI_BFP_ENUMERATION_HANDLER) (
    BioAPI_BFP_SCHEMA **BFPSchemaArray,
    uint32_t *NumberOfElements);
```

и параметры, указанные в таблице 101.

Таблица 101

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие ПБУ)
no_BFPSchemaArray	Входной параметр, исходящее значение	Входной параметр, входящее значение
Элементы группы параметров «Схема ПБФ» (см. 9.2.11) с именами, начинающимися с **"BFPSchema_1_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Элементы группы параметров «Схема ПБФ» с именами, начинающимися с **"BFPSchema_2_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Элементы группы параметров «Схема ПБФ» с именами, начинающимися с **"BFPSchema_3_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
Элементы группы параметров «Схема ПБФ» с именами, начинающимися с **"BFPSchema_4_"**	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
no_NumberOfElements	Входной параметр, исходящее значение	Входной параметр, входящее значение
NumberOfElements	Выходной параметр, входящее значение	Выходной параметр, исходящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

9.94.2 Ограничения на параметры

9.94.2.1 Исходящими значениями **no_BFPSchemaArray** и **no_NumberOfElements** должны быть или корректно представленные логические значения (см. 7.5), или пустые строки. Входящими значениями должны быть стандартно представленные логические значения (см. 7.5.2).

9.94.2.2 Исходящим значением каждой группы параметров «Схема ПБФ» должно быть корректно представленное значение собственного типа **BioAPI_BFP_SCHEMA**. Входящим значением должно быть или стандартно представленное значение этого типа, или набор пустых строк.

9.94.2.3 Исходящим значением **NumberOfElements** должно быть или корректно представленное целое число в интервале от 0 до 4294967295, или пустая строка. Входящим значением должно быть или стандартно представленное целое число в том же интервале, или пустая строка.

9.94.2.4 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.94.3 Входные параметры вызова функции

9.94.3.1 Если исходящим значением **no_BFPSchemaArray** является **"true"**, то собственному параметру **BFPSchemaArray** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **BioAPI_BFP_SCHEMA*** (указатель).

9.94.3.2 Если исходящим значением **no_NumberOfElements** является **"true"**, то собственному параметру **NumberOfElements** должно быть присвоено NULL-значение, в противном случае ему должно быть присвоено значение адреса переменной типа **uint32_t**.

9.94.4 Выходные параметры вызова функции

9.94.4.1 Входящее значение группы параметров «Схема ПБФ» с префиксом **"BFPSchema_X_"** (где **X** = 1, 2, 3 или 4) определяют следующим образом. Если собственный параметр **BFPSchemaArray** или переменная, на которую указывает этот собственный параметр имеют NULL-значение, или собственный параметр **NumberOfElements** имеет NULL-значение, или значение целочисленной переменной, на которую указывает этот собственный параметр, меньше **X**, то входящим значением должен быть набор пустых строк. В противном случае входящим значением должно быть стандартно представленное (см. 9.2.11.5) значение типа **BioAPI_BFP_SCHEMA** в позиции **X** массива, на которую указывает переменная, на которую, в свою очередь, указывает собственный параметр **BFPSchemaArray**.

9.94.4.2 Входящее значение **NumberOfElements** определяют следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то входящим значением должна быть пустая строка. В противном случае входящим значением должно быть стандартно представленное целое число в переменной типа **uint32_t**, на которую указывает собственный параметр **NumberOfElements**.

9.94.5 Входные параметры вызова связанного процесса

9.94.5.1 Входящим значением **no_BFPSchemaArray** является **"true"**, если собственный параметр **BFPSchemaArray** имеет NULL-значение, в противном случае входящим значением должно должно быть **"false"**.

9.94.5.2 Входящим значением **no_NumberOfElements** является **"true"**, если собственный параметр **NumberOfElements** имеет NULL-значение, в противном случае входящим значением должно быть **"false"**.

9.94.6 Выходные параметры вызова связанного процесса

9.94.6.1 Если собственный параметр **BFPSchemaArray** имеет значение, отличающееся от NULL-значения, и целое число (например, **N**), представленное исходящим значением **NumberOfElements**, больше нуля, то для хранения массива **N** элементов типа **BioAPI_BFP_SCHEMA** должен быть выделен блок памяти достаточного размера, а его адрес должен быть записан в переменную, на которую указывает собственный параметр **BFPSchemaArray**.

9.94.6.2 Исходящее значение группы параметров «Схема элементов» с префиксом **BFPSchema_X_** (где **X** = 1, 2, 3 или 4) должно быть обработано следующим образом. Если собственный параметр **BFPSchemaArray** имеет NULL-значение или целое число, представленное исходящим значением **NumberOfElements**, меньше **X**, то исходящее значение группы параметров должно быть проигнорировано. В противном случае значение типа **BioAPI_BFP_SCHEMA**, представленное исходящим значением группы параметров, должно быть записано в элемент в позиции **X** массива элементов типа **BioAPI_BFP_SCHEMA**, на который указывает переменная, на которую, в свою очередь, указывает собственный параметр **BFPSchemaArray**.

9.94.6.3 Исходящее значение **NumberOfElements** должно быть обработано следующим образом. Если собственный параметр с тем же именем имеет NULL-значение, то исходящее значение должно быть проигнорировано. В противном случае целое число, представленное исходящим значением, должно быть записано в переменную типа **uint32_t**, на которую указывает собственный параметр **NumberOfElements**.

9.94.6.4 Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.94.7 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то:

a) переменной, на которую указывает собственный параметр **NumberOfElements**, должно быть присвоено значение, равное нулю;

b) переменной, на которую указывает собственный параметр **BFPSchemaArray**, должно быть присвоено NULL-значение;

c) встроенная функция должна возвратить значение, равное нулю.

9.95 Функция BioSPI_MEMORY_FREE_HANDLER

9.95.1 Схема вызова функции

Данная функция принадлежит интерфейсу обратного вызова приложения и имеет следующий основной синтаксис вызова:

```
typedef BioAPI_RETURN (BioAPI *BioSPI_MEMORY_FREE_HANDLER) (
  void* Ptr);
```

и параметры, указанные в таблице 102.

Таблица 102

Параметр	Вызов функции (модель испытаний на соответствие инфраструктур)	Вызов связанного процесса (модель испытаний на соответствие ПБУ)
Ptr	Входной параметр, исходящее значение	Входной параметр, входящее значение
return	Возвращаемое значение, входящее значение	Выходной параметр, исходящее значение

9.95.2 Ограничения на параметры

9.95.2.1 Исходящим значением **Ptr** должно быть корректно представленное целое число (см. 7.4) в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число (см. 7.4.3) в том же интервале.

9.95.2.2 Исходящим значением **return** должно быть корректно представленное целое число в интервале от 0 до 4294967295. Входящим значением должно быть стандартно представленное целое число в том же интервале.

9.95.3 Входные параметры вызова функции

Целое число, представленное исходящим значением **Ptr**, должно быть присвоено собственному параметру с тем же именем.

9.95.4 Выходные параметры вызова функции

Выходные параметры отсутствуют.

9.95.5 Входные параметры вызова связанного процесса

Входящим значением **Ptr** должно быть стандартно представленное целое число в собственном параметре с тем же именем.

9.95.6 Выходные параметры вызова связанного процесса

Целое число, представленное исходящим значением **return**, должно быть возвращено собственной функцией.

9.95.7 Возвращаемое значение по умолчанию

Если ни один процесс не связан с этой функцией стандартного интерфейса BioAPI, то встроенная функция должна возвратить значение, равное нулю.

10 Встроенные переменные

10.1 Переменные, имеющие постоянное значение

Переменные, имеющие постоянные значения (константы):

__BioAPI_NO_PRODUCT_OWNER_AVAILABLE	"0"
__BioAPI_NO_PRODUCT_TYPE_AVAILABLE	"0"
__BioAPI_NO_TYPE_AVAILABLE	"0"
__BioAPI_NO_PURPOSE_AVAILABLE	"0"
__BioAPI_NO_SUBTYPE_AVAILABLE	"0"
__BioAPI_NO_YEAR_AVAILABLE	"0"
__BioAPI_NO_MONTH_AVAILABLE	"0"
__BioAPI_NO_DAY_AVAILABLE	"0"
__BioAPI_NO_HOUR_AVAILABLE	"99"
__BioAPI_NO_MINUTE_AVAILABLE	"99"
__BioAPI_NO_SECOND_AVAILABLE	"99"
__BioAPI_DONT_CARE	"0"
__BioAPI_DONT_INCLUDE	"4294967295"
__BioAPI_BIR_DATA_TYPE_RAW	"1"
__BioAPI_BIR_DATA_TYPE_INTERMEDIATE	"2"
__BioAPI_BIR_DATA_TYPE_PROCESSED	"4"
__BioAPI_INVALID_BIR_HANDLE	"–1"
__BioAPI_UNSUPPORTED_BIR_HANDLE	"–2"
__BioAPI_PURPOSE_VERIFY	"1"
__BioAPI_PURPOSE_IDENTIFY	"2"
__BioAPI_PURPOSE_ENROLL	"3"
__BioAPI_PURPOSE_ENROLL_FOR_VERIFICATION_ONLY	"4"
__BioAPI_PURPOSE_ENROLL_FOR_IDENTIFICATION_ONLY	"5"
__BioAPI_PURPOSE_AUDIT	"6"
__BioAPI_FALSE	"0"
__BioAPI_TRUE	"1"
__BioAPI_CATEGORY_ARCHIVE	"1"
__BioAPI_CATEGORY_MATCHING_ALG	"2"
__BioAPI_CATEGORY_PROCESSING_ALG	"4"
__BioAPI_CATEGORY_SENSOR	"8"
__BioAPI_DB_INVALID_HANDLE	"–1"
__BioAPI_DB_DEFAULT_HANDLE	"0"
__BioAPI_DB_DEFAULT_UUID_PTR	"0"
__BioAPI_NOT_SET	"–1"
__BioAPI_NOT_SUPPORTED	"–2"
__BioAPI_CAPTURE_SAMPLE	"1"
__BioAPI_CANCEL	"2"
__BioAPI_CONTINUE	"3"
__BioAPI_VALID_SAMPLE	"4"
__BioAPI_INVALID_SAMPLE	"5"
__BioAPI_DB_TYPE	"1"
__BioAPI_ARRAY_TYPE	"2"
__BioAPI_PRESET_ARRAY_TYPE	"3"
__BioAPI_INDICATOR_ACCEPT	"1"
__BioAPI_INDICATOR_REJECT	"2"

```
__BioAPI_INDICATOR_READY                    "3"
__BioAPI_INDICATOR_BUSY                     "4"
__BioAPI_INDICATOR_FAILURE                  "5"

__BioAPI_DATABASE_ID_INPUT                  "1"
__BioAPI_BIR_HANDLE_INPUT                   "2"
__BioAPI_FULLBIR_INPUT                      "3"

__BioAPI_NOTIFY_INSERT                      "1"
__BioAPI_NOTIFY_REMOVE                      "2"
__BioAPI_NOTIFY_FAULT                       "3"
__BioAPI_NOTIFY_SOURCE_PRESENT              "4"
__BioAPI_NOTIFY_SOURCE_REMOVED              "5"

__BioAPI_POWER_NORMAL                       "1"
__BioAPI_POWER_DETECT                       "2"
__BioAPI_POWER_SLEEP                        "3"

__BioAPI_OK                                 "0"

__BioAPI_INSTALL_ACTION_INSTALL             "0"
__BioAPI_INSTALL_ACTION_REFRESH             "1"
__BioAPI_INSTALL_ACTION_UNINSTALL           "2"

#define BIOAPI_FRAMEWORK_ERROR              "0"
#define BIOAPI_BSP_ERROR                    "16777216"
#define BIOAPI_UNIT_ERROR                   "33554432"

__BioAPIERR_FW_INTERNAL_ERROR               "257"
__BioAPIERR_FW_MEMORY_ERROR                 "258"
__BioAPIERR_FW_INVALID_POINTER              "259"
__BioAPIERR_FW_INVALID_INPUT_POINTER        "260"
__BioAPIERR_FW_INVALID_OUTPUT_POINTER       "261"
__BioAPIERR_FW_FUNCTION_NOT_SUPPORTED       "262"
__BioAPIERR_FW_OS_ACCESS_DENIED             "263"
__BioAPIERR_FW_FUNCTION_FAILED              "264"
__BioAPIERR_FW_INVALID_UUID                 "265"
__BioAPIERR_FW_INCOMPATIBLE_VERSION         "266"
__BioAPIERR_FW_INVALID_DATA                 "267"
__BioAPIERR_FW_UNABLE_TO_CAPTURE            "268"
__BioAPIERR_FW_TOO_MANY_HANDLES             "269"
__BioAPIERR_FW_TIMEOUT_EXPIRED              "270"
__BioAPIERR_FW_INVALID_BIR                  "271"
__BioAPIERR_FW_BIR_SIGNATURE_FAILURE        "272"
__BioAPIERR_FW_UNABLE_TO_STORE_PAYLOAD      "273"
__BioAPIERR_FW_NO_INPUT_BIRS                "274"
__BioAPIERR_FW_UNSUPPORTED_FORMAT           "275"
__BioAPIERR_FW_UNABLE_TO_IMPORT             "276"
__BioAPIERR_FW_INCONSISTENT_PURPOSE         "277"
__BioAPIERR_FW_BIR_NOT_FULLY_PROCESSED      "278"
__BioAPIERR_FW_PURPOSE_NOT_SUPPORTED        "279"
__BioAPIERR_FW_USER_CANCELLED               "280"
__BioAPIERR_FW_UNIT_IN_USE                  "281"
__BioAPIERR_FW_INVALID_BSP_HANDLE           "282"
__BioAPIERR_FW_FRAMEWORK_NOT_INITIALIZED    "283"
__BioAPIERR_FW_INVALID_BIR_HANDLE           "284"
__BioAPIERR_FW_CALIBRATION_NOT_SUCCESSFUL   "285"
__BioAPIERR_FW_PRESET_BIR_DOES_NOT_EXIST    "286"
__BioAPIERR_FW_BIR_DECRYPTION_FAILURE       "287"
```

__BioAPIERR_FW_COMPONENT_FILE_REF_NOT_FOUND	"513"
__BioAPIERR_FW_BSP_LOAD_FAIL	"514"
__BioAPIERR_FW_BSP_NOT_LOADED	"515"
__BioAPIERR_FW_UNIT_NOT_INSERTED	"516"
__BioAPIERR_FW_INVALID_UNIT_ID	"517"
__BioAPIERR_FW_INVALID_CATEGORY	"518"
__BioAPIERR_FW_INVALID_DB_HANDLE	"768"
__BioAPIERR_FW_UNABLE_TO_OPEN_DATABASE	"769"
__BioAPIERR_FW_DATABASE_IS_LOCKED	"770"
__BioAPIERR_FW_DATABASE_DOES_NOT_EXIST	"771"
__BioAPIERR_FW_DATABASE_ALREADY_EXISTS	"772"
__BioAPIERR_FW_INVALID_DATABASE_NAME	"773"
__BioAPIERR_FW_RECORD_NOT_FOUND	"774"
__BioAPIERR_FW_MARKER_HANDLE_IS_INVALID	"775"
__BioAPIERR_FW_DATABASE_IS_OPEN	"776"
__BioAPIERR_FW_INVALID_ACCESS_REQUEST	"777"
__BioAPIERR_FW_END_OF_DATABASE	"778"
__BioAPIERR_FW_UNABLE_TO_CREATE_DATABASE	"779"
__BioAPIERR_FW_UNABLE_TO_CLOSE_DATABASE"	"780"
__BioAPIERR_FW_UNABLE_TO_DELETE_DATABASE	"781"
__BioAPIERR_FW_DATABASE_IS_CORRUPT	"782"
__BioAPIERR_FW_LOCATION_ERROR	"1024"
__BioAPIERR_FW_OUT_OF_FRAME	"1025"
__BioAPIERR_FW_INVALID_CROSSWISE_POSITION	"1026"
__BioAPIERR_FW_INVALID_LENGTHWISE_POSITION	"1027"
__BioAPIERR_FW_INVALID_DISTANCE	"1028"
__BioAPIERR_FW_LOCATION_TOO_RIGHT	"1029"
__BioAPIERR_FW_LOCATION_TOO_LEFT	"1030"
__BioAPIERR_FW_LOCATION_TOO_HIGH	"1031"
__BioAPIERR_FW_LOCATION_TOO_LOW	"1032"
__BioAPIERR_FW_LOCATION_TOO_FAR	"1033"
__BioAPIERR_FW_LOCATION_TOO_NEAR	"1034"
__BioAPIERR_FW_LOCATION_TOO_FORWARD	"1035"
__BioAPIERR_FW_LOCATION_TOO_BACKWARD	"1036"
__BioAPIERR_FW_QUALITY_ERROR	"1281"
__BioAPIERR_BSP_INTERNAL_ERROR	"16777473"
__BioAPIERR_BSP_MEMORY_ERROR	"16777474"
__BioAPIERR_BSP_INVALID_POINTER	"16777475"
__BioAPIERR_BSP_INVALID_INPUT_POINTER	"16777476"
__BioAPIERR_BSP_INVALID_OUTPUT_POINTER	"16777477"
__BioAPIERR_BSP_FUNCTION_NOT_SUPPORTED	"16777478"
__BioAPIERR_BSP_OS_ACCESS_DENIED	"16777479"
__BioAPIERR_BSP_FUNCTION_FAILED	"16777480"
__BioAPIERR_BSP_INVALID_UUID	"16777481"
__BioAPIERR_BSP_INCOMPATIBLE_VERSION	"16777482"
__BioAPIERR_BSP_INVALID_DATA	"16777483"
__BioAPIERR_BSP_UNABLE_TO_CAPTURE	"16777484"
__BioAPIERR_BSP_TOO_MANY_HANDLES	"16777485"
__BioAPIERR_BSP_TIMEOUT_EXPIRED	"16777486"
__BioAPIERR_BSP_INVALID_BIR	"16777487"
__BioAPIERR_BSP_BIR_SIGNATURE_FAILURE	"16777488"
__BioAPIERR_BSP_UNABLE_TO_STORE_PAYLOAD	"16777489"
__BioAPIERR_BSP_NO_INPUT_BIRS	"16777490"

__BioAPIERR_BSP_UNSUPPORTED_FORMAT	"16777491"
__BioAPIERR_BSP_UNABLE_TO_IMPORT	"16777492"
__BioAPIERR_BSP_INCONSISTENT_PURPOSE	"16777493"
__BioAPIERR_BSP_BIR_NOT_FULLY_PROCESSED	"16777494"
__BioAPIERR_BSP_PURPOSE_NOT_SUPPORTED	"16777495"
__BioAPIERR_BSP_USER_CANCELLED	"16777496"
__BioAPIERR_BSP_UNIT_IN_USE	"16777497"
__BioAPIERR_BSP_INVALID_BSP_HANDLE	"16777498"
__BioAPIERR_BSP_FRAMEWORK_NOT_INITIALIZED	"16777499"
__BioAPIERR_BSP_INVALID_BIR_HANDLE	"16777500"
__BioAPIERR_BSP_CALIBRATION_NOT_SUCCESSFUL	"16777501"
__BioAPIERR_BSP_PRESET_BIR_DOES_NOT_EXIST	"16777502"
__BioAPIERR_BSP_BIR_DECRYPTION_FAILURE	"16777503"
__BioAPIERR_BSP_COMPONENT_FILE_REF_NOT_FOUND	"16777729"
__BioAPIERR_BSP_BSP_LOAD_FAIL	"16777730"
__BioAPIERR_BSP_BSP_NOT_LOADED	"16777731"
__BioAPIERR_BSP_UNIT_NOT_INSERTED	"16777732"
__BioAPIERR_BSP_INVALID_UNIT_ID	"16777733"
__BioAPIERR_BSP_INVALID_CATEGORY	"16777734"
__BioAPIERR_BSP_INVALID_DB_HANDLE	"16777984"
__BioAPIERR_BSP_UNABLE_TO_OPEN_DATABASE	"16777985"
__BioAPIERR_BSP_DATABASE_IS_LOCKED	"16777986"
__BioAPIERR_BSP_DATABASE_DOES_NOT_EXIST	"16777987"
__BioAPIERR_BSP_DATABASE_ALREADY_EXISTS	"16777988"
__BioAPIERR_BSP_INVALID_DATABASE_NAME	"16777989"
__BioAPIERR_BSP_RECORD_NOT_FOUND	"16777990"
__BioAPIERR_BSP_MARKER_HANDLE_IS_INVALID	"16777991"
__BioAPIERR_BSP_DATABASE_IS_OPEN	"16777992"
__BioAPIERR_BSP_INVALID_ACCESS_REQUEST	"16777993"
__BioAPIERR_BSP_END_OF_DATABASE	"16777994"
__BioAPIERR_BSP_UNABLE_TO_CREATE_DATABASE	"16777995"
__BioAPIERR_BSP_UNABLE_TO_CLOSE_DATABASE	"16777996"
__BioAPIERR_BSP_UNABLE_TO_DELETE_DATABASE	"16777997"
__BioAPIERR_BSP_DATABASE_IS_CORRUPT	"16777998"
__BioAPIERR_BSP_LOCATION_ERROR	"16778240"
__BioAPIERR_BSP_OUT_OF_FRAME	"16778241"
__BioAPIERR_BSP_INVALID_CROSSWISE_POSITION	"16778242"
__BioAPIERR_BSP_INVALID_LENGTHWISE_POSITION	"16778243"
__BioAPIERR_BSP_INVALID_DISTANCE	"16778244"
__BioAPIERR_BSP_LOCATION_TOO_RIGHT	"16778245"
__BioAPIERR_BSP_LOCATION_TOO_LEFT	"16778246"
__BioAPIERR_BSP_LOCATION_TOO_HIGH	"16778247"
__BioAPIERR_BSP_LOCATION_TOO_LOW	"16778248"
__BioAPIERR_BSP_LOCATION_TOO_FAR	"16778249"
__BioAPIERR_BSP_LOCATION_TOO_NEAR	"16778250"
__BioAPIERR_BSP_LOCATION_TOO_FORWARD	"16778251"
__BioAPIERR_BSP_LOCATION_TOO_BACKWARD	"16778252"
__BioAPIERR_BSP_QUALITY_ERROR	"16778497"
__BioAPIERR_UNIT_INTERNAL_ERROR	"33554689"
__BioAPIERR_UNIT_MEMORY_ERROR	"33554690"
__BioAPIERR_UNIT_INVALID_POINTER	"33554691"
__BioAPIERR_UNIT_INVALID_INPUT_POINTER	"33554692"
__BioAPIERR_UNIT_INVALID_OUTPUT_POINTER	"33554693"

__BioAPIERR_UNIT_FUNCTION_NOT_SUPPORTED	"33554694"
__BioAPIERR_UNIT_OS_ACCESS_DENIED	"33554695"
__BioAPIERR_UNIT_FUNCTION_FAILED	"33554696"
__BioAPIERR_UNIT_INVALID_UUID	"33554697"
__BioAPIERR_UNIT_INCOMPATIBLE_VERSION	"33554698"
__BioAPIERR_UNIT_INVALID_DATA	"33554699"
__BioAPIERR_UNIT_UNABLE_TO_CAPTURE	"33554700"
__BioAPIERR_UNIT_TOO_MANY_HANDLES	"33554701"
__BioAPIERR_UNIT_TIMEOUT_EXPIRED	"33554702"
__BioAPIERR_UNIT_INVALID_BIR	"33554703"
__BioAPIERR_UNIT_BIR_SIGNATURE_FAILURE	"33554704"
__BioAPIERR_UNIT_UNABLE_TO_STORE_PAYLOAD	"33554705"
__BioAPIERR_UNIT_NO_INPUT_BIRS	"33554706"
__BioAPIERR_UNIT_UNSUPPORTED_FORMAT	"33554707"
__BioAPIERR_UNIT_UNABLE_TO_IMPORT	"33554708"
__BioAPIERR_UNIT_INCONSISTENT_PURPOSE	"33554709"
__BioAPIERR_UNIT_BIR_NOT_FULLY_PROCESSED	"33554710"
__BioAPIERR_UNIT_PURPOSE_NOT_SUPPORTED	"33554711"
__BioAPIERR_UNIT_USER_CANCELLED	"33554712"
__BioAPIERR_UNIT_UNIT_IN_USE	"33554713"
__BioAPIERR_UNIT_INVALID_BSP_HANDLE	"33554714"
__BioAPIERR_UNIT_FRAMEWORK_NOT_INITIALIZED	"33554715"
__BioAPIERR_UNIT_INVALID_BIR_HANDLE	"33554716"
__BioAPIERR_UNIT_CALIBRATION_NOT_SUCCESSFUL	"33554717"
__BioAPIERR_UNIT_PRESET_BIR_DOES_NOT_EXIST	"33554718"
__BioAPIERR_UNIT_BIR_DECRYPTION_FAILURE	"33554719"
__BioAPIERR_UNIT_COMPONENT_FILE_REF_NOT_FOUND	"33554945"
__BioAPIERR_UNIT_BSP_LOAD_FAIL	"33554946"
__BioAPIERR_UNIT_BSP_NOT_LOADED	"33554947"
__BioAPIERR_UNIT_UNIT_NOT_INSERTED	"33554948"
__BioAPIERR_UNIT_INVALID_UNIT_ID	"33554949"
__BioAPIERR_UNIT_INVALID_CATEGORY	"33554950"
__BioAPIERR_UNIT_INVALID_DB_HANDLE	"33555200"
__BioAPIERR_UNIT_UNABLE_TO_OPEN_DATABASE	"33555201"
__BioAPIERR_UNIT_DATABASE_IS_LOCKED	"33555202"
__BioAPIERR_UNIT_DATABASE_DOES_NOT_EXIST	"33555203"
__BioAPIERR_UNIT_DATABASE_ALREADY_EXISTS	"33555204"
__BioAPIERR_UNIT_INVALID_DATABASE_NAME	"33555205"
__BioAPIERR_UNIT_RECORD_NOT_FOUND	"33555206"
__BioAPIERR_UNIT_MARKER_HANDLE_IS_INVALID	"33555207"
__BioAPIERR_UNIT_DATABASE_IS_OPEN	"33555208"
__BioAPIERR_UNIT_INVALID_ACCESS_REQUEST	"33555209"
__BioAPIERR_UNIT_END_OF_DATABASE	"33555210"
__BioAPIERR_UNIT_UNABLE_TO_CREATE_DATABASE	"33555211"
__BioAPIERR_UNIT_UNABLE_TO_CLOSE_DATABASE	"33555212"
__BioAPIERR_UNIT_UNABLE_TO_DELETE_DATABASE	"33555213"
__BioAPIERR_UNIT_DATABASE_IS_CORRUPT	"33555214"
__BioAPIERR_UNIT_LOCATION_ERROR	"33555456"
__BioAPIERR_UNIT_OUT_OF_FRAME	"33555457"
__BioAPIERR_UNIT_INVALID_CROSSWISE_POSITION	"33555458"
__BioAPIERR_UNIT_INVALID_LENGTHWISE_POSITION	"33555459"
__BioAPIERR_UNIT_INVALID_DISTANCE	"33555460"
__BioAPIERR_UNIT_LOCATION_TOO_RIGHT	"33555461"
__BioAPIERR_UNIT_LOCATION_TOO_LEFT	"33555462"

__BioAPIERR_UNIT_LOCATION_TOO_HIGH	**"33555463"**
__BioAPIERR_UNIT_LOCATION_TOO_LOW	**"33555464"**
__BioAPIERR_UNIT_LOCATION_TOO_FAR	**"33555465"**
__BioAPIERR_UNIT_LOCATION_TOO_NEAR	**"33555466"**
__BioAPIERR_UNIT_LOCATION_TOO_FORWARD	**"33555467"**
__BioAPIERR_UNIT_LOCATION_TOO_BACKWARD	**"33555468"**
__BioAPIERR_UNIT_QUALITY_ERROR	**"33555713"**

10.2 Переменные, имеющие значения, которые могут изменяться

Примечание — Некоторые из приведенных в данном подразделе встроенных переменных связаны друг с другом. Значение каждой переменной, входящей в совокупность, изменяется в соответствии с 7.3.3.

10.2.1 __exit

Данной переменной должно быть присвоено значение **"false"** в момент начала выполнения основного процесса. Когда компонент тестирования получает запрос на приостановку процесса, данной переменной должно быть присвоено значение **"true"**; это значение должно оставаться неизменным до завершения основного процесса.

Примечание — Механизм, посредством которого компонент тестирования получает такого рода запрос, определяется конкретной реализацией.

10.2.2 __native_functions_called

Значением этой переменной должно быть стандартно представленное целое число, которое является суммой вызовов собственной функции, связанных с вызовом функции стандартного интерфейса BioAPI любым процессом, которые были выполнены в течение основного процесса.

10.2.3 __native_functions_returned

Значением этой переменной должно быть стандартно представленное целое число, которое является суммой вызовов собственной функции, связанных с вызовом функции стандартного интерфейса BioAPI любым процессом, которые завершились (с любым значением) в течение основного процесса.

Примечание — В любой момент значение этой переменной меньше или равно значению переменной **__native_functions_called** (с учетом 7.3.3).

10.2.4 __native_functions_cancelled

Значением этой переменной должно быть стандартно представленное целое число, которое является суммой вызовов собственной функции, связанных с вызовом функции стандартного интерфейса BioAPI любым процессом, которые завершились с кодом ошибки **BioAPIERR_USER_CANCELLED** в течение основного процесса.

Примечание — В любой момент значение этой переменной меньше или равно значению переменной **__native_functions_returned** (с учетом 7.3.3).

11 Протокол испытания

11.1 Для вывода данных о результатах проведенного испытания абстрактная испытательная машина должна использовать файл XML. Корневым элементом файла XML должен быть элемент **<conformance_test_log>** (см. 11.2). Файл XML должен быть корректным. Корректность файла XML определяют путем его сравнения со схемой, приведенной в приложении С.

11.2 Элемент **<conformance_test_log>** должен иметь следующие атрибуты:

a) **date_time** (обязательный атрибут) — корректное значение даты и времени в формате, соответствующем ИСО 8601;

b) **standard** (обязательный атрибут) — значением этого атрибута должно быть **"ISO/IEC 19784-1 (BioAPI)"**;

В состав элемента **<conformance_test_log>** в указанном порядке должны входить:

a) один элемент **<TestingLaboratory>** (см. 11.3), содержащий контактные данные организации, проводящей испытания;

b) один элемент **<Vendor>** (см. 11.4), содержащий контактные данные поставщика тестируемой реализации;

c) один элемент **<Biometric_Product>** (см. 11.6), содержащий уникальный идентификатор испытуемой биометрической продукции;

d) один элемент **<CTS_ID>**, содержащий строку символов, которая идентифицирует КТСБПИ;

e) один элемент **<test_assertion>**, содержащий имя исполняемого утверждения, имя пакета, в котором содержится утверждение, и значения всех входных параметров утверждения;

f) ни одного или более появлений любых из следующих элементов в любом порядке:

1) **<function>** (см. 11.10) — этот элемент представляет собой вызов функции стандартного интерфейса BioAPI (включая обратные вызовы), которые происходят в процессе испытания;

2) **<inline_response>** (см. 11.12) — этот элемент представляет собой выходные данные испытания, которые формируются элементом **<assert_condition>** во время проведения испытания.

11.3 Элемент **<TestingLaboratory>** не должен иметь атрибутов. В состав элемента **<TestingLaboratory>** в указанном порядке должны входить:

a) один элемент **<Name>** — строка символов;

b) один элемент **<Address>** (см. 11.5) — почтовый адрес;

c) один элемент **<Phone>** — строка символов;

d) дополнительный элемент **<E-mail>** — адрес электронной почты;

e) дополнительный элемент **<Url>** — адрес страницы в интернете.

11.4 Элемент **<Vendor>** должен иметь те же атрибуты и содержимое, что и элемент **<TestingLaboratory>**.

11.5 Элемент **<Address>** не должен иметь атрибутов. В состав элемента **<Address>** в указанном порядке должны входить:

a) один элемент **<Street>** (улица) — строка символов;

b) один элемент **<City>** (город) — строка символов;

c) дополнительный элемент **<StateOrProvince>** (штат или область) — строка символов;

d) один элемент **<ZipOrPostalCode>** (почтовый индекс) — строка символов;

e) один элемент **<Country>** (страна) — строка символов.

11.6 Элемент **<BiometricProduct>** должен иметь следующие атрибуты:

a) **Name** (обязательный атрибут) — строка символов;

b) **SerialNo** (обязательный атрибут) — строка символов.

В состав элемента **SerialNo** должен входить один элемент **<Description>** — строка символов.

11.7 Элемент **<test_assertion>** не должен иметь атрибутов. В состав элемента **<test_assertion>** в указанном порядке должны входить:

a) один элемент **<package_name>** — строка символов, представляющая собой универсальный уникальный идентификатор (см. 6.6);

b) один элемент **<assertion_name>** — строка символов;

c) один элемент **<Description>** — строка символов, содержащая описание утверждения и ни одного или более элементов **<input>** (см. 11.8), каждый из которых содержит имя и значение входного параметра утверждения.

11.8 Элемент **<input>** должен иметь следующие атрибуты:

a) **name** (обязательный атрибут) — строка символов, подходящая для создания «NCName» (неквалифицированного имени) в пространствах имен W3C XML;

b) **value** (обязательный атрибут) — строка символов;

Элемент **<input>** не должен иметь текстового содержимого.

11.9 Элемент **<output>** должен иметь те же атрибуты и содержимое, что и элемент **<input>**.

11.10 Элемент **<function>** должен иметь следующий атрибут:

- **dir** (обязательный атрибут) — одна из строк символов **"incoming"**, **"outcoming"**, определяющая направление вызова функции по отношению к испытуемому компоненту.

В состав элемента **<function>** в указанном порядке должны входить:

a) один элемент **<name>** — строка символов;

b) ни одного или более элементов **<input>** (см. 11.8), каждый из которых содержит имя и значение входного параметра вызова функции;

c) ни одного или более элементов **<output>** (см. 11.9), каждый из которых содержит имя и значение выходного параметра вызова функции;

d) один элемент **<return>** (см. 11.11), содержащий возвращаемое значение вызова функции.

11.11 Элемент **<return>** должен иметь следующий атрибут:

- **value** (обязательный атрибут) — строка символов.

Элемент **<return>** не должен иметь текстового содержимого.

11.12 Элемент **<inline_response>** не должен иметь атрибутов. В состав элемента **<inline_response>** в указанном порядке должны входить:

a) один элемент **<asserted_condition>** — строка символов;

b) один элемент **<conformance>** — одна из четырех строк символов **"error"** (ошибка), **"pass"** (пройден), **"fail"** (непройден), **"undecided"** (нерешено).

12 Отчет о результатах испытаний

12.1 Результаты испытания одного или более утверждений в отношении тестируемой реализации должны быть зафиксированы в файле XML. Корневым элементом файла XML должен быть элемент **<conformance_test_report>** (см. 12.2).

12.2 Элемент **<conformance_test_report>** не должен иметь атрибутов. В состав элемента **<conformance_test_report>** в указанном порядке должны входить:

a) один элемент **<Title>** — название отчета (строка символов);

b) один элемент **<TestingLaboratory>** (см. 11.3) — контактные данные организации, проводящей испытания;

c) один элемент **<ReportID>** — уникальный идентификатор отчета о результатах испытаний (строка символов);

d) один элемент **<Vendor>** (см. 11.4) — контактные данные поставщика тестируемой реализации;

e) один элемент **<Biometric_Product>** (см. 11.6), содержащий уникальный идентификатор испытуемой биометрической продукции;

f) один элемент **<IUT_Description>** — описание тестируемой реализации (строка символов);

g) один элемент **<Date_of_test>** — дата начала проведения испытания (строка символов в формате, который соответствует стандарту ИСО 8601);

h) один элемент **<CTS_ID>**, содержащий строку символов, которая идентифицирует КТСБПИ;

i) один элемент **<Environment>** — платформа, аппаратное обеспечение, операционная система, с помощью которых проводят испытания, и КТСБПИ (строка символов);

j) один элемент **<Test_Configuration>** (см. 12.3) — данные о конфигурации теста;

k) один элемент **<Date_of_issue>** — дата составления отчета о результатах проведенных испытаний (строка символов в формате, соответствующем ИСО 8601);

l) один элемент **<Report_authorization>** (см. 12.5) — данные о составителях отчета о проведенных испытаниях, которые являются ответственными за его содержание.

12.3 Элемент **<Test_configuration>** не должен иметь атрибутов. В состав элемента **<Test_configuration>** должны входить один или более элементов **<Assertion>** (см. 12.4).

12.4 Элемент **<Assertion>** должен иметь следующие атрибуты:

a) **name** (обязательный атрибут) — строка символов, подходящая для создания «NCName» (неквалифицированного имени) в пространствах имен W3C XML;

b) **package** (обязательный атрибут) — строка символов, которая представляет собой универсальный уникальный идентификатор (см. 7.6);

В состав элемента **<Assertion>** в указанном порядке должны входить:

a) дополнительный элемент **<description>**;

b) ни одного или более элементов **<input>** (см. 11.8), каждый из которых содержит имя и значение входного параметра утверждения;

c) один элемент **<conformance>**, содержащий обобщенный ответ о соответствии утверждению, полученный на основании всех элементов **<conformance>**, которые присутствуют в протоколе испытания утверждения. Обобщенный ответ о соответствии утверждению должен представлять собой одну из следующих строк символов:

1 **"error"**, если есть один или более элементов **<conformance>**, содержащих строку **"error"** в отчете об испытаниях утверждения или

2 **"fail"**, если присутствует один или более элементов **<conformance>**, содержащих строку **"fail"**, но не присутствует ни одного, содержащего строку **"error"**, или

3 **"undecided"**, если присутствует один или более элементов **<conformance>**, содержащих строку **"undecided"**, но не присутствует ни одного, содержащего строки **"error"** или **"fail"**, или

4 **"pass"**, если присутствует один или более элементов **<conformance>**, содержащих строку **"pass"**, но не присутствует ни одного, содержащего строки **"error"**, **"fail"** или **"undecided"**, или

5 **"error"**, если в отчете об испытаниях нет элементов **<conformance>**;

d) дополнительный элемент **<Condition_evaluated>**, содержащий условие в составе элемента **<assert_condition>**.

12.5 Элемент **<Report_authorization>** не должен иметь атрибутов. В состав элемента **<Report_authorization>** должны входить один или более элементов **<Tester>** (см. 12.6).

12.6 Элемент **<Tester>** должен иметь следующие атрибуты:

a) **Name** (обязательный атрибут) — фамилия, имя, отчество человека (строка символов);

b) **Title** (обязательный атрибут) — должность человека (строка символов).

13 Комплект тестов на соответствие BioAPI

13.1 Общие положения

13.1.1 Комплект тестов на соответствие биометрическому программному интерфейсу (КТСБПИ) является сочетанием выполняемых контрольных примеров, соответствующих тестовых данных и процедуры испытаний. Возможно проведение испытаний на соответствие только тем требованиям, которые указаны в спецификации BioAPI. Комплект тестов может включать в себя тесты для проверки требований, указанных в спецификации BioAPI как необязательные, однако нельзя устанавливать обязательность соответствия этим требованиям.

13.1.2 До начала выполнения КТСБПИ может потребоваться подготовка набора записей данных. Набор данных должен включать в себя некоторое количество определенных заранее ЗБИ, которые могут содержать как корректные, так и некорректные данные. Набор данных может включать в себя записи, заведомо содержащие ошибку, которые необходимы для проверки обработки тестовой реализацией ошибочных данных в соответствии с требованиями конкретных тестовых примеров.

13.1.3 В процедуре проведения испытаний приведены требования к процессу выполнения КТСБПИ и инструкции для оператора, проводящего испытания. Кроме того, процедура проведения испытаний должна быть описана достаточно подробно для того, чтобы обеспечить возможность повторного испытания данной реализации с получением идентичных результатов. Для этого необходимо, чтобы КТСБПИ был по возможности максимально автоматизирован.

13.2 Структура комплекта тестов на соответствие биометрическому программному интерфейсу

13.2.1 В соответствии с предыдущими разделами настоящего стандарта, КТСБПИ имеет иерархическую структуру, основным элементом которой является контрольный пример. Каждый контрольный пример обычно имеет одну цель, например определение того, что ТР имеет определенную необходимую способность или демонстрирует определенное необходимое поведение. В состав каждого контрольного примера входят:

a) описание цели испытания (то есть, что подвергается испытанию — условия, требования или способности, которые должны быть определены с помощью данного испытания);

b) критерий прохождения/не прохождения контрольного примера;

c) ссылка на раздел спецификации BioAPI, на основании которого подготовлен данный контрольный пример.

13.2.2 В соответствии с изложенным ниже, каждый контрольный пример предусматривает объективные, воспроизводимые, точно выраженные и правильные результаты тестирования. Каждый контрольный пример можно проследить до утверждения или утверждений в спецификации BioAPI и, кроме того, каждый контрольный пример составлен так, чтобы можно было точно определить поведение соответствующей ТР. Каждый контрольный пример осуществляет проверку условия с получением результата **"pass"** или **"fail"**. Контрольные примеры могут быть объединены в группы или сценарии в рамках КТСБПИ; каждая группа контрольных примеров может состоять из одного или более контрольных примеров, сгруппированных в логической последовательности. Каждая группа контрольных примеров может иметь конкретную цель испытания. Контрольный пример может быть разбит на самостоятельные единицы, называемые тестовыми шагами. Тестовый шаг имеет имя и может использоваться в дальнейшем для создания новых контрольных примеров.

13.2.3 Результатом испытания является последовательность событий, которые происходили во время выполнения контрольного примера. Результат испытания включает в себя все входные и выходные данные ТР в момент контроля и наблюдения.

13.2.4 Ожидаемым результатом испытания является результат, определенный контрольным примером, то есть событиями, происходящими во время выполнения контрольного примера. Результатом ожида-

емого результата испытания всегда является назначение конкретному контрольному примеру заключения испытания.

13.2.5 Возможны следующие заключения испытания:

a) **Pass** (прошел) — означает, что результат испытания свидетельствует о соответствии требованию(ям), на котором(ых) сосредоточена цель испытания данного контрольного примера;

b) **Fail** (не прошел) — означает, что результат испытания свидетельствует о несоответствии хотя бы одному из требований, на которых сосредоточена цель испытания, или содержит хотя бы одно некорректное событие тестирования;

c) **Inconclusive or Undecided** (неубедительный или нерешенный) — означает, что результат испытания является таким, что ни одно из заключений **"pass"** или **"fail"** вынесено быть не может.

13.2.6 Непредвиденным результатом испытания является результат, который не был определен контрольным примером, то есть события, которые происходили во время выполнения контрольного примера, не совпадали ни с одной последовательностью событий, указанных в контрольном примере. Непредвиденный результат испытания может быть вызван ошибкой в записи контрольного примера или неправильным завершением контрольного примера.

13.2.7 Ошибку контрольного примера записывают в том случае, если обнаруживаются ошибки или в самом контрольном примере, или в его реализации (такие ошибки называются ошибками времени выполнения контрольного примера).

13.2.8 Неправильное завершение контрольного примера записывают в том случае, если выполнение контрольного примера завершается абстрактной испытательной машиной преждевременно не из-за ошибки в контрольном примере, а по иной причине.

13.2.9 КТСБПИ формируется на основе цели испытания и определяет следующее:

a) все последовательности тестовых шагов и ожидаемых результатов испытания, которые необходимы для достижения цели испытания;

b) заключение, которое должно быть присвоено каждой возможной последовательности результатов испытания, формирующей полный путь от входа в контрольный пример до выхода из него.

13.2.10 Для описания всех утверждений используют стандартизированное описание тестов. Описание предполагает, что внутренняя структура или исходный код ТР не доступны человеку, проводящему испытание, то есть испытание на соответствие, проводимое КТСБПИ, является испытанием по принципу «черного ящика».

Приложение А
(обязательное)

XML-схема языка утверждения

Все пакеты должны быть корректными в соответствии с XML-схемой.

```
<?xml version='1.0' encoding="utf-8"?>
<xs:schema xmlns:xs="gost.e-ecolog.ru                    ">
   <xs:element name="package" type="package"/>
   <xs:complexType name="package">
      <xs:sequence>
         <xs:element name="author" type="xs:string"/>
         <xs:element name="description" type="xs:string"/>
         <xs:element name="assertion" type="assertion"
               minOccurs="0" maxOccurs="unbounded"/>
         <xs:element name="activity" type="activity"
               minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="name" type="uuid" use="required"/>
   </xs:complexType>
   <xs:complexType name="assertion">
      <xs:sequence>
         <xs:element name="description" type="xs:string"/>
         <xs:element name="input" type="assertionInput"
               minOccurs="0" maxOccurs="unbounded"/>
         <xs:element name="invoke" type="assertionInvoke"/>
         <xs:element name="bind" type="assertionBind"
               minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="name" type="xs:NCName" use="required"/>
      <xs:attribute name="model" type="model" use="required"/>
   </xs:complexType>
   <xs:complexType name="assertionInput">
      <xs:attribute name="name" type="xs:NCName" use="required"/>
   </xs:complexType>
   <xs:complexType name="assertionInvoke">
      <xs:sequence>
         <xs:element name="input" type="invokeInput"
               minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="activity" type="xs:NCName" use="required"/>
      <xs:attribute name="package" type="uuid"/>
   </xs:complexType>
   <xs:complexType name="invokeInput">
      <xs:attribute name="name" type="xs:NCName" use="required"/>
      <xs:attribute name="value" type="xs:string"/>
      <xs:attribute name="var" type="xs:NCName"/>
   </xs:complexType>
   <xs:complexType name="invokeOutput">
      <xs:attribute name="name" type="xs:NCName" use="required"/>
      <xs:attribute name="setvar" type="xs:NCName" use="required"/>
   </xs:complexType>
   <xs:complexType name="invokeReturn">
      <xs:attribute name="setvar" type="xs:NCName" use="required"/>
   </xs:complexType>
```

```
<xs:complexType name="assertionBind">
    <xs:attribute name="function" type="xs:NCName" use="required"/>
    <xs:attribute name="activity" type="xs:NCName" use="required"/>
    <xs:attribute name="package" type="uuid"/>
</xs:complexType>
<xs:complexType name="activity">
    <xs:sequence>
        <xs:element name="input" type="activityInputOutput"
              minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="output" type="activityInputOutput"
              minOccurs="0" maxOccurs="unbounded"/>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="set" type="set"/>
            <xs:element name="add" type="arithmetic"/>
            <xs:element name="subtract" type="arithmetic"/>
            <xs:element name="wait_until" type="wait_until"/>
            <xs:element name="assert_condition"
                  type="assert_condition"/>
            <xs:element name="invoke" type="activityInvoke"/>
        </xs:choice>
    </xs:sequence>
    <xs:attribute name="name" type="xs:NCName" use="required"/>
    <xs:attribute name="atomic" type="xs:boolean" default="false"/>
</xs:complexType>
<xs:complexType name="activityInputOutput">
    <xs:attribute name="name" type="xs:NCName" use="required"/>
</xs:complexType>
<xs:complexType name="set">
    <xs:sequence>
        <xs:element name="only_if" type="only_if" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:NCName" use="required"/>
    <xs:attribute name="value" type="xs:string"/>
    <xs:attribute name="var" type="xs:NCName"/>
</xs:complexType>
<xs:complexType name="arithmetic">
    <xs:sequence>
        <xs:element name="only_if" type="only_if" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:NCName" use="required"/>
    <xs:attribute name="value" type="xs:integer"/>
    <xs:attribute name="var" type="xs:NCName"/>
</xs:complexType>
<xs:complexType name="activityInvoke">
    <xs:sequence>
        <xs:element name="only_if" type="only_if" minOccurs="0"/>
        <xs:element name="input" type="invokeInput"
              minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="output" type="invokeOutput"
              minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="return" type="invokeReturn" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="activity" type="xs:NCName"/>
    <xs:attribute name="package" type="uuid"/>
    <xs:attribute name="break_on_break" type="xs:boolean"
          default="false"/>
    <xs:attribute name="function" type="xs:NCName"/>
    <xs:attribute name="timeout_value" type="xs:integer"/>
    <xs:attribute name="timeout_var" type="xs:NCName"/>
    <xs:attribute name="setvar" type="xs:NCName"/>
</xs:complexType>
```

```
<xs:complexType name="only_if">
    <xs:group ref="expression"/>
    <xs:attribute name="var" type="xs:NCName"/>
</xs:complexType>
<xs:complexType name="wait_until">
    <xs:group ref="expression"/>
    <xs:attribute name="timeout_value" type="xs:integer"/>
    <xs:attribute name="timeout_var" type="xs:NCName"/>
    <xs:attribute name="setvar" type="xs:NCName"/>
    <xs:attribute name="var" type="xs:NCName"/>
</xs:complexType>
<xs:complexType name="assert_condition">
    <xs:group ref="expression"/>
    <xs:attribute name="response_if_true" type="response"
        default="pass"/>
    <xs:attribute name="response_if_false" type="response"
        default="fail"/>
    <xs:attribute name="break_if_false" type="xs:boolean"
        default="false"/>
    <xs:attribute name="var" type="xs:NCName"/>
</xs:complexType>
<xs:complexType name="logicalBinaryOperator">
    <xs:group ref="expression"/>
    <xs:attribute name="var1" type="xs:NCName"/>
    <xs:attribute name="var2" type="xs:NCName"/>
</xs:complexType>
<xs:complexType name="logicalUnaryOperator">
    <xs:group ref="expression"/>
    <xs:attribute name="var" type="xs:NCName"/>
</xs:complexType>
<xs:complexType name="arithmeticBinaryOperator">
    <xs:group ref="expression"/>
    <xs:attribute name="var1" type="xs:NCName"/>
    <xs:attribute name="var2" type="xs:NCName"/>
    <xs:attribute name="value1" type="xs:integer"/>
    <xs:attribute name="value2" type="xs:integer"/>
</xs:complexType>
<xs:complexType name="stringBinaryOperator">
    <xs:group ref="expression"/>
    <xs:attribute name="var1" type="xs:NCName"/>
    <xs:attribute name="var2" type="xs:NCName"/>
    <xs:attribute name="value1" type="xs:string"/>
    <xs:attribute name="value2" type="xs:string"/>
</xs:complexType>
<xs:complexType name="variableOperator">
    <xs:attribute name="var" type="xs:NCName" use="required"/>
</xs:complexType>
<xs:simpleType name="uuid">
    <xs:restriction base="xs:token">
        <xs:pattern value="[a-fA-F0-9] {8}\-([a-fA-F0-9] {4}\-)
            {3} [a-fA-F0-9] {12}"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="model">
    <xs:restriction base="xs:token">
        <xs:enumeration value="applicationTesting"/>
        <xs:enumeration value="frameworkTesting"/>
        <xs:enumeration value="BSPTesting"/>
    </xs:restriction>
</xs:simpleType>
```

```
    <xs:simpleType name="response">
        <xs:restriction base="xs:token">
            <xs:enumeration value="pass"/>
            <xs:enumeration value="fail"/>
            <xs:enumeration value="undecided"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:group name="expression">
        <xs:sequence>
            <xs:element name="description" type="xs:string"
                minOccurs="0"/>
            <xs:choice minOccurs="0" maxOccurs="unbounded">
                <xs:element name="and" type="logicalBinaryOperator"/>
                <xs:element name="or" type="logicalBinaryOperator"/>
                <xs:element name="xor" type="logicalBinaryOperator"/>
                <xs:element name="not" type="logicalUnaryOperator"/>
                <xs:element name="equal_to" type="arithmeticBinaryOperator"/>
                <xs:element name="not_equal_to" type="arithmeticBinaryOperator"/>
                <xs:element name="greater_than"
                    type="arithmeticBinaryOperator"/>
                <xs:element name="greater_than_or_equal_to"
                    type="arithmeticBinaryOperator"/>
                <xs:element name="less_than"
                    type="arithmeticBinaryOperator"/>
                <xs:element name="less_than_or_equal_to"
                    type="arithmeticBinaryOperator"/>
                <xs:element name="same_as" type="stringBinaryOperator"/>
                <xs:element name="not_same_as" type="stringBinaryOperator"/>
                <xs:element name="existing"  type="variableOperator"/>
                <xs:element name="not_existing" type="variableOperator"/>
            </xs:choice>
        </xs:sequence>
    </xs:group>
</xs:schema>
```

Приложение В
(обязательное)

Схема языка утверждения ASN.1

Все пакеты должны быть корректными в соответствии со схемой ASN.1:

```
BioAPI-CT-Assertion-Language
{iso standard 19784-1 modules(0) ct-assertion-language(1)}
DEFINITIONS XER INSTRUCTIONS AUTOMATIC TAGS ::= BEGIN

PackageElement ::= [LOWERCASED] Package

Package ::= SEQUENCE {
    name [ATTRIBUTE] Uuid,
    author UTF8String,
    description UTF8String,
    assertions [UNTAGGED] SEQUENCE OF assertion Assertion,
    activities [UNTAGGED] SEQUENCE OF activity Activity
}

Assertion ::= SEQUENCE {
    name [ATTRIBUTE] NCName,
    model [ATTRIBUTE] Model,
    description UTF8String,
    inputs [UNTAGGED] SEQUENCE OF input AssertionInput,
    invoke AssertionInvoke,
    binds [UNTAGGED] SEQUENCE OF bind AssertionBind
}

AssertionInput ::= SEQUENCE {
    name [ATTRIBUTE] NCName
}

AssertionInvoke ::= SEQUENCE {
    activity [ATTRIBUTE] NCName,
    package [ATTRIBUTE] Uuid OPTIONAL,
    inputs [UNTAGGED] SEQUENCE OF input InvokeInput
}
InvokeInput ::= SEQUENCE {
    name [ATTRIBUTE] NCName,
    value [ATTRIBUTE] UTF8String OPTIONAL,
    var [ATTRIBUTE] NCName OPTIONAL
}

InvokeOutput ::= SEQUENCE {
    name [ATTRIBUTE] NCName,
    setvar [ATTRIBUTE] NCName
}

InvokeReturn ::= SEQUENCE {
    setvar [ATTRIBUTE] NCName
}
AssertionBind ::= SEQUENCE {
    function [ATTRIBUTE] NCName,
    activity [ATTRIBUTE] NCName,
    package [ATTRIBUTE] Uuid OPTIONAL,
}

Activity ::= SEQUENCE {
    name [ATTRIBUTE] NCName,
    atomic [ATTRIBUTE] BOOLEAN DEFAULT FALSE,
    inputs [UNTAGGED] SEQUENCE OF input ActivityInputOutput,
    outputs [UNTAGGED] SEQUENCE OF output ActivityInputOutput,
```

```
    choices [UNTAGGED] SEQUENCE OF [UNTAGGED] CHOICE {
        set Set,
        add Arithmetic,
        subtract Arithmetic,
        wait-until [NAME AS "wait_until"] WaitUntil,
        assert-condition [NAME AS "assert_condition"] AssertCondition,
        invoke ActivityInvoke
    }
}

ActivityInputOutput ::= SEQUENCE {
    name [ATTRIBUTE] NCName
}

Set ::= SEQUENCE {
    name [ATTRIBUTE] NCName,
    value [ATTRIBUTE] UTF8String OPTIONAL,
    var [ATTRIBUTE] NCName OPTIONAL,
    only-if [NAME AS "only_if"] OnlyIf OPTIONAL
}

Arithmetic ::= SEQUENCE {
    name [ATTRIBUTE] NCName,
    value [ATTRIBUTE] INTEGER OPTIONAL,
    var [ATTRIBUTE] NCName OPTIONAL,
    only-if [NAME AS "only_if"] OnlyIf OPTIONAL
}

ActivityInvoke ::= SEQUENCE {
    activity [ATTRIBUTE] NCName,
    package [ATTRIBUTE] Uuid,
    break-on-break [ATTRIBUTE] [NAME AS "break_on_break"]
        BOOLEAN DEFAULT FALSE,
    function [ATTRIBUTE] NCName,
    timeout-value [ATTRIBUTE] [NAME AS "timeout_value"]
        UTF8String OPTIONAL,
    timeout-var [ATTRIBUTE] [NAME AS "timeout_var"]
        NCName OPTIONAL,
    setvar [ATTRIBUTE] NCName OPTIONAL,
    only-if [NAME AS "only_if"] OnlyIf OPTIONAL,
    inputs [UNTAGGED] SEQUENCE OF input InvokeInput,
    outputs [UNTAGGED] SEQUENCE OF input InvokeInput,
    return InvokeReturn OPTIONAL
}

OnlyIf ::= SEQUENCE {
    var [ATTRIBUTE] NCName OPTIONAL,
    expression [UNTAGGED] Expression
}

WaitUntil ::= SEQUENCE {
    timeout-value [ATTRIBUTE] [NAME AS "timeout_value"]
        UTF8String OPTIONAL,
    timeout-var [ATTRIBUTE] [NAME AS "timeout_var"]
        NCName OPTIONAL,
    setvar [ATTRIBUTE] NCName OPTIONAL,
    var [ATTRIBUTE] NCName OPTIONAL,
    expression [UNTAGGED] Expression
}

AssertCondition ::= SEQUENCE {
    response-if-true [ATTRIBUTE] [NAME AS "response_if_true"]
        Response DEFAULT pass,
    response-if-false [ATTRIBUTE] [NAME AS "response_if_false"]
        Response DEFAULT fail,
    break-if-false [ATTRIBUTE] [NAME AS "break_if_false"]
        BOOLEAN DEFAULT FALSE,
```

```
    var [ATTRIBUTE] NCName OPTIONAL,
    expression [UNTAGGED] Expression
}

LogicalBinaryOperator ::= SEQUENCE {
    var1 [ATTRIBUTE] NCName OPTIONAL,
    var2 [ATTRIBUTE] NCName OPTIONAL,
    expression [UNTAGGED] Expression
}

LogicalUnaryOperator ::= SEQUENCE {
    var [ATTRIBUTE] NCName OPTIONAL,
    expression [UNTAGGED] Expression
}

ArithmeticBinaryOperator ::= SEQUENCE {
    var1 [ATTRIBUTE] NCName OPTIONAL,
    var2 [ATTRIBUTE] NCName OPTIONAL,
    value1 [ATTRIBUTE] INTEGER OPTIONAL,
    value2 [ATTRIBUTE] INTEGER OPTIONAL,
    expression [UNTAGGED] Expression
}

StringBinaryOperator ::= SEQUENCE {
    var1 [ATTRIBUTE] NCName OPTIONAL,
    var2 [ATTRIBUTE] NCName OPTIONAL,
    value1 [ATTRIBUTE] UTF8String OPTIONAL,
    value2 [ATTRIBUTE] UTF8String OPTIONAL,
    expression [UNTAGGED] Expression
}

VariableOperator ::= SEQUENCE {
    var [ATTRIBUTE] NCName
}

Uuid ::= VisibleString (SIZE(36)) (FROM "0123456789abcdefABCDEF-")
Model ::= ENUMERATED {
    applicationTesting, frameworkTesting, bspTesting
}

Response ::= ENUMERATED {
    pass, fail, undecided
}

NCName ::= UTF8String
Expression ::= SEQUENCE {
    description UTF8String OPTIONAL,
    choices [UNTAGGED] SEQUENCE OF [UNTAGGED] CHOICE {
        and LogicalBinaryOperator,
        or LogicalBinaryOperator,
        xor LogicalBinaryOperator,
        not LogicalUnaryOperator,
        equal-to [NAME AS "equal_to"] ArithmeticBinaryOperator,
        not-equal-to [NAME AS "not_equal_to"] ArithmeticBinaryOperator,
        greater-than [NAME AS "greater_than"] ArithmeticBinaryOperator,
        greater-than-or-equal-to [NAME AS "greater_than_or_equal_to"]
            ArithmeticBinaryOperator,
        less-than [NAME AS "less_than"] ArithmeticBinaryOperator,
        less-than-or-equal-to [NAME AS "less_than_or_equal_to"]
            ArithmeticBinaryOperator,
        same-as [NAME AS "same_as"] StringBinaryOperator,
        not-same-as [NAME AS "not_same_as"] StringBinaryOperator,
        existing VariableOperator,
        not-existing [NAME AS "not_existing"] VariableOperator
    }
}
ENCODING-CONTROL XER
GLOBAL-DEFAULTS MODIFIED-ENCODINGS
TEXT Model:bspTesting AS «BSPTesting"
END
```

Приложение С
(обязательное)

XML-схема протокола теста

```
<?xml version='1.0' encoding="utf-8"?>
<xs:schema xmlns:xs="gost.e-ecolog.ru                    ">

   <xs:element name="conformance_test_log"
         type="conformance_test_log"/>

   <xs:complexType name="conformance_test_log">
      <xs:sequence>
         <xs:element name="TestingLaboratory" type="Contact"/>
         <xs:element name="Vendor" type="Contact"/>
         <xs:element name="Biometric_Product" type="Biometric_Product"/>
         <xs:element name="CTS_ID" type="xs:string"/>
         <xs:element name="test_assertion" type="test_assertion" />
         <xs:group ref="func_inline_response_group"
               minOccurs="0" maxOccurs="unbounded" />
      </xs:sequence>
      <xs:attribute name="date_time" type="xs:string" use="required"/>
      <xs:attribute name="standard" type="xs:string" />
   </xs:complexType>

   <xs:group name="func_inline_response_group">
      <xs:choice>
         <xs:element name="function" type="function"/>
         <xs:element name="inline_response" type="inline_response" />
      </xs:choice>
   </xs:group>

   <xs:complexType name="Contact">
      <xs:sequence>
         <xs:element name="Name" type="xs:string"/>
         <xs:element name="Address" type="Address" />
         <xs:element name="Phone" type="xs:string"/>
         <xs:element name="Email" type="xs:string" minOccurs="0" />
         <xs:element name="Url" type="xs:string" minOccurs="0" />
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="Address">
      <xs:sequence>
         <xs:element name="Street" type="xs:string"/>
         <xs:element name="City" type="xs:string" />
         <xs:element name="StateOrProvince" type="xs:string" minOccurs="0"/>
         <xs:element name="ZipOrPostalCode" type="xs:string" />
         <xs:element name="Country" type="xs:string" />
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="Biometric_Product">
      <xs:sequence>
         <xs:element name="Description" type="xs:string"/>
      </xs:sequence>
      <xs:attribute name="Name" type="xs:string" use="required" />
      <xs:attribute name="SerialNo" type="xs:string" use="required"/>
   </xs:complexType>
```

```
    <xs:complexType name="test_assertion">
        <xs:sequence>
            <xs:element name="package_name" type="xs:string" />
            <xs:element name="assertion_name" type="xs:string" />
            <xs:element name="description" type="xs:string" />
            <xs:element name="input" type="inputOutputType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="inputOutputType">
        <xs:attribute name="name" type="xs:NCName" use="required"/>
        <xs:attribute name="value" type="xs:string" use="required"/>
    </xs:complexType>

    <xs:complexType name="function">
        <xs:sequence>
            <xs:element name="name" type="xs:string" />
            <xs:element name="input" type="inputOutputType" minOccurs="0" maxOccurs="unbounded" />
            <xs:element name="output" type="inputOutputType" minOccurs="0" maxOccurs="unbounded" />
            <xs:element name="return" type="return" />
        </xs:sequence>
        <xs:attribute name="dir" type="calldirection" default="outgoing"/>
    </xs:complexType>

    <xs:simpleType name="calldirection">
        <xs:restriction base="xs:token">
            <xs:enumeration value="outgoing"/>
            <xs:enumeration value="incoming"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:complexType name="return">
        <xs:attribute name="value" type="xs:string" use="required"/>
    </xs:complexType>

    <xs:complexType name="inline_response">
        <xs:sequence>
            <xs:element name="asserted_condition" type="xs:string"/>
            <xs:element name="conformance">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:enumeration value="error" />
                        <xs:enumeration value="pass"/>
                        <xs:enumeration value="fail"/>
                        <xs:enumeration value="undecided"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>

</xs:schema>
```

Приложение D
(справочное)

Руководство по проведению испытаний

Приложение представляет собой руководство по проведению испытаний на соответствие биометрического программного интерфейса, включая элементы испытания на соответствие и описание категорий испытаний.

D.1 Введение

D.1.1 Процесс испытаний на соответствие является завершающим для всех операций, которые необходимы для подтверждения соответствия ТР спецификации BioAPI в соответствии с ИСО 19784-1. Процесс испытаний на соответствие состоит из трех этапов:

- конфигурирование и подготовка к проведению испытаний;
- разработка методики испытания;
- проведение испытаний на соответствие.

D.2 Конфигурирование и подготовка к проведению испытаний

D.2.1 После того как программное обеспечение КТСБПИ доставлено в испытательную лабораторию, необходимо проверить наличие следующей сопроводительной документации:

- руководства по установке и настройке КТСБПИ;
- руководства по подготовке и конфигурированию условий испытания;
- руководств для пользователя и администратора КТСБПИ, включая руководство по устранению неполадок.

D.2.2 Затем лаборатория устанавливает и настраивает КТСБПИ и ТР. Процесс считается завершенным, если и КТСБПИ, и ТР успешно установлены, настроены в соответствии с прилагающейся документацией и готовы к работе в соответствии с ПСБПИ в данных условиях испытания (с использованием данных аппаратного и программного обеспечения).

D.2.3 Если при проведении испытания необходим набор конкретных данных, то он должен быть подготовлен до начала испытаний на соответствие.

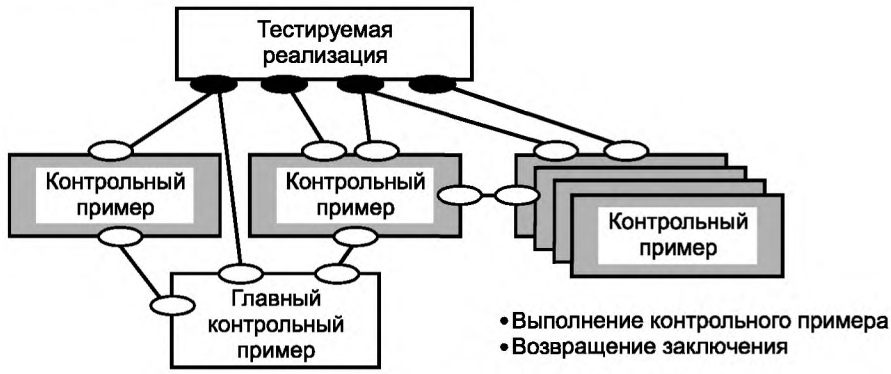


Рисунок D.1 — Схема проведения испытания

D.3 Категории функциональных групп тестов

D.3.1 При проведении испытаний на соответствие основное внимание должно быть направлено на определение соответствия спецификации BioAPI, описанной в ИСО 19784-1; помимо этого необходимо установить правильность работы функций, описанных в ИСО 19784-1. Для этого используют функциональные тесты, объединенные в четыре группы по виду испытуемой функции:

- функция управления ПБУ;
- функция управления ЗБИ;
- функция верификации;
- функция идентификации.

Примечание — Выделение вышеупомянутых групп позволяет провести более полную проверку соответствия спецификации BioAPI.

D.3.1.1 Функция управления ПБУ отвечает за управление областью памяти ПБУ. Если для работы выбран модуль ПБУ, должно происходить следующее.

D.3.1.2 Функция инициализирует модуль услуги. Процесс инициализации включает в себя регистрацию обработчика событий модуля и запуск всех событий. Приложение может предусматривать функцию обработки событий для получения уведомлений о событиях добавления и удаления данных, а также о событии появления

пользователя на рабочем месте. Многие приложения могут независимо и одновременно загружать одного и того же ПБУ, при этом каждое приложение может устанавливать собственного обработчика событий и, таким образом, все приложения получат уведомление о событии.

D.3.1.3 BSPLoad: функция BioAPI_Init должна быть вызвана до вызова BioAPI_BSPLoad. Функция BioAPI_BSPAttach может быть вызвана несколько раз во время вызова BioAPI_BSPLoad.

D.3.1.4 BSPAttach: эта функция присоединяет модуль поставщика услуги и проверяет его версию на соответствие версии, ожидаемой приложением. В вызывающем процессе должен быть определен присоединяемый модуль.

D.3.1.5 BSPDetach: эта функция отсоединяет приложение от модуля поставщика услуги.

D.3.1.6 BSPUnload: эта функция отключает события и снимает регистрацию функции сообщений о событиях BioAPI. Модуль биометрических услуг может осуществлять отмену действий, выполненных во время инициализации, проведенной функцией BioAPI_BSPUnload.

D.3.1.7 Испытание функций загрузки и присоединения ПБУ: в случае успешного завершения вызова функции BioAPI_BSPLoad (с возвращением значения BioAPI_OK), вызывается функция BioAPI_BSPAttach. Если вызов функции BioAPI_OK завершается возвращением значения BioAPI_OK, то функции загрузки и присоединения ПБУ считают выдержавшими испытание с результатом **"pass"**.

D.3.1.8 Испытание функций отсоединения и выгрузки ПБУ: в случае успешного завершения вызова функции BioAPI_BSPDetach (с возвращением значения BioAPI_OK), вызывается функция BioAPI_BSPUnload. Если вызов функции BioAPI_BSPUnload завершается возвращением значения BioAPI_OK, то функции отсоединения и выгрузки ПБУ считают выдержавшими испытание с результатом **"pass"**.

D.3.1.9 Лаборатория, проводящая испытания: сертификация соответствия должна быть основана на оценке соответствия третьей стороной (сторонней организацией).

D.4 Сценарии испытания

D.4.1 Сценарий для проверки функций управления ЗБИ

D.4.1.1 Для проверки управления ЗБИ необходимы следующие функции:

- **BioAPI_Capture;**
- **BioAPI_FreeBIRHandle;**
- **BioAPI_GetBIRFromHandle;**
- **BioAPI_GetHeaderFromHandle.**

D.4.1.2 Для тестирования функции создания ЗБИ необходимо запустить BioAPI_Capture до запуска других функций управления ЗБИ, потому что для создания ЗБИ необходим биометрический образец. Схема испытания приведена на рисунке D.2.

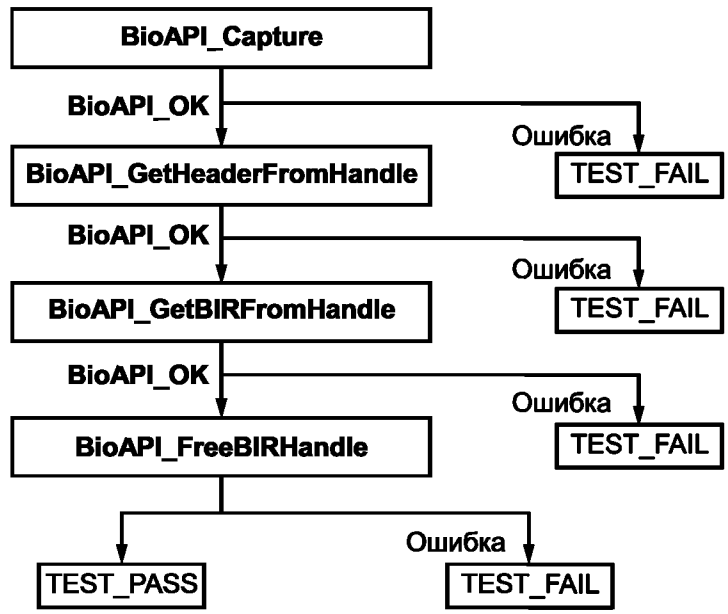


Рисунок D.2 — Схема испытания функции, осуществляющей управление ЗБИ

D.4.2 Сценарий для проверки функций верификации

D.4.2.1 Функции, необходимые для проверки верификации, приведены ниже. Для проведения верификации сначала ПБУ или приложение должны предварительно получить биометрический образец и определить его соответствие (или несоответствие) хранящемуся биометрическому шаблону. Данный этап испытания изображен на рисунке D.3:

- **BioAPI_Enroll;**
- **BioAPI_Verify;**

- BioAPI_Capture;
- BioAPI_CreateTemplate;
- BioAPI_Process;
- BioAPI_VerifyMatch.

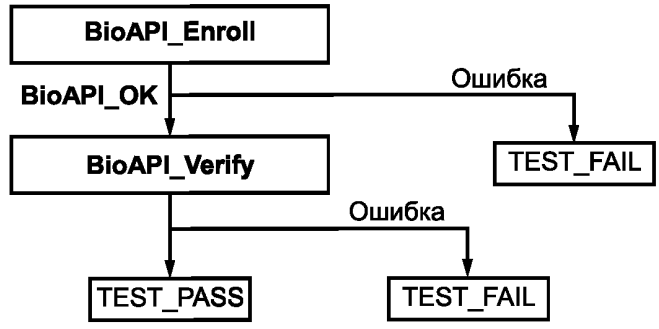


Рисунок D.3 — Схема испытания функции верификации

D.4.2.2 Проверка регистрации биометрического образца должна быть проведена до начала проверки верификации. В результате проверки должно быть установлено, был ли вызов функций BioAPI_Verify и BioAPI_Identify проведен после вызова функции BioAPI_Enroll. С помощью функции BioAPI_Capture осуществляют регистрацию образца, который затем обрабатывают с помощью функции BioAPI_Process. Работа по идентификации выполняется функцией BioAPI_IdentifyMatch. Данный этап испытания изображен на рисунке D.4.

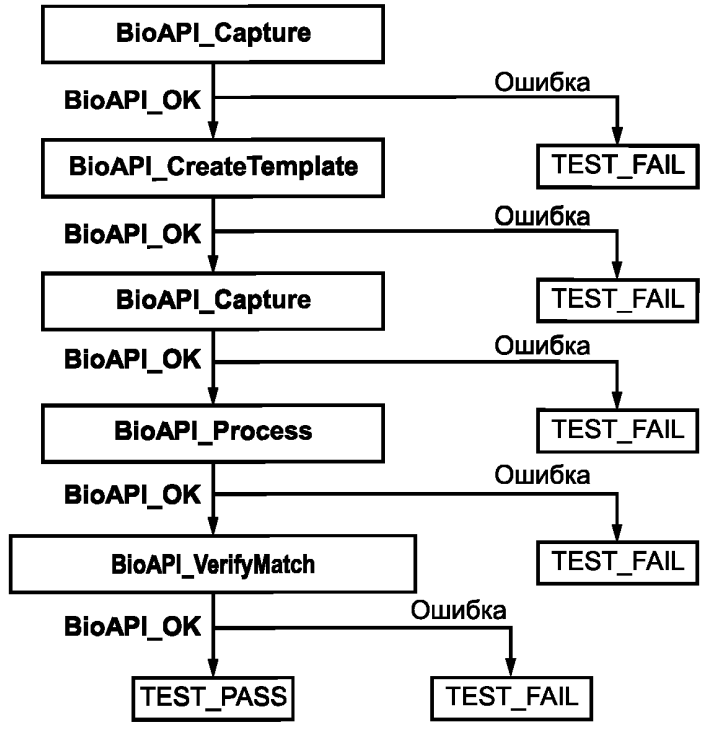


Рисунок D.4 — Схема испытания процесса верификации

D.4.3 Сценарий для проверки функции идентификации

D.4.3.1 Идентификация представляет собой процесс сравнения представленного биометрического образца с контрольной выборкой шаблонов (схема «один ко многим») с целью определения соответствия образца какому-либо из контрольных шаблонов в данной контрольной выборке для установления человека, соответствующего шаблону. Функции, необходимые для проверки идентификации, приведены ниже. Для проведения верификации сначала ПБУ или приложение должны предварительно получить биометрический образец и определить его соответствие (или несоответствие) хранящемуся биометрическому шаблону. Этот этап тестирования изображен на рисунке D.5:

- BioAPI_Enroll;
- BioAPI_Verify;
- BioAPI_Identify;
- BioAPI_Capture;
- BioAPI_CreateTemplate;
- BioAPI_Process;
- BioAPI_IdentifyMatch.

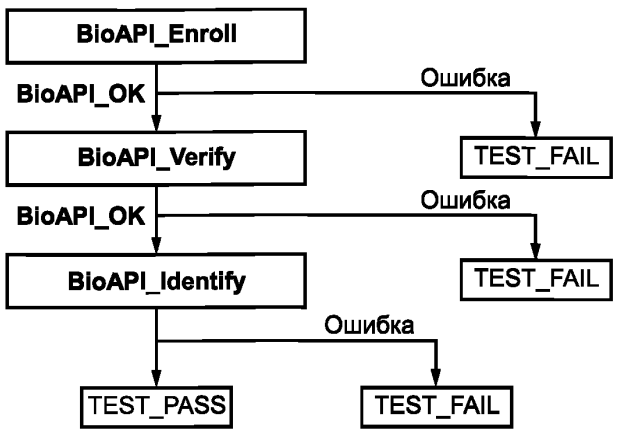


Схема D.5 — Схема испытания функции идентификации

D.4.3.2 Проверка регистрации биометрического образца должна быть выполнена до начала проверки идентификации. В результате проверки должно быть установлено, был ли вызов функций BioAPI_Verify и BioAPI_Identify после вызова функции BioAPI_Enroll. С помощью функции BioAPI_Capture осуществляется регистрация образца, который затем обрабатывается с помощью функции BioAPI_Process. Работа по идентификации выполняется функцией BioAPI_IdentifyMatch. Данный этап тестирования изображен на рисунке D.6.

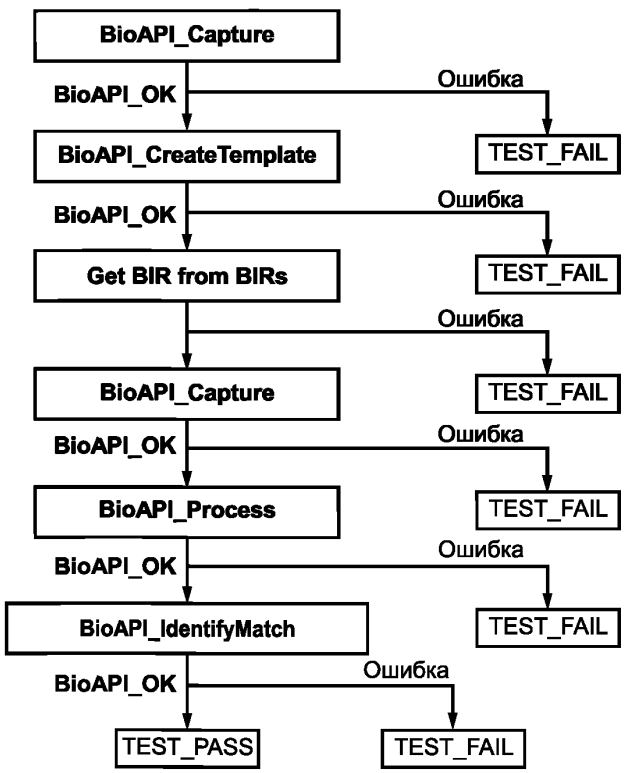


Рисунок D.6 — Схема испытания процесса идентификации

Приложение Е
(справочное)

Программа оценки соответствия BioAPI

Е.1 Введение

В данном приложении описана общая структура процесса определения соответствия BioAPI. Описание основано на стандартах, одобренных ИСО и относящихся к определению соответствия. Отдельные реализации процесса определения соответствия могут различаться, однако все реализации должны содержать все элементы структуры процесса, описанные ниже.

Е.2 Общие положения

Программы оценки соответствия разработаны для спецификаций и стандартов, используемых для важнейших приложений, совместной работы с другими приложениями, а также для обеспечения безопасности систем. Биометрические системы, базирующиеся на спецификации BioAPI, удовлетворяют большинству из указанных условий, что доказывает необходимость применения программ определения соответствия BioAPI. Такая программа, в общих чертах, состоит из двух частей: оценки соответствия и сертификации соответствия.

Оценка соответствия определена в документе [1] как «систематическое исследование того, в какой мере продукт, процесс или услуга удовлетворяют определенным требованиям», в данном случае — требованиям спецификации BioAPI. Оценка соответствия реализаций BioAPI должна включать в себя: инспектирование, которое представляет собой «оценку соответствия путем осмотра и принятия решения, как правило, сопровождающуюся измерением, тестированием или калибровкой» и испытания на соответствие, которое является «определением соответствия путем тестирования».

Сертификат соответствия представляет собой письменное заверение, предоставленное третьей стороной, в том, что «продукт, процесс или услуга удовлетворяет указанным требованиям». Другими словами, сертификат соответствия является признанием того, что определение соответствия завершено и требования, установленные сертификационной организацией и необходимые для выдачи сертификата, выполнены, то есть удачное завершение испытаний на соответствие приводит к выдаче сертификата (и/или отметки) соответствия тестируемой реализации спецификации BioAPI.

Сертификат не должен рассматриваться в качестве гарантии отсутствия в сертифицированном продукте дефектов или несоответствий; он лишь подтверждает, что во время оценки и процесса сертификации несоответствия не были найдены.

Оценка соответствия с сертификацией могут стать предпосылкой к началу поставок. Важно отметить, что сертификация не может существовать без оценки, однако оценка может существовать без сертификации.

Несмотря на то что оценка и сертификация могут проводиться разными организациями, важно, чтобы для программы оценки соответствия была предусмотрена централизованная программа финансирования. Финансирующая компания, агентство или консорция заинтересованы в успехе программы и, как правило, они разрабатывают и обслуживают программу, а также берут на себя обязанность управления программой в целом и становятся централизованным источником информации о программе.

Компания, производящая биометрическое оборудование в коммерческих целях, не должна являться спонсором программы оценки соответствия, но объединение таких компаний может стать спонсором в том случае, если она руководит программой таким образом, что это не дает преимущества отдельному поставщику биометрических услуг или группе поставщиков.

В программу оценки соответствия BioAPI должны входить следующие элементы:

a) спецификация BioAPI, которая включает в себя раздел о соответствии и хотя бы одну надежную ссылочную реализацию спецификации, которая, по определению, соответствует настоящему стандарту. Спецификация устанавливает требования, которым должна удовлетворять ТР для подтверждения соответствия. Раздел о соответствии определяет применимость (то есть, что должно соответствовать и/или быть испытано) и указывает условия, которые должны быть соблюдены, чтобы подтвердить соответствие;

b) хотя бы один КТСБПИ, являющийся сочетанием исполняемых контрольных примеров, с соответствующими тестовыми данными, процедурой тестирования и сопроводительной документацией КТСБПИ;

c) правила оценки соответствия или порядок осуществления действий, определенный в ISO/IEC Guide 60 «Conformity assessment — Code of good practice», который определяет действия организации, ответственной за проведение испытаний, разрешение споров и выдачу сертификата соответствия. Политика может включать в себя рабочие процедуры, включающие в себя (но не ограниченные указанными), процедуры аккредитации для испытательных лабораторий сторонней организации и процедуры утверждения КТСБПИ. Политика должна фиксировать действия, продукцию, услуги, отчетность, ответственность, полномочия и руководство, необходимые для программы оценки соответствия. Также она предназначена для информирования клиента, который хочет подвергнуть свой продукт оценке на соответствие BioAPI о требованиях, ожидаемых результатах и ресурсах, предоставляемых им и требуемых от него, а также о процедурах, которые необходимо выполнить для получения сертификата, и о правилах, которым они согласны придерживаться.

Е.3 Процесс оценки соответствия

В данном подразделе описана общая структура процесса оценки соответствия BioAPI. Отдельные реализации процесса оценки соответствия могут варьироваться, однако они должны содержать все элементы структуры процесса.

Определение соответствия BioAPI и процесс сертификации начинаются тогда, когда поставщик биометрического продукта и испытательная лаборатория заключают контракт на проведение испытания биометрической продукции на соответствие BioAPI. Поставщик и испытательная лаборатория обсуждают область применения испытаний, их стоимость и последовательность выполнения. Испытательная лаборатория не должна находиться в положении, в котором может получить прибыль или понести убытки в зависимости от результата испытания представленной биометрической продукции. Испытательная лаборатория должна быть аккредитованной лабораторией, которая соответствует требованиям ISO/IEC 17025 [6].

Испытуемую продукцию подвергают испытаниям в определенной среде, включающей в себя аппаратные средства компьютера и программное обеспечение, необходимые для поддержки реализации, и надежную образцовую реализацию спецификации BioAPI. Испытания проводят по утвержденной методике. Испытание заключается в проведении комплекта тестов на соответствие тестируемой реализации и сравнении полученного результата с результатом, установленным в настоящем стандарте. Испытательная лаборатория обычно проводит оценку после того, как поставщик проведет собственное испытание в качестве подготовки к оценке соответствия.

После испытания результаты записывают в отчет об испытании на соответствие. Отчет также должен содержать данные о поставщике, TP, среде, в которой проводилось испытание, используемом комплекте тестов на соответствие и любые другие данные, собранные в процессе утверждения соответствия. После этого отчет об испытании на соответствие предоставляют на рассмотрение в орган по сертификации («организация, которая осуществляет сертификацию» [1]) для выдачи сертификата соответствия — «документа, выдаваемого по правилам системы сертификации и подтверждающего, что должным образом установленный продукт, процесс или услуга соответствуют требованиям конкретного стандарта или иного нормативного документа» [1].

Если испытание биометрического продукта прошло успешно, все результаты испытания включены в отчет и соблюдены условия, необходимые для выдачи сертификата соответствия, орган по сертификации выдает поставщику данный сертификат. При этом орган по сертификации должен соблюдать необходимые требования, установленные в ISO/IEC 17030 [7] и ISO/IEC Guide 23 [3]. Как правило, орган по сертификации ведет и публикует журнал записей, содержащий список продуктов, получивших сертификат соответствия.

Вопросы и разногласия по поводу методики испытания, процедур, результатов испытания и управления программой оценки соответствия направляют в контролирующий орган (далее — КО). КО должен решить все вопросы и разногласия и сообщить о решении всем участникам. КО действует от имени органа по сертификации.

Е.4 Испытательная лаборатория

Испытательная лаборатория является организацией, которая предоставляет услуги по измерению, проверке, испытаниям или другие услуги, необходимые для оценки соответствия биометрической продукции ее спецификации. Испытательная лаборатория может принадлежать продавцу, покупателю или быть сторонней организацией.

Испытательная лаборатория может быть организацией или физическим лицом, и может быть аккредитована официальной аккредитационной организацией, такой как NIST NVLAP (Национальный институт стандартов и технологий США, Национальная программа добровольной лабораторной аккредитации) или быть признана покупателем, продавцом или обладателем сертификата соответствия, имеющим право проводить испытания.

Испытательная лаборатория проводит испытания на соответствие с использованием комплекта тестов на соответствие биометрическому программному интерфейсу (КТСБПИ) по утвержденной методике испытания. Настоящий стандарт не запрещает проведение испытаний изготовителем биометрической продукции, однако полная оценка соответствия спецификации BioAPI допускается только с привлечением сторонних испытательных лабораторий, которые проводят испытания на соответствие согласно установленным правилам. Сертификат соответствия должен основываться на оценке соответствия сторонней организацией.

Согласно положениями ISO/IEC 17025 [6], существует ряд общих требований, которым должна соответствовать лаборатория, чтобы быть признанной компетентной для проведения особых испытаний. Лаборатория должна соответствовать, по крайней мере, следующим требованиям.

Лаборатория должна быть правомерной с юридической точки зрения. Она должна быть организована и функционировать таким образом, чтобы ее постоянное, временное и мобильное оборудование соответствовало требованиям ИСО/МЭК 17025. Лаборатория должна иметь:

a) управленческий персонал с полномочиями и ресурсами, необходимыми для исполнения своих обязанностей;

b) соглашения, доказывающие, что персонал не подвергается коммерческому, финансовому и любому иному давлению, которое может отрицательно влиять на качество выполняемой работы;

c) технического директора, ответственного за все технические операции;

d) руководителя, ответственного за системы контроля качества и их реализацию.

Лаборатория должна организовать и обслуживать систему контроля качества в соответствии с требованиями испытаний на соответствие биометрического продукта. Элементы системы контроля качества должны быть

задокументированы. Документация по управлению качеством должна быть доступна персоналу лаборатории. Лаборатория должна определить и документально зафиксировать свою политику и цели в руководстве по качеству (документе, определяющем систему менеджмента качества организации). Руководство по качеству должно постоянно обновляться руководителем системы контроля качества. Руководство по качеству должно включать в себя:

a) утверждение о политике в области качества, включающее в себя цели и обязательства, составленное высшим руководящим составом организации;

b) структуру организации и структуру управления лаборатории, их место в вышестоящей организации и соответствующие организационные диаграммы;

c) связи между администрацией, техническими операциями, службами поддержки и системой контроля качества;

d) процедуры для контролируемого поддерживания документации в надлежащем состоянии;

e) должностные инструкции основного персонала и ссылки на должностные инструкции другого персонала;

f) сферу деятельности испытательной лаборатории;

g) ссылку на использованные методы испытаний, включающих в себя процедуры, которые необходимо выполнить для возвращения к исходной точке и введения поправок, когда бы ни были обнаружены несоответствия, возникшие в процессе испытаний;

h) процедуры проведения проверок;

i) процедуры обеспечения конфиденциальности и защиты прав собственности.

Система контроля качества, принятая для соответствия требованиям ИСО/МЭК 17025 [6], должна ежегодно пересматриваться руководством лаборатории. Все замечания и введенные поправки должны быть задокументированы. Руководитель, отвечающий за качество, должен принять необходимые меры для того, чтобы выполнить эти действия в установленные сроки.

В лаборатории должно быть достаточное число сотрудников с необходимым образованием, подготовкой, техническими знаниями и опытом работы для выполнения своих обязанностей. Лаборатория должна вести документальный учет квалификации, подготовки и опыта технического персонала.

Лаборатория должна иметь систему документации для точного определения биометрического продукта до начала испытаний во избежание недоразумений при идентификации аналогичных продуктов.

Персонал лаборатории должен вести записи, позволяющие сохранять данные о всех наблюдениях, вычислениях и полученных данных, отчетах о тестировании и сертификатов соответствия за установленный период. Записи для каждого испытания должны содержать достаточное количество данных, необходимых для его повторения. Все записи, сертификаты и отчеты должны храниться в надежном месте с соблюдением конфиденциальности и недоступности для клиента, если иное не указано в ИСО/МЭК 17025 [6].

Вся конфиденциальная информация должна быть защищена от внешних и внутренних опасностей и неправильного использования. Любая информация, касающаяся поставщика, чья биометрическая система подвергается испытанию, должна оставаться конфиденциальной до завершения испытания или до тех пор, пока поставщик не согласится на опубликование результатов испытания.

Е.5 Орган по сертификации соответствия

Орган по сертификации несет ответственность за выдачу сертификата соответствия для подтвержденной биометрической продукции. Органом по сертификации может быть торговая ассоциация, консорция, орган по стандартизации или государственная организация, которые удовлетворяют требованиям спецификации ISO/IEC Guide 65 [9]. Орган по сертификации несет ответственность за выдачу сертификатов на продукцию, которая признана соответствующей спецификации BioAPI.

Орган по сертификации может быть назначен по соглашению между поставщиками, покупателями, консорциями, организациями по стандартизации или испытательными лабораториями. Назначение органа по сертификации должно подтверждаться каждые два года.

Управление сертификационной организацией должно осуществляться спонсором программы оценки соответствия BioAPI. Спонсор должен устанавливать порядок работы сертификационной организации таким образом, чтобы не покровительствовать отдельным поставщикам или группам поставщиков биометрической продукции.

Орган по сертификации должен устанавливать процедуры испытаний и затем следовать установленным процедурам. Как правило, орган по сертификации должен следовать правилам, установленным в ISO/IEC Guide 28 [5]. Орган по сертификации также может устанавливать процедуры назначения, управления и поддерживания в должном состоянии КТСБПО и должен гарантировать, что эти процедуры должным образом выполняются.

Орган по сертификации должен просматривать отчеты, полученные испытательной лабораторией в процессе оценки соответствия BioAPI, и утверждать каждый сертификат соответствия. Орган по сертификации должен обеспечить свободный доступ к отчетам об испытании и сертификатам соответствия в случае, если производитель или поставщик биометрического продукта не требуют соблюдения правил конфиденциальности.

Е.6 Орган надзора за оценкой соответствия

При проведении оценки соответствия необходима поддержка консультативного или надзорного органа, роль которого заключается в разрешении споров и предоставлении технического руководства. Орган надзора

должен быть независимой организацией объективных экспертов. На практике, в состав органа надзора должны входить представители Федерального агентства по техническому регулированию и метрологии и испытательных лабораторий.

Орган надзора должен:

a) решать технические или организационные спорные вопросы, возникающие в процессе оценки соответствия BioAPI любой испытательной лабораторией;

b) действовать от имени и выполнять функцию консультативного органа для сертификационной организации и вносить рекомендации по изменению процесса сертификации и/или политики органа по сертификации;

c) содействовать сертификационной организации в формировании критериев для признания испытательных лабораторий и в оценке испытательных лабораторий в соответствии с этими критериями;

d) выступать в качестве посредника между соответствующими организациями по стандартизации и испытательными лабораториями и вести запись всех запросов и соответствующих решений и постановлений органа надзора.

Спорный вопрос может быть поднят поставщиком, испытательной лабораторией или иной организацией (например, разработчиком) в любой момент процесса оценки соответствия. Спорный вопрос должен содержать формулировку проблемы, основание для возникновения и желаемое решение. Контрольный орган должен разрешать все спорные вопросы, которые должны решаться в результате выражения общего мнения или в соответствии с документально оформленной политикой и процедурами органа по надзору.

E.7 Сертификаты соответствия BioAPI

В случае успешного завершения процесса утверждения соответствия результаты испытаний должны быть резюмированы в сертификате соответствия. Таким образом, сертификат соответствия является квинтэссенцией отчета об испытаниях на соответствие. Поскольку этот документ часто используется при покупке биометрической продукции, он должен включать в себя всю необходимую информацию для покупателя и поставщика.

Сертификат соответствия должен включать в себя утверждения органа по сертификации о том, что TP была оценена в определенной среде, в определенный день, с использованием определенного(ых) метода(ов) испытаний, и что результаты испытаний соответствуют установленным требованиям. Сертификат должен иметь срок действия с установленной датой окончания срока действия.

Решение о выдаче сертификата соответствия должно быть основано на результатах оценки и установленных критериях выдачи сертификата. Эти критерии могут предусматривать как полное, так и неполное соответствие биометрической продукции требованиям спецификации BioAPI.

Сертификат соответствия должен соответствовать требованиям ISO/IEC Guide 23:1982 [3] и ISO/IEC 17030:2003 [7]. Сертификат соответствия должен содержать, по крайней мере, следующую информацию:

a) заголовок;

b) уникальный идентификатор сертификата соответствия;

c) точно выраженное определение испытанного биометрического продукта;

d) наименование и адрес предприятия-изготовителя биометрического продукта;

e) ссылку на соответствующий(ие) стандарт(ы);

f) наименование и адрес испытательной лаборатории;

g) утверждение соответствия;

h) дату окончания срока действия сертификата;

i) подпись и указание должности сотрудника(ов), ответственных за содержание сертификата соответствия, и дату выдачи сертификата.

Руководящие указания по принятию результатов оценки соответствия, сертификаты и критерии соответствия, а также действия, необходимые в случае его неправильного использования, приведены в ISO/IEC Guide 68 [10] и ISO/IEC Guide 27 [4].

Приложение F
(справочное)

XML-диаграммы языка утверждений

Все пакеты должны быть корректными в соответствии с XML схемой языка утверждения (см. приложение А).

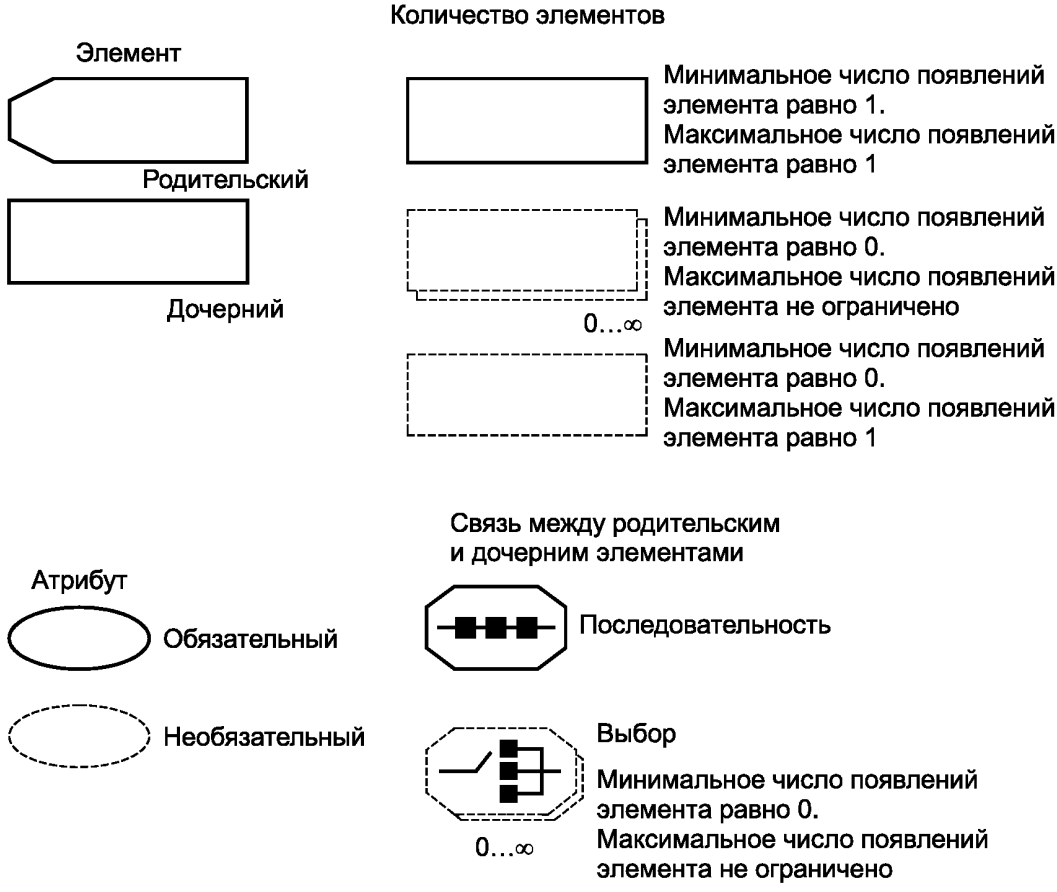


Рисунок F.1 — Пояснения к рисункам F.2 — F.33

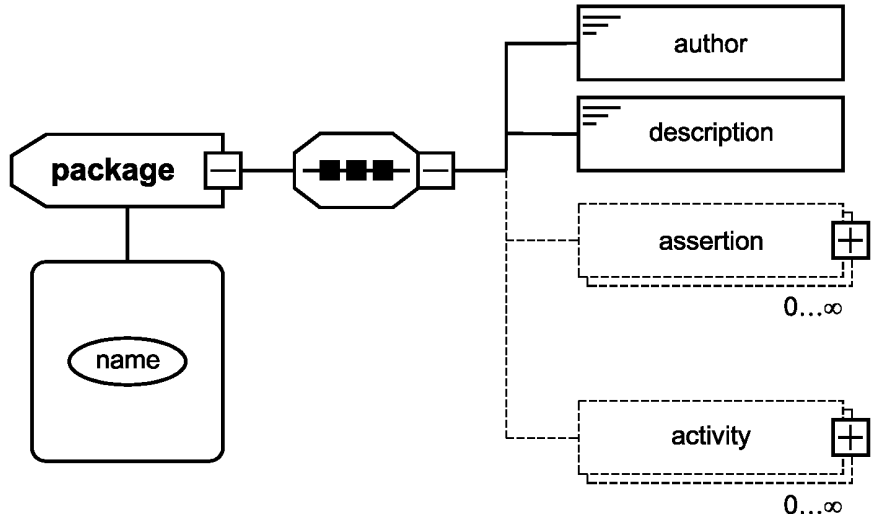


Рисунок F.2 — Элемент **<package>**

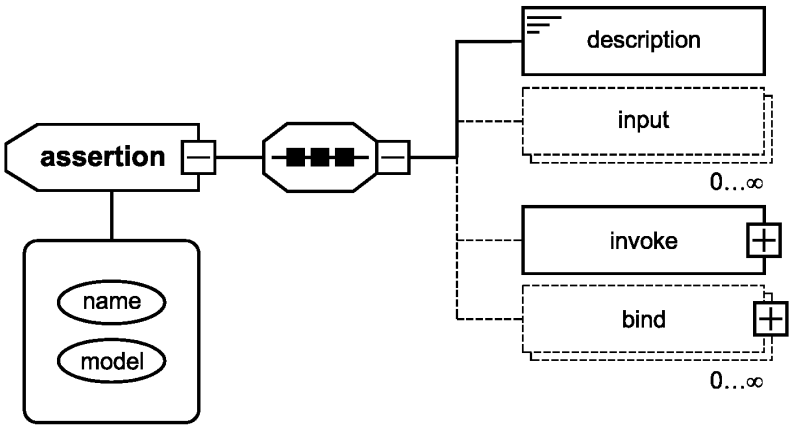


Рисунок F.3 — Элемент **<assertion>** (дочерний для элемента **<package>**)

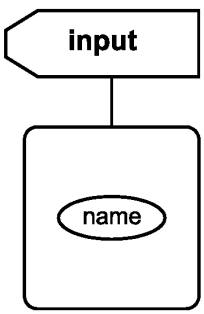


Рисунок F.4 — Элемент **<input>** (дочерний для элемента **<assertion>**)

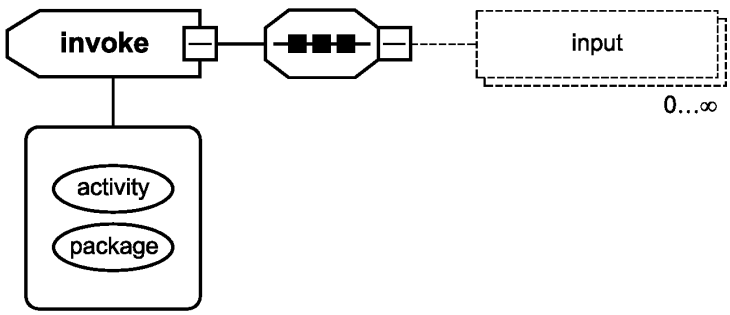


Рисунок F.5 — Элемент **<invoke>** (дочерний для элемента **<assertion>**)

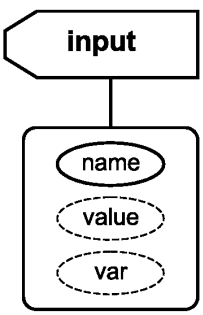


Рисунок F.6 — Элемент **<input>** (дочерний для элемента **<invoke>**)

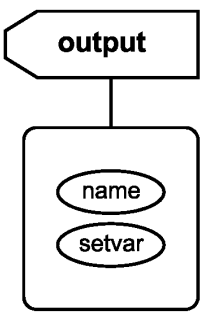


Рисунок F.7 — Элемент **<output>** (дочерний для элемента **<invoke>**)

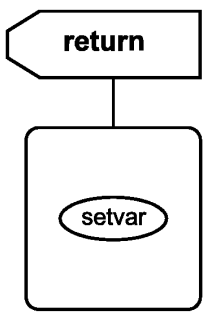


Рисунок F.8 — Элемент **<return>** (дочерний для элемента **<invoke>**)

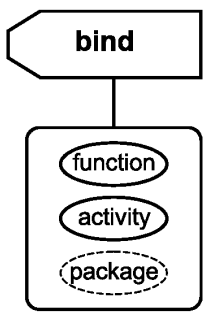


Рисунок F.9 — Элемент **<bind>** (дочерний для элемента **<assertion>**)

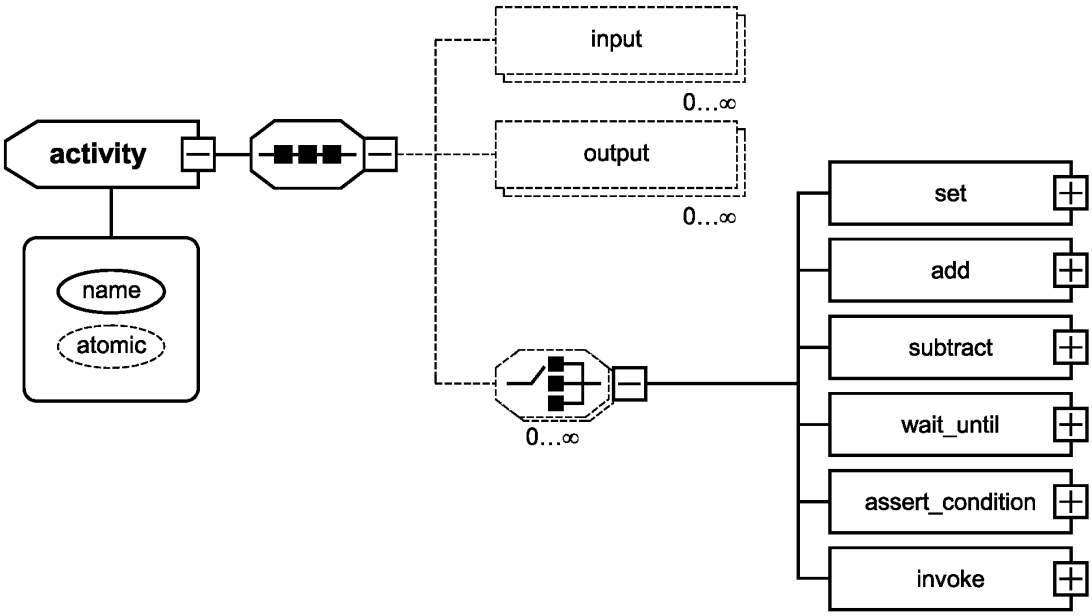


Рисунок F.10 — Элемент **<activity>** (дочерний для элемента **<package>**)

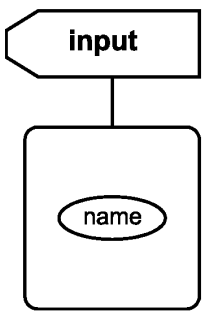


Рисунок F.11 — Элемент **<input>** (дочерний для элемента **<activity>**)

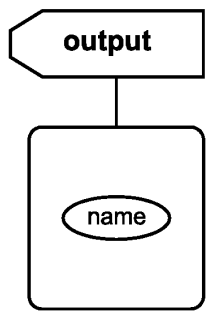


Рисунок F.12 — Элемент **<output>** (дочерний для элемента **<activity>**)

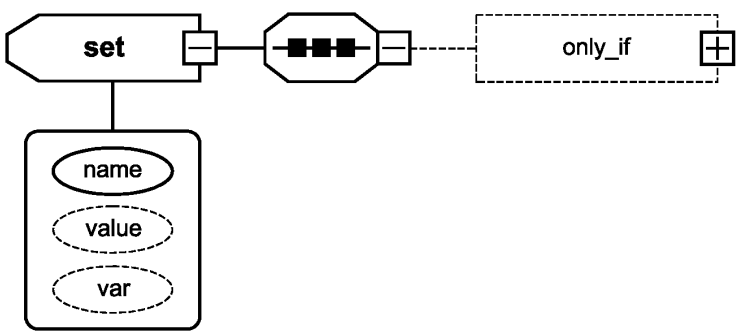


Рисунок F.13 — Элемент **<set>**

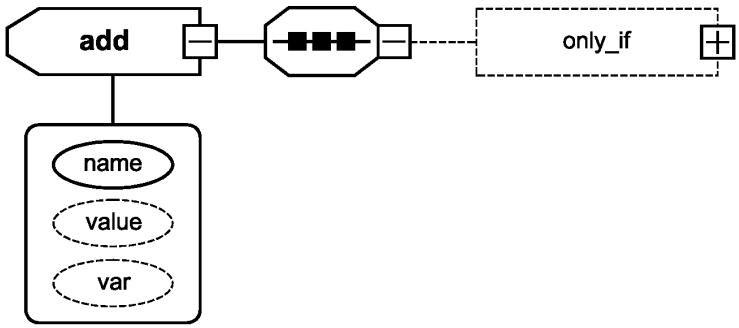


Рисунок F.14 — Элемент **<add>**

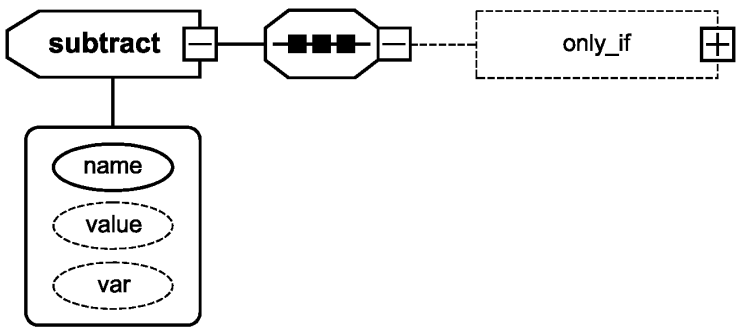


Рисунок F.15 — Элемент **<subtract>**

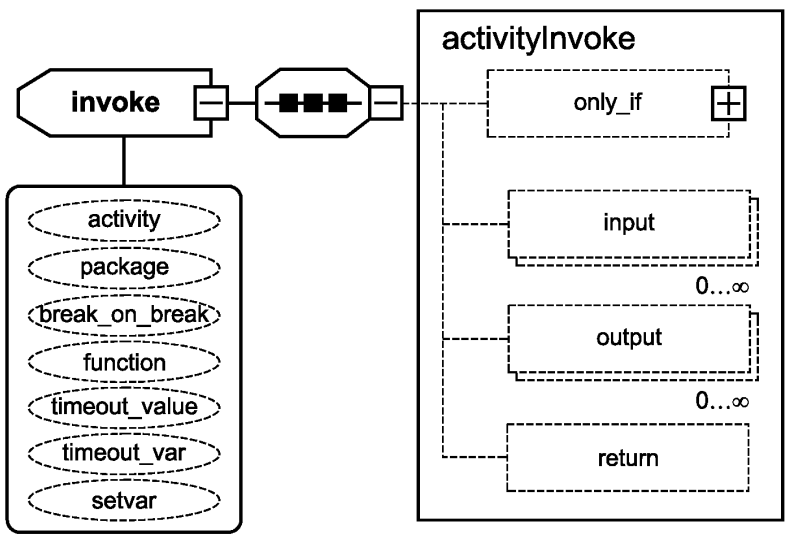


Рисунок F.16 — Элемент **<invoke>** (дочерний для элемента **<activity>**)

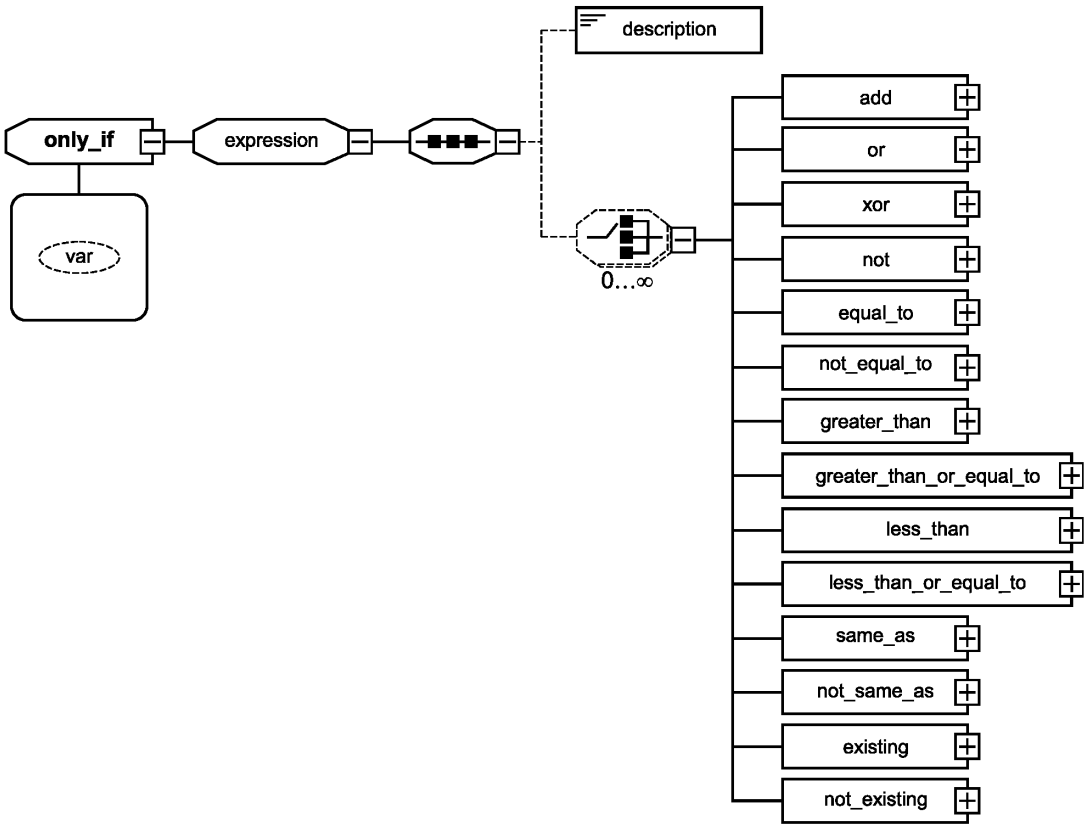


Рисунок F.17 — Элемент **<only_if>**

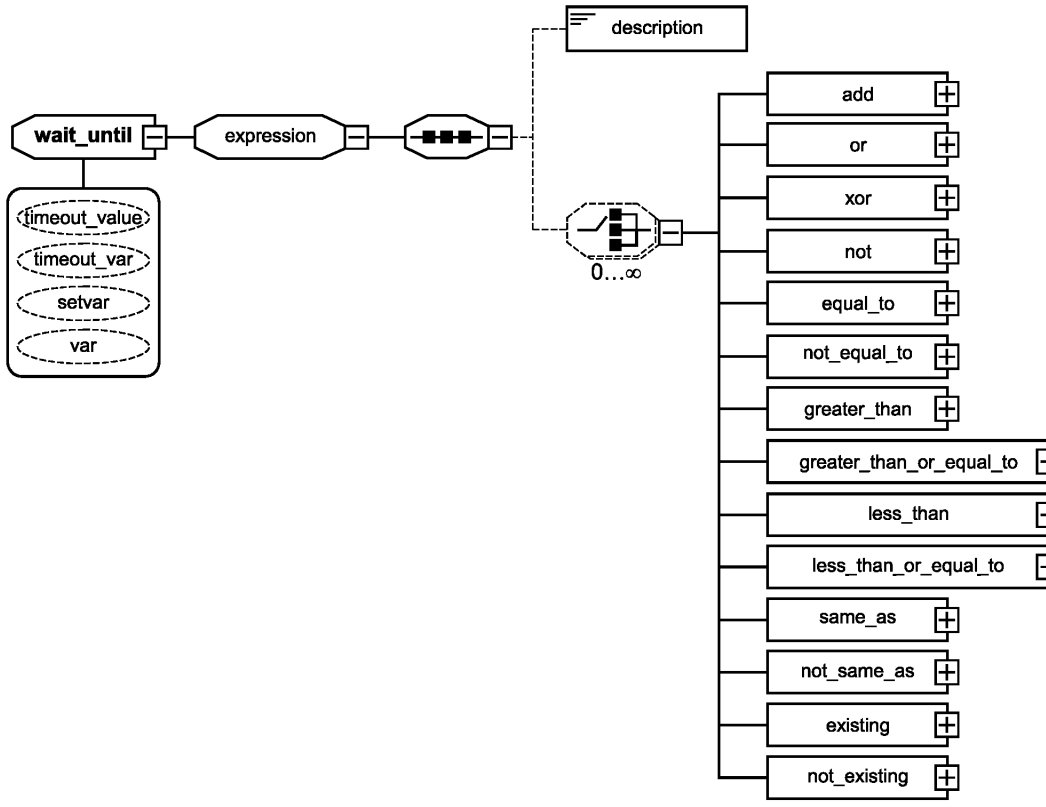


Рисунок F.18 — Элемент **<wait_until>**

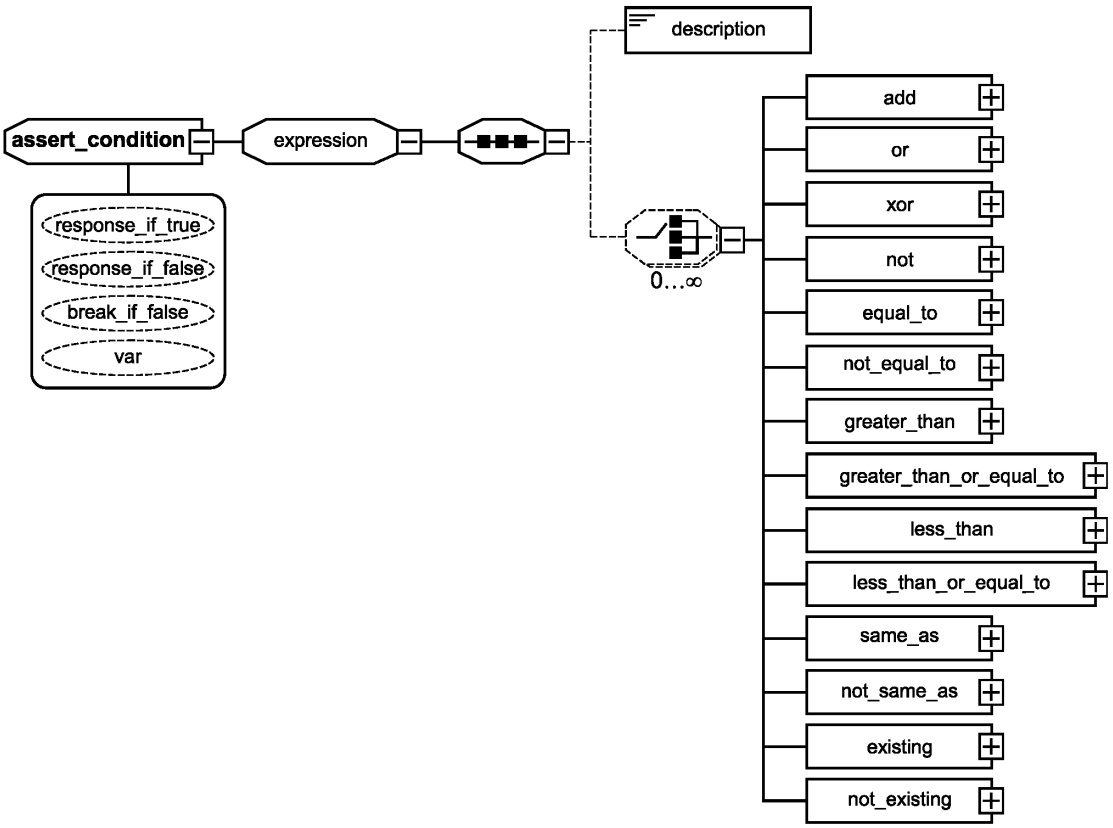


Рисунок F.19 — Элемент **<assert_condition>**

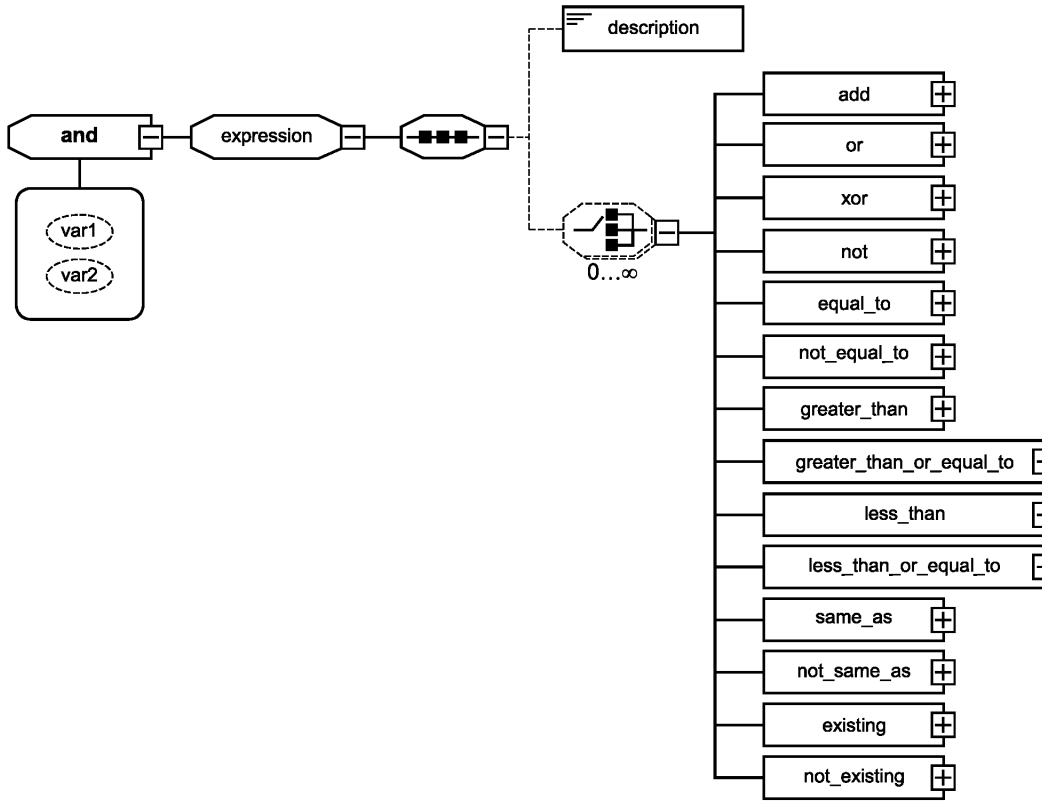


Рисунок F.20 — Элемент **<and>**

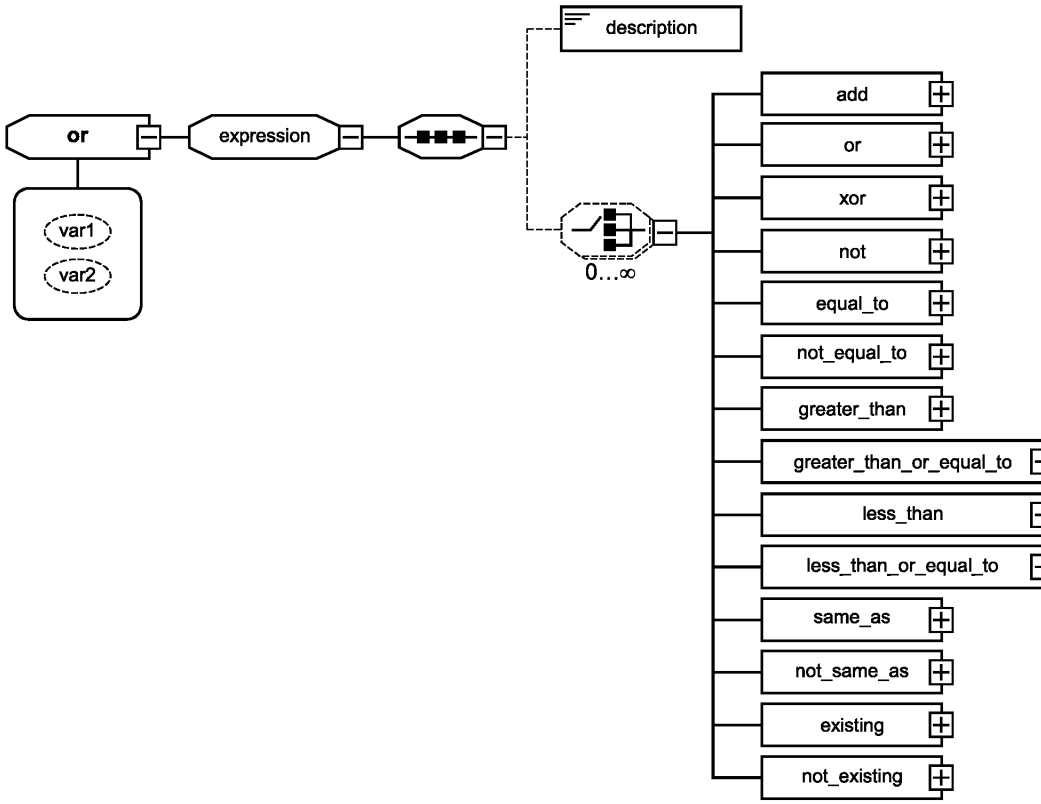


Рисунок F.21 — Элемент **<or>**

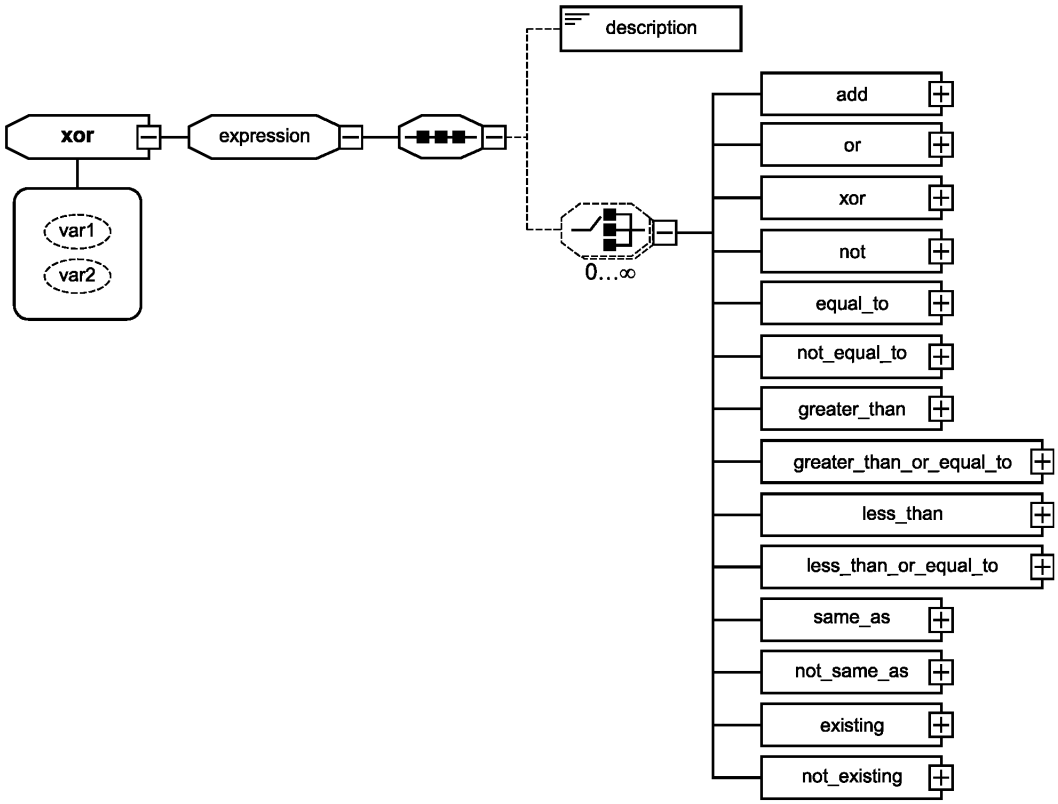


Рисунок F.22 — Элемент **<xor>**

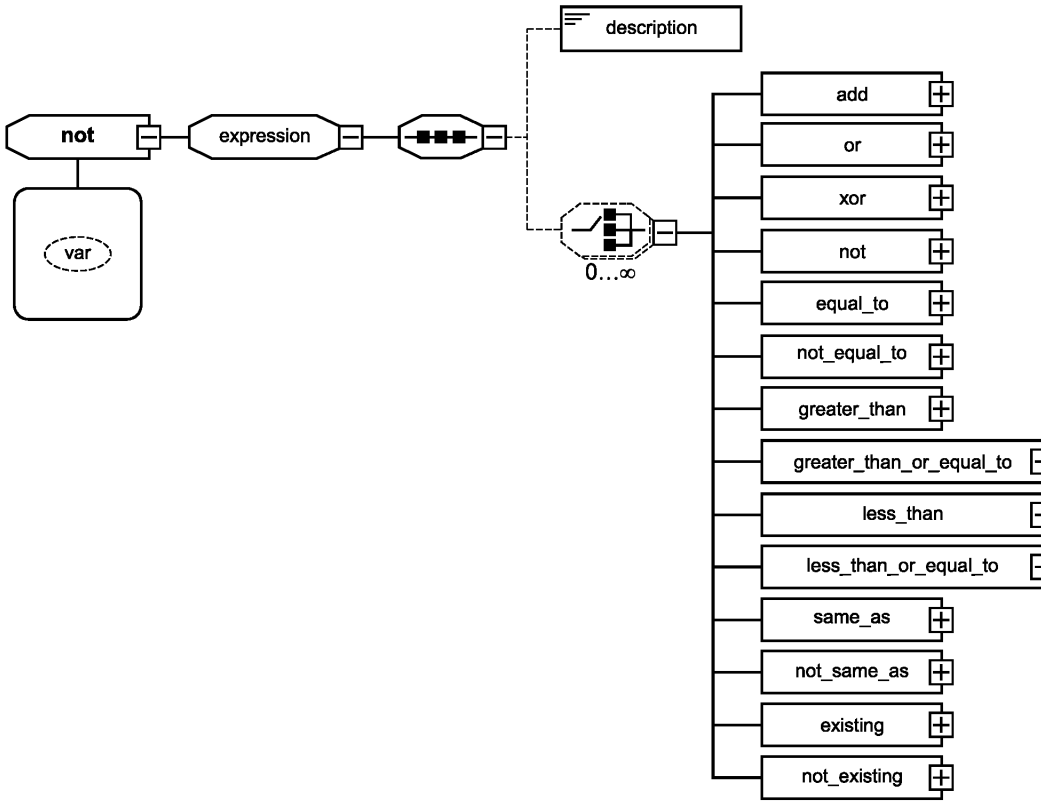


Рисунок F.23 — Элемент **<not>**

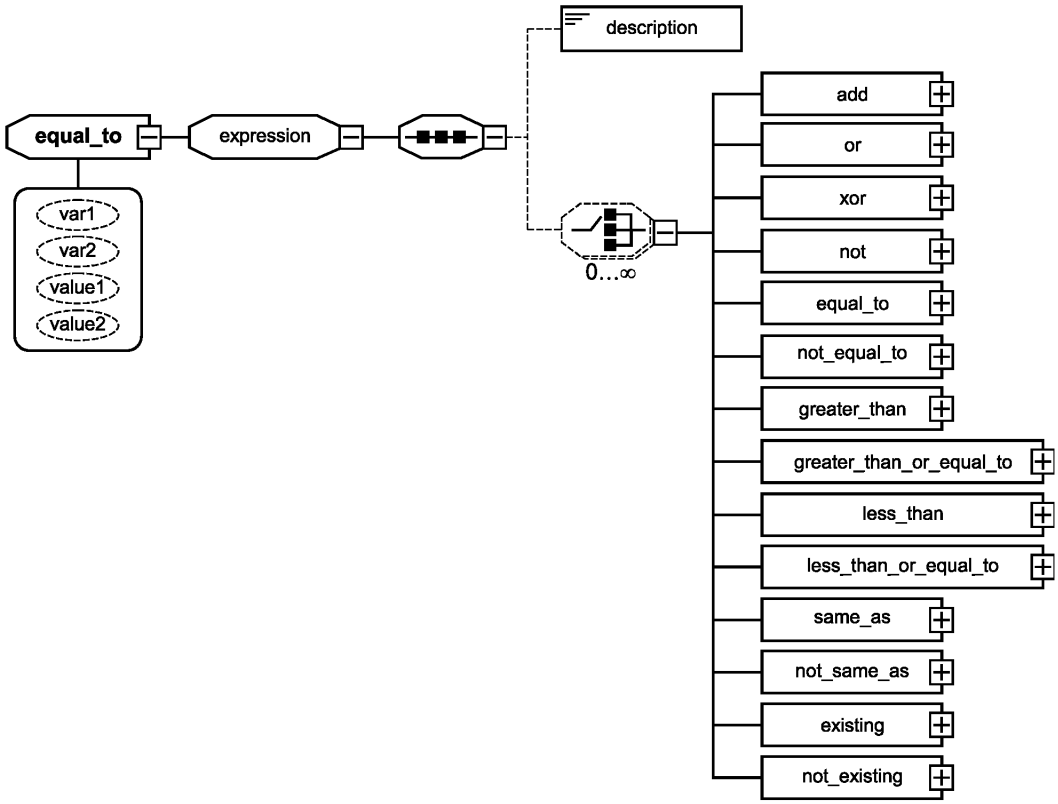


Рисунок F.24 — Элемент **<equal_to>**

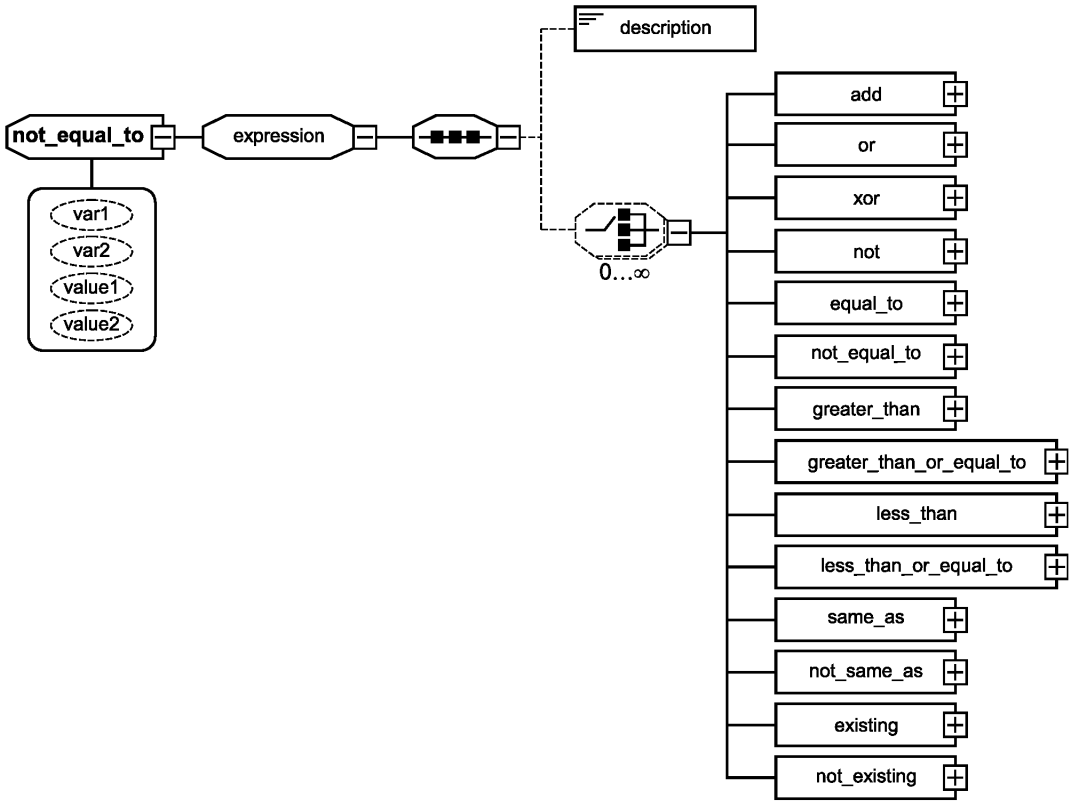


Рисунок F.25 — Элемент **<not_equal_to>**

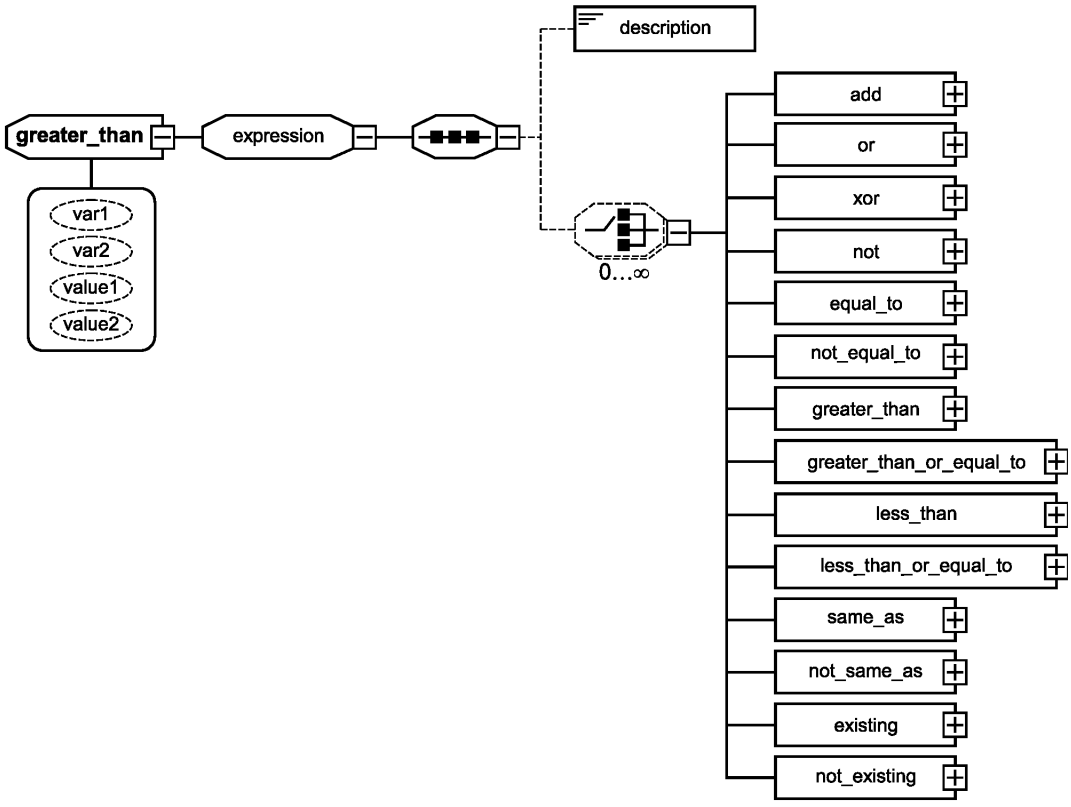


Рисунок F.26 — Элемент **<greater_than>**

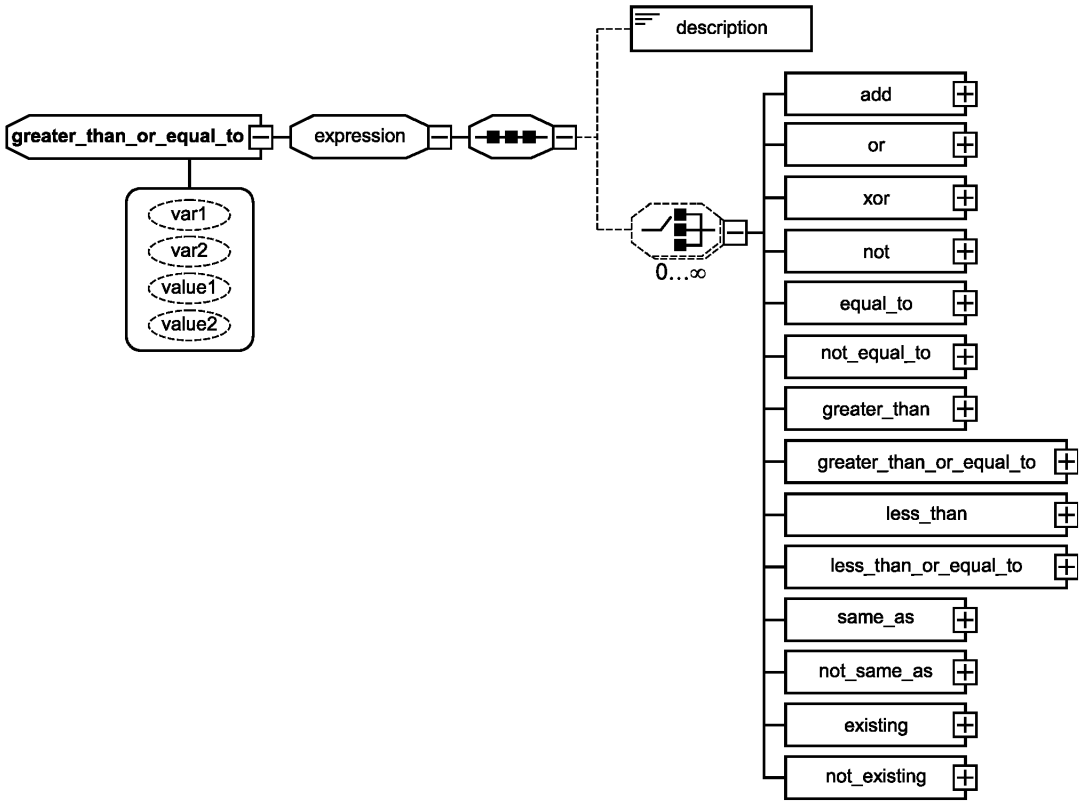


Рисунок F.27 — Элемент **<greater_than_or_equal_to>**

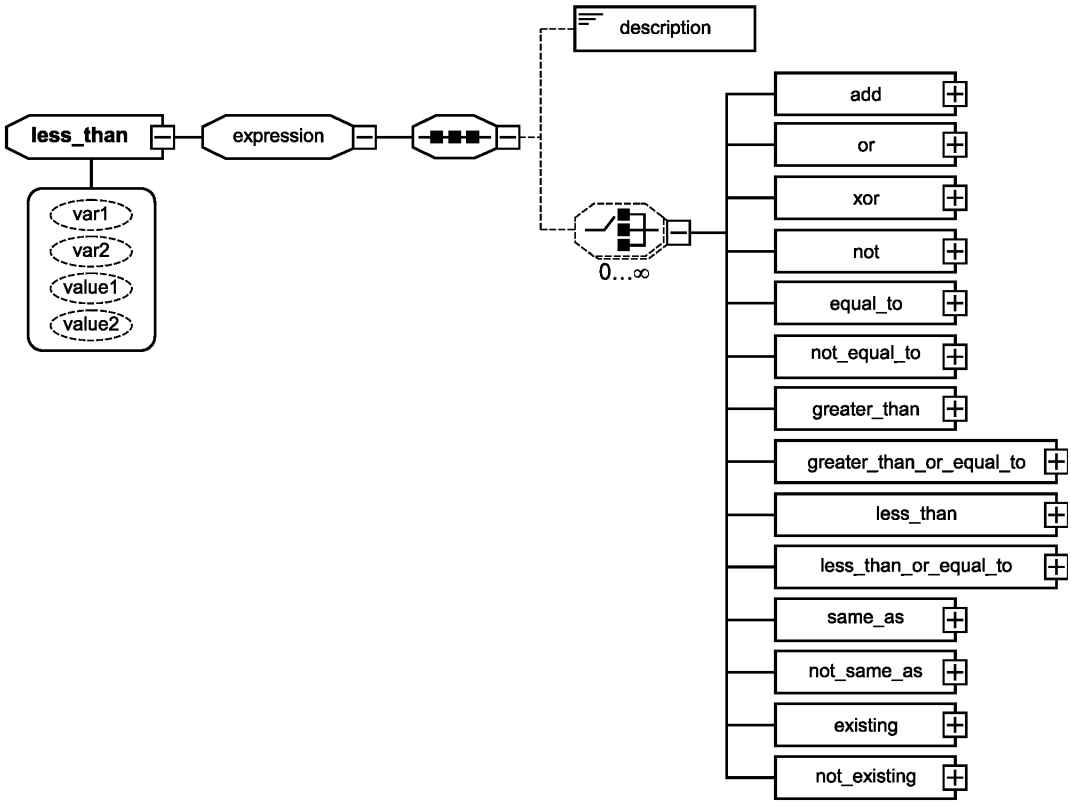


Рисунок F.28 — Элемент **<less_than>**

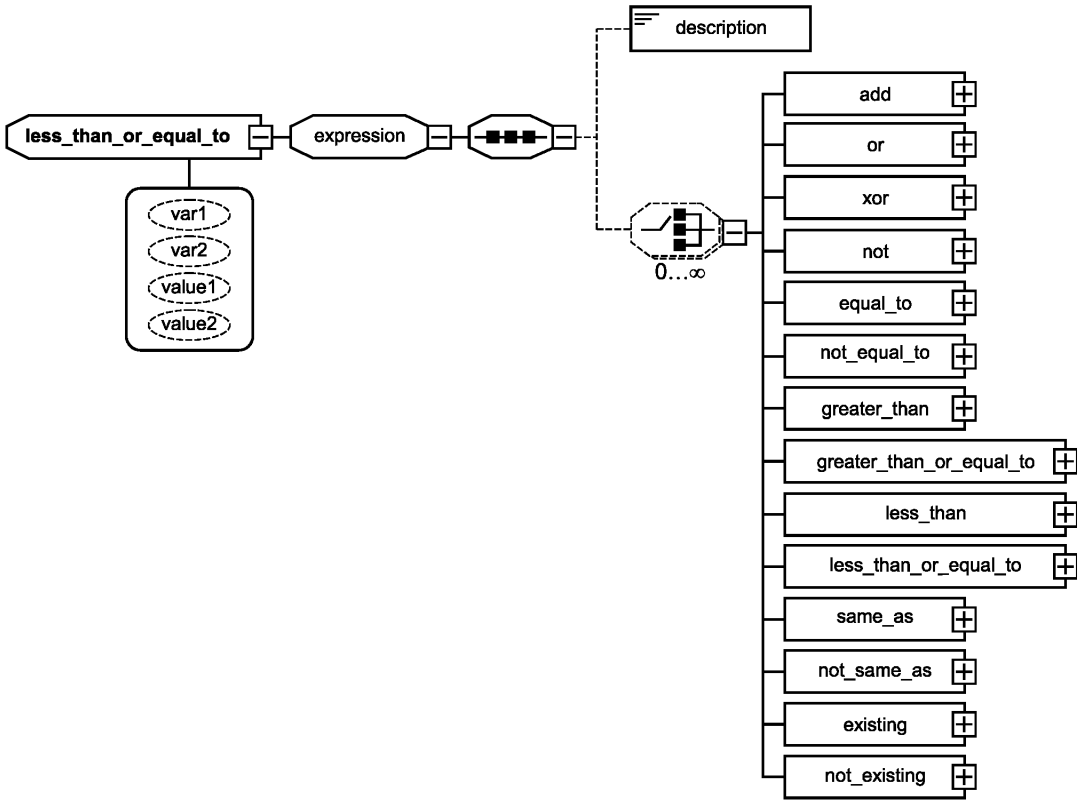


Рисунок F.29 — Элемент **<less_than_or_equal_to>**

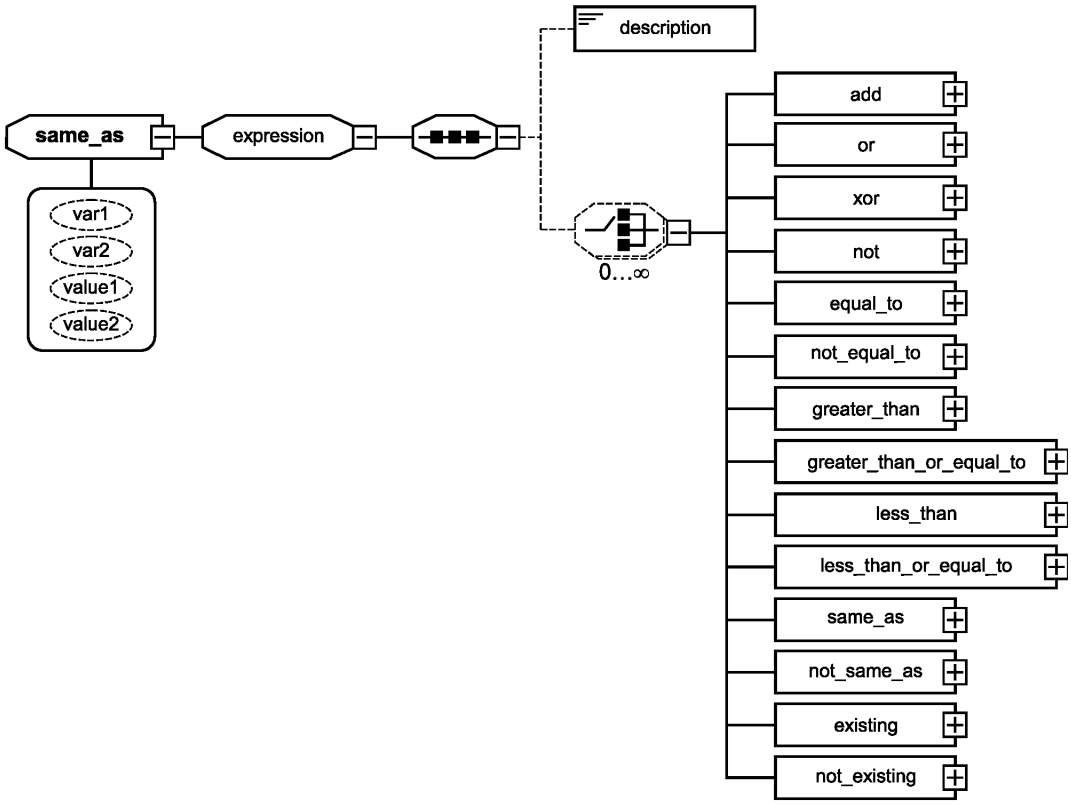


Рисунок F.30 — Элемент **<same_as>**

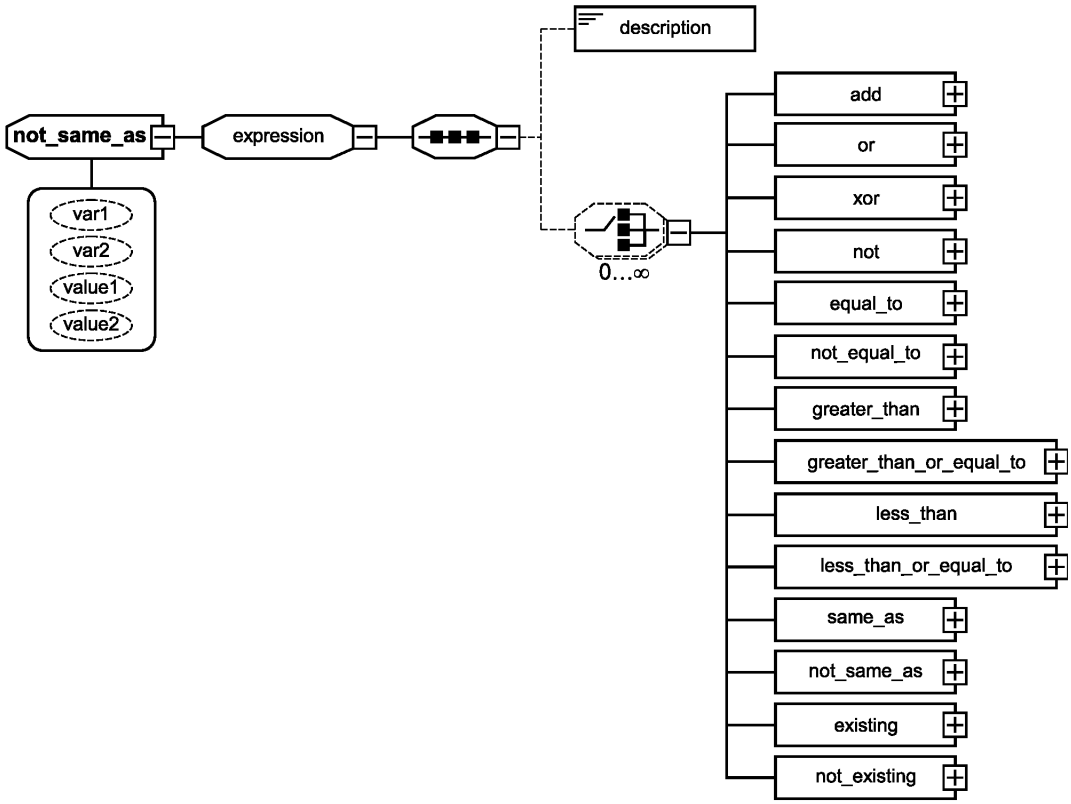


Рисунок F.31 — Элемент **<not_same_as>**

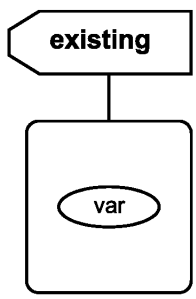


Рисунок F.32 —
Элемент **<existing>**

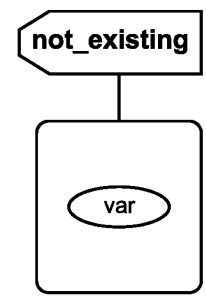


Рисунок F.33 —
Элемент **<not_existing>**

Приложение ДА
(справочное)

Сведения о соответствии ссылочных международных стандартов ссылочным национальным стандартам Российской Федерации

Сведения о соответствии ссылочного международного стандарта ссылочному национальному стандарту Российской Федерации приведены в таблице ДА.1.

Таблица ДА.1

Обозначение ссылочного международного стандарта	Степень соответствия	Обозначение и наименование соответствующего национального стандарта
ИСО/МЭК 19784-1—2006	IDT	ГОСТ Р 19784-1—2007 (ИСО/МЭК 19784-1—2006) «Спецификация биометрического программного интерфейса»

Примечание — В настоящей таблице использовано следующее условное обозначение степени соответствия стандарта.

- IDT — идентичный стандарт.

Библиография

[1] ISO/IEC Guide 2:2004 Standardization and related activities — General vocabulary

[2] ISO/IEC Guide 7:1994 Guidelines for drafting of standards suitable for use for conformity assessment

[3] ISO/IEC Guide 23:1982 Methods of indicating conformity with standards for third-party certification systems

[4] ISO/IEC Guide 27:1983 Guidelines for corrective action to be taken by a certification body in the event of misuse of its mark of conformity

[5] ISO/IEC Guide 28:2004 Conformity assessment — Guidance on a third-party certification system for products

[6] ISO/IEC 17025:2005 General requirements for the competence of testing and calibration laboratories

[7] ISO/IEC 17030:2003 Conformity assessment — General requirements for third-party marks of conformity

[8] ISO/IEC 17011:2004 Conformity assessment — General requirements for accreditation bodies accrediting conformity assessment bodies

[9] ISO/IEC Guide 65:1996 General requirements for bodies operating product certification systems

[10] ISO/IEC Guide 68:2002 Arrangements for the recognition and acceptance of conformity assessment results

[11] ISO/IEC 9646 (all parts) Information technology — Open Systems Interconnection — Conformity testing methodology and framework

[12] ISO/IEC TR 10183-1:1993 Information technology — Text and office systems — Office Document Architecture (ODA) and interchange format — Technical Report on ISO 8613 implementation testing — Part 1: Testing methodology

[13] ISO/IEC 10641:1993 Information technology — Computer graphics and image processing — Conformity testing of implementations of graphics standards

[14] ISO/IEC 13210:1999 Information technology — Requirements and Guidelines for Test Methods Specifications and Test Method Implementations for Measuring Conformance to POSIX Standards

[15] ISO/IEC 18009:1999 Information technology — Programming languages — Ada: Conformity assessment of a language processor

[16] ISO/IEC 19785-1 Information technology — Common Biometric Exchange Formats Framework — Part 1: Data element specification

[17] HOGAN, M.D. Overview of Conformity Assessment, NIST ITL, 2003

[18] JENSEN, M.W. and LEATHRUM, J.F. Conformance Test Design Methodology and Assertion Driven Test Case Generator for the Conformance Testing of Interface Standards, Clemson University, 1991

[19] GRAY, M., GOLDFINE, A., ROSENTHAL, L. and CARNAHAN, L. Conformity Testing, NIST

[20] ROSENTHAL, L., SKALL, M. and CARNAHAN, L. Conformity Testing and Certification Framework, NIST, 2001

[21] CARNAHAN, L., ROSENTHAL, L. and SKALL, M. Conformity Testing and Certification Model for Software Specifications, NIST

[22] STEWART, S.L. NISTIR 5679, Roadmap for the Computer Integrated Manufacturing Application Framework, NIST, 1995

[23] ROSENBLUM, D.S., A Practical Approach to Programming with Assertions, IEEE Transactions on Software Engineering, Vol. 21, No. 1, 1995

[24] OSTERWEIL, Leon & Lori A. Clarke, A Proposed Testing and Research Analysis Initiative, IEEE Software, September 1992

[25] VOAS, J. How Assertions Can Increase Test Effectiveness, IEEE Software, 1997

[26] FONG, E. Conformance Test Framework for Government Smart Card, NIST ITL/SDCTC

[27] VOAS, J., SCHATZ, M. and SCHMIDT, M. The Testability-Based Assertion Placement Tool For Object-Oriented Software, NIST GCR 98-735

[28] KIM J. An Evaluation Model for Conformance and Security on Biometric Systems, Inha University, Ph.D Thesis, August 2005

УДК 004.93'1:006.89 ОКС 35.040 П85

Ключевые слова: автоматическая идентификация, биометрическая идентификация, испытания на соответствие биометрическому программному интерфейсу

Редактор *Т. А. Леонова*
Технический редактор *Н. С. Гришанова*
Корректор *Е. Ю. Митрофанова*
Компьютерная верстка *А. П. Финогеновой*

Сдано в набор 16.12.2010. Подписано в печать 18.03.2011. Формат $60\times84^1/_8$. Бумага офсетная. Гарнитура Ариал. Печать офсетная. Усл. печ. л. 26,97. Уч.-изд. л. 30,00. Тираж 109 экз. Зак. 1733.

ФГУП «СТАНДАРТИНФОРМ», 123995 Москва, Гранатный пер., 4.
gost.e-ecolog.ru gost.e-ecolog.ru
Набрано и отпечатано в Калужской типографии стандартов, 248021 Калуга, ул. Московская, 256.

gost.e-ecolog.ru