

ЦЕНТРАЛЬНЫЙ БАНК РОССИЙСКОЙ ФЕДЕРАЦИИ
(БАНК РОССИИ)

УТВЕРЖДЕН
ВАМБ.00143-06-ЛУ

**«СИГНАТУРА-СЕРТИФИКАТ L» ВЕРСИЯ 6
ПРИКЛАДНОЙ ПРОГРАММНЫЙ ИНТЕРФЕЙС**

**БИБЛИОТЕКА
ДЛЯ ЯЗЫКА JAVA**

Руководство программиста

ВАМБ.00143-06 33 02

2023

Аннотация

Данный документ содержит описание библиотеки прикладного программного интерфейса (библиотеки) для языка Java к функциям сервисов Центра Сертификации (ЦС) и Центра Регистрации (ЦР) программного комплекса (ПК) ВАМБ.00128-06 «"Сигнатура-сертификат L" версия 6» (далее по тексту — ПК «Сигнатура-сертификат L»), а также рекомендации по встраиванию и использованию данной библиотеки в прикладное программное обеспечение (ПО).

Документ предназначен для разработчиков прикладных программ (внешних по отношению к ПК «Сигнатура-сертификат L») как руководство по программированию с использованием библиотеки работы с сервисами Удостоверяющего Центра (УЦ).

При встраивании библиотеки предполагается, что программист имеет знания о существующей архитектуре системы сертификатов открытых ключей, используемых рекомендациях и стандартах.

Документ разработан специалистами ООО «Валидата».

Содержание

1	БИБЛИОТЕКА ПРИКЛАДНОГО ПРОГРАММНОГО ИНТЕРФЕЙСА	4
1.1	Назначение библиотеки	4
1.2	Характеристики библиотеки	4
1.2.1	Описание формата XML модифицирующего шаблона	5
1.2.2	Описание формата XML запроса на отзыв	5
1.3	Использование библиотеки	5
1.3.1	Условия, необходимые для использования библиотеки	5
1.3.2	Состав библиотеки	6
1.4	Описание библиотеки	7
1.4.1	Типы	7
1.4.2	Использование классов	8
1.4.3	Кодирование и декодирование данных	9
1.4.4	Константы	10
1.4.5	Общие классы	11
1.4.6	Классы параметров поиска объектов	11
1.5	Описание функций получения описаний ошибок	12
1.6	Описание функций ЦР	12
1.6.1	Функции инициализации и деинициализации	12
1.6.2	Функции проверки соединения	12
1.6.3	Функции обработки запросов на выпуск сертификата	13
1.6.4	Функции обработки запросов на отзыв сертификата	14
1.6.5	Функции получения объектов из справочника по указанному критерию поиска	14
1.7	Описание функций ЦР для вызова функций ЦС	15
1.7.1	Функции вызова ЦС для выпуска сертификата	15
1.7.2	Функции вызова ЦС для отзыва сертификата	16
1.8	Описание функций ЦС	16
1.8.1	Функции инициализации и деинициализации	16
1.8.2	Функции проверки соединения	17
1.8.3	Функции выпуска сертификата	17
1.8.4	Функции отзыва сертификата	18
2	ОПИСАНИЕ ОШИБОЧНЫХ СИТУАЦИЙ	19
	ПЕРЕЧЕНЬ СОКРАЩЕНИЙ	24

1 БИБЛИОТЕКА ПРИКЛАДНОГО ПРОГРАММНОГО ИНТЕРФЕЙСА

1.1 Назначение библиотеки

Библиотека прикладного программного интерфейса предназначена для предоставления программного интерфейса, выполняемого с помощью Java JRE (среда выполнения) версия 1.8 или более новые к функциям ЦС и ЦР, входящих в состав ПК ВАМБ.00128-06 «Сигнатура-сертификат L» версия 6» (далее по тексту библиотека).

Библиотека обеспечивает удаленный доступ к функциям ЦР и ЦС обработки запросов на выпуск и отзыв сертификатов. Удаленный доступ к функциям осуществляется посредством использования протокола DCE-RPC поверх протокола ТСР/ІР.

Библиотека обеспечивает доступ к следующим функциям ЦР:

- проверка соединения с ЦР;
- обработка запроса на выпуск сертификата с выработкой запроса для ЦС;
- обработка запроса на выпуск сертификата с модифицирующим шаблоном с выработкой запроса для ЦС;
- обработка выпущенного ЦС сертификата;
- обработка запроса на отзыв сертификата с выработкой запроса для ЦС;
- получение объектов из указанного справочника ЦР по критерию поиска;
- проверка соединения ЦР с ЦС;
- вызов ЦС для выпуска сертификата по запросу на выпуск;
- вызов ЦС для выпуска сертификата по запросу на выпуск с модифицирующим шаблоном;
- вызов ЦС для обработки запроса на отзыв сертификата;

Библиотека обеспечивает доступ к следующим функциям ЦС:

- проверка соединения с ЦС;
- выпуск сертификата по запросу на выпуск;
- выпуск сертификата по запросу на выпуск с модифицирующим шаблоном;
- обработка запроса на отзыв сертификата;

1.2 Характеристики библиотеки

Запрос на выпуск сертификата передаётся в формате PKCS#10. Модифицирующий шаблон передаётся в формате XML. Запрос на отзыв сертификата для ЦС может быть сформирован по запросу на отзыв в формате XML.

Форматы сертификатов ключей проверки ЭП и/или открытых ключей шифрования, списков аннулированных сертификатов (САС) и PKCS#10 запросов на получение сертификатов, формируемых и поддерживаемых библиотекой, соответствуют Рекомендациям по стандартизации Р 1323565.1.023-2022 «Использование алгоритмов ГОСТ Р 34.10-2012, ГОСТ Р 34.11-2012 в сертификате, списке

аннулированных сертификатов (CRL) и запросе на сертификат PKCS#10 инфраструктуры открытых ключей X.509», которые в свою очередь соответствуют одноименной спецификации Технического комитета № 26.

Библиотека предназначена для работы в операционных системах (ОС), указанных в документе ВАНБ.00143-06 30 01 «“Сигнатура-сертификат L” версия 6. Прикладной программный интерфейс. Формуляр».

1.2.1 Описание формата XML модифицирующего шаблона

Формат XML модифицирующего шаблона совпадает с форматом XML запроса на сертификат, описание которого приведено в документе ВАНБ.00126-06 33 01 «“Сигнатура-клиент L” версия 6. Руководство программиста».

1.2.2 Описание формата XML запроса на отзыв

XML запрос на аннулирование (отзыв) представляет собой текстовый XML документ в кодировке UTF-8, содержащий информацию, достаточную для аннулирования сертификата.

Ниже приведен пример XML запроса на аннулирование без необходимой декларации XML документа (без XML Declaration):

```
<pkiUser>
  <Certificate>000102030405060708090a0b0c0d0e0f</Certificate>
  <ReasonCode>1</ReasonCode>
  <InvalidDate>1673513400</InvalidDate>
</pkiUser>
```

Аннулируемый сертификат содержится в элементе pkiUser -> Certificate. В этом элементе должна присутствовать шестнадцатеричная текстовая строка (hex-string) байтов сертификата в DER-кодировке.

Причина аннулирования содержится в элементе (опциональном) pkiUser -> ReasonCode и представляет собой число, описанное в документе ВАНБ.00126-06 33 01 «“Сигнатура-клиент L” версия 6. Руководство программиста». Если данный элемент не задан, причина аннулирования не задается.

Момент времени аннулирования содержится в элементе (опциональном) pkiUser -> InvalidDate и представляет собой время в секундах, прошедшее с 00:00 01.01.1970 UTC. Если данный элемент не задан, используется текущий момент времени.

1.3 Использование библиотеки

В данном документе дано описание методов и классов с указанием соответствующих констант, структур и функций из библиотеки.

1.3.1 Условия, необходимые для использования библиотеки

При использовании библиотеки необходимо соблюдать следующие условия:

- библиотека предназначена для использования в приложениях, написанных на языке программирования Java и собираемых с помощью Java SDK (средство разработки) версия 1.8 или более новые;
- перед началом использования библиотеки для обращения к функциям ЦР и ЦС необходимо запустить сервисы ЦР и ЦС;

1.3.2 Состав библиотеки

В состав библиотеки входят, в том числе, следующие файлы (X.X.X в названии - версия библиотеки, пути указаны относительно каталога установки):

- **lib/CARASVCLib-X.X.X.jar** - архив, содержащий собственно библиотеку;
- **lib/jarapac-X.X.X.jar** - архив, содержащий библиотеку Java DCE-RPC (вместе с транспортной частью);
- **lib/jcifs-1.1.2.jar** - архив, содержащий библиотеку Java реализации протоколов CIFS/SMB;
- **lib/CARASVCTest-X.X.X.jar** - командная тестовая утилита для тестирования вызова функций библиотеки;
- **bin/CARASVCTest.cmd** - командная процедура, позволяющая запускать тестовую утилиту в ОС Windows;
- **bin/CARASVCTest** - командная процедура, позволяющая запускать тестовую утилиту в ОС zLinux и в ОС Linux.

Для корректного функционирования библиотеки необходимо добавить полные пути ко всем вышеперечисленным архивам в хранилище CLASSPATH, используемое Java JRE и/или IBM WebSphere Application Server.

Поскольку архивы lib/CARASVCLib-X.X.X.jar и lib/jarapac-X.X.X.jar используют классы (Java.io.File и другие), осуществляющие доступ к файлам, свойствам и TCP/IP сокетам, при использовании Java 2 Security Manager в IBM WebSphere Application Server необходимо внести данные архивы в список разрешенных к загрузке посредством редактирования двух конфигурационных файлов:

1. Файл политик библиотек IBM WebSphere Application Server `${install_root}/config/cells/${cell_name}/nodes/${node_name}/library.policy`:

```
grant {
    permission java.net.SocketPermission "${regServiceHost}:1434",
    "connect, resolve";
    permission java.net.SocketPermission "${certServiceHost}:1333",
    "connect, resolve";
    permission java.io.FilePermission "${validataRoot}/lib/jarapac-
-X.X.X.jar", "read";
    permission java.io.FilePermission "\\${validataTemp}\\-", "read";
    permission java.util.PropertyPermission "*", "read, write";
};
```

2. Файл политик приложения IBM WebSphere Application Server (имя файла зависит от имени приложения,

В нашем случае - CARASVCTest-X.X.X.ear) `${install_root}/config/cells/${cell_name}/applications/CARASVCTest-X.X.X.ear/deployments/CARASVCTest-X.X.X/META-INF/was.policy`:

```
grant codeBase "file:${application}" {
    permission java.io.FilePermission "${validataRoot}/lib/
jarapac-X.X.X.jar", "read";
    permission java.io.FilePermission "\\${validataTemp}\\-", "
read";
    permission java.util.PropertyPermission "*", "read, write";
};
```

Переменные `validataRoot` (путь к каталогу установки библиотеки, например, `/opt/Validata/Java`), `regServiceHost/certServiceHost` (имя узла где выполняются сервис ЦР/ЦС соответственно) и `validataTemp` (путь к каталогу с рабочими файлами) можно определить с помощью Administrative Console (Консоли управления), находящейся в *Servers->Application Servers->\${server_name}->Java and Process Management->Process Definition->Java Virtual Machine->Custom Properties*. Заданные при выполнении настройки значения будут сохранены в конфигурационном файле сервера IBM WebSphere Application Server `${install_root}/config/cells/${cell_name}/nodes/${node_name}/servers/${server_name}/server.xml`.

Данные действия необходимо повторить на каждом используемом сервере. Обратите внимание, что разрешения `java.net.SocketPermission` и `java.io.FilePermission` должны быть указаны для работы приложения (в нашем случае - CARASVCTest-X.X.X.ear), тогда как разрешения `java.util.PropertyPermission` указаны для того, чтобы избежать записи полученных исключительных ситуаций в файл протоколов.

1.4 Описание библиотеки

1.4.1 Типы

При использовании библиотеки необходимо обратить внимание на следующее:

- при использовании функций библиотеки возникновение ошибочной ситуации может быть показано либо с помощью ненулевого кода возврата, выдаваемого функциями по их завершении, либо при возникновении исключительной ситуации (Java Exception), выбрасываемой во время выполнения функций;
- тип кода возврата, выдаваемого функциями библиотеки после их завершения, соответствует встроенному типу `int` языка программирования Java. Присутствие ненулевого кода возврата обычно обозначает ошибку, произошедшую на ЦР или ЦС во время выполнения запрошенной функции, и наиболее часто возникающую при передаче неправильных параметров в запрошенную функцию;
- тип исключительной ситуации (Java Exception) соответствует стандартному типу `IOException` языка программирования Java. Выбрасывание исключительной ситуации обычно означает некорректную работу протоколов DCE-RPC и/или TCP/IP, и требует проверки наличия и работоспособности подсоединения ком-

пьютеров сервера прикладного ПО и ЦР/ЦС посредством сегмента локальной вычислительной сети. При получении ошибки в виде исключительной ситуации следует прекратить использование существующего объекта (контекста) библиотеки и создать вместо него новый;

- тип всех строковых данных, используемых в функциях библиотеки, соответствует стандартному типу *String* языка программирования Java. Следует обратить внимание на то, что ЦР/ЦС всегда принимает и возвращает строковые данные, содержащие символы русского языка, в кодировке Microsoft Windows 1251 (Code Page 1251), которые библиотека получает из или преобразует во внутреннее представление типа *String*;

- тип всех констант-битовых масок, используемых в функциях библиотеки, соответствует встроенному типу *int* языка программирования Java.

1.4.2 Использование классов

Поскольку требования языка программирования Java диктуют использование классов (Java Classes), все предоставляемые библиотекой типы, функции и константы доступны посредством использования основного класса библиотеки *RASvcLibrary* для ЦР и *CASvcLibrary* для ЦС, находящихся в пакете (Java Package) *CARASVCLib*. Необходимо обратить внимание, что все составные типы, используемые в функциях библиотеки, также организованы как подклассы основных классов библиотеки *RASvcLibrary* и *CASvcLibrary*. Перед началом использования библиотеки необходимо создать объект (Java Class Instance) соответствующего основного класса библиотеки, используя предоставленный для этого конструктор класса (Java Class Constructor) *RASvcLibrary(String serverName, Properties properties)* или *CASvcLibrary(String serverName, Properties properties)*. В качестве параметра *serverName* (Имя Сервера) используется строка формата <DNS Name/IP Address>[<TCP Port>]. Параметр *properties* (значение которого может быть *null*) может содержать ниже следующие конфигурационные значения (логические значения передаются как строки "true" или "false", а числовые значения как строки вида "12345"):

- *CARASVCLib.bEnableAPIDebug* (логическое, по умолчанию == *false*) - значение *true* включает режим выдачи отладочных сообщений при вызове функций библиотеки. Не рекомендуется использовать значение *true* при загрузке ввиду большого объема выдаваемых отладочных сообщений;

- *CARASVCLib.sDebuggingOutput* (строковое, по умолчанию == "") - имя файла, в конец которого будут дописываться отладочные сообщения, выдаваемые библиотекой. Если значение равно "", то отладочные сообщения записываются в стандартный поток вывода (*System.out*);

- *CARASVCLib.iSocketMaximumCalls* (числовое, ≥ 0 , по умолчанию == 0) - значение, указывающее максимальное количество DCE-RPC вызовов, после выполнения которых библиотека автоматически выполняет переподсоединение к серверу. Если значение равно 0, то переподсоединение не производится;

- *CARASVCLib.iSocketInactivityInterval* (числовое, ≥ 0 , по умолчанию == 0) - значение (в секундах) максимального интервала между DCE-RPC вызовами, после истечения которого библиотека автоматически выполняет переподсоединение к серверу. Если значение равно 0, то переподсоединение не производится;

- *rpc.ncacn_ip_tcp.bEnableSocketIODebug* (логическое, по умолчанию == *false*) - значение *true* включает режим выдачи отладочных сообщений при выполнении DCE-RPC вызовов. Не рекомендуется использовать значение *true* при нагрузке ввиду большого объема выдаваемых отладочных сообщений;

- *rpc.ncacn_ip_tcp.bEnableSocketKeepAlive* (логическое, по умолчанию == *true*) - значение *true* включает режим TCP KeepAlive (периодическая посылка данных для контроля состояния) подключения к серверу;

- *rpc.ncacn_ip_tcp.iSocketIOTimeout* (числовое, ≥ 0 , по умолчанию == *0*) - значение (в секундах) максимального времени выполнения операции TCP ввода-вывода (подсоединение, чтение и запись данных), после истечения которого будет выброшена исключительная ситуация *SocketTimeoutException*. Если значение равно *0*, то установка максимального времени выполнения не производится.

Например, чтобы подсоединиться к ЦР, находящемуся на узле с именем *garpc* и ожидающему запросов на подсоединение к TCP порту 1333, необходимо создать объект основного класса библиотеки следующим образом: *CARASVCLib.RASvcLibrary raLib = new CARASVCLib.RASvcLibrary("rarpc[1333]", null)*. При создании объекта основного класса библиотеки инициализируется контекст (внутри данного объекта), полностью описывающий данное подключение к ЦР. При необходимости использования нескольких подключений к ЦР одновременно, следует создать требуемое число объектов (по одному на подключение) основного класса библиотеки. При необходимости использования библиотеки из нескольких потоков приложения необходимо обеспечить эксклюзивное использование каждого контекста конкретным потоком, т.е. не допускается одновременное использование данного контекста несколькими потоками приложения.

1.4.3 Кодирование и декодирование данных

Для обмена данными с ЦР/ЦС используется протокол DCE-RPC поверх протокола TCP/IP. Все входные данные, передаваемые в функции и возвращаемые из функций библиотеки, кодируются в соответствии со спецификацией *Distributed Computing Environment* версия 1.2 (DCE версия 1.2). При заполнении входных данных необходимо выполнить следующее условие, связанное с особенностями процедуры кодирования: все ссылки на классы (и все ссылки из них и всех их подклассов), передаваемые как входные данные в функции библиотеки, должны быть не равны *null* и должны ссылаться на существующие объекты соответствующих классов - если в описании функции не указано иначе. Исключения составляют только ссылки на массивы и объекты стандартного типа *String* языка Java. Невыполнение этого требования может привести к выбрасыванию исключительной ситуации *java.lang.NullPointerException*. Например, при заполнении параметра *pREVREQBlock* (класса *CASvcLibrary.mem_blk_t*), используемого в функции *CASvcLibrary.CA_ProcessREVREQBlock*, ссылка на объект класса должна быть не равна *null*. В тоже время, ссылка на массив *buf* из данного объекта может быть равна *null*. В последнем случае исключение не будет сгенерировано, но будет возвращён код ошибки, указывающий на неверный формат запроса на отзыв.

1.4.4 Константы

Битовые маски значения параметра *fXmlFlags* для функций обработки запроса с модифицирующим шаблоном в формате XML.

```
int CARALib.CARALIB_XML_CA_CERTIFICATE = 1;  
int CARALib.CARALIB_XML_RA_CERTIFICATE = 1 « 1;  
int CARALib.CARALIB_XML_CREATE_PRIVATE_KEY = 1 « 2;  
int CARALib.CARALIB_XML_AGENTS_DISALLOWED = 1 « 3;  
int CARALib.CARALIB_XML_NO_KEY_IDENTIFIER = 1 « 4;  
int CARALib.CARALIB_XML_SWAP_SUBJECT_NAME = 1 « 5;
```

Значения параметра *fStoreType* функции получения списка объектов *RASvcLibrary.RA_FindObjects*.

```
int CARALib.CARASEARCH_STORE_PSE = 1;  
int CARALib.CARASEARCH_STORE_LOCAL = 2;  
int CARALib.CARASEARCH_STORE_CERT = 3;  
int CARALib.CARASEARCH_STORE_REG = 4;
```

Значения параметра *fObjectType* функции получения списка объектов *RASvcLibrary.RA_FindObjects*.

```
int CARALib.CARACOMMON_OBJECT_X509 = 1;  
int CARALib.CARACOMMON_OBJECT_CRL = 2;  
int CARALib.CARACOMMON_OBJECT_REQ = 3;  
int CARALib.CARACOMMON_OBJECT_REVREQ = 4;  
int CARALib.CARACOMMON_OBJECT_XMLREQ = 5;
```

Значения параметра *fValueType* функции получения списка объектов *RASvcLibrary.RA_FindObjects*.

```
int CARALib.CARASEARCH_VALUE_ANY_OBJECT = 1;  
int CARALib.CARASEARCH_VALUE_LAST_RECORDS = 2;  
int CARALib.CARASEARCH_VALUE_SUBJECT_STR = 3;  
int CARALib.CARASEARCH_VALUE_SUBJECT_SUBSTR = 4;  
int CARALib.CARASEARCH_VALUE_ISSUER_STR = 5;  
int CARALib.CARASEARCH_VALUE_ISSUER_SUBSTR = 6;  
int CARALib.CARASEARCH_VALUE_ISSUER_AND_SERIAL = 7;  
int CARALib.CARASEARCH_VALUE_KEY_IDENTIFIER = 8;  
int CARALib.CARASEARCH_VALUE_SERIAL_NUMBER = 9;  
int CARALib.CARASEARCH_VALUE_SUBJECT_KEYID = 10;  
int CARALib.CARASEARCH_VALUE_ISSUER_KEYID = 11;  
int CARALib.CARASEARCH_VALUE_ALTNAME_EMAIL = 12;  
int CARALib.CARASEARCH_VALUE_ALTNAME_COMPANY = 13;  
int CARALib.CARASEARCH_VALUE_ALTNAME_SURNAME = 14;  
int CARALib.CARASEARCH_VALUE_EXPIRES_WITHIN = 15;  
int CARALib.CARASEARCH_VALUE_KEY_EXPIRES_WITHIN = 16;  
int CARALib.CARASEARCH_VALUE_VALID_FROM = 17;  
int CARALib.CARASEARCH_VALUE_VALID_TO = 18;  
int CARALib.CARASEARCH_VALUE_KEY_VALID_FROM = 19;  
int CARALib.CARASEARCH_VALUE_KEY_VALID_TO = 20;  
int CARALib.CARASEARCH_VALUE_VALUE_SIGNING_TIME = 21;
```

Значения элементов параметра-массива *pFltParse* функции получения списка объектов *RASvcLibrary.RA_FindObjects*.

```
int CARALib.CARAU_PARSE_NAME_RDN_ORGANIZATIONAL_UNIT = 13;
int CARALib.CARAU_PARSE_NAME_RDN_ORGANIZATIONAL_NAME = 14;
int CARALib.CARAU_PARSE_NAME_RDN_STREET_ADDRESS = 15;
int CARALib.CARAU_PARSE_NAME_RDN_LOCALITY_NAME = 16;
int CARALib.CARAU_PARSE_NAME_RDN_STATE_PROVINCE_NAME = 17;
int CARALib.CARAU_PARSE_NAME_RDN_COUNTRY_NAME = 18;
int CARALib.CARAU_PARSE_NAME_RDN_DOMAIN_COMPONENT = 19;
```

Значения поля *type* класса *RASvcLibrary.find_object_param_t*.

```
int RASvcLibrary.FOP_IAS = 0;
int RASvcLibrary.FOP_LONG = 1;
int RASvcLibrary.FOP_STRING = 2;
int RASvcLibrary.FOP_INTERVAL = 3;
```

1.4.5 Общие классы

Классы *RASvcLibrary.mem_blk_t* и *CASvcLibrary.mem_blk_t*

Блок данных.

Состав:

– int **len**

Размер буфера (не может быть более 2 Гбайт).

– byte[] **buf**

Буфер (массив с данными).

Классы *RASvcLibrary.mem_blk_array_t* и *CASvcLibrary.mem_blk_array_t*

Массив блоков данных.

Состав:

– int **len**

Количество блоков данных в массиве.

– mem_blk_t[] **buf**

Буфер (массив с блоками данных).

1.4.6 Классы параметров поиска объектов

Класс *RASvcLibrary.issuer_and_serial_t*

Издатель и серийный номер сертификата.

Состав:

– String **issuer**

Издатель (в виде строки Distinguished Name (DN) LDAP, например, CN=Users,DC=x509,DC=ru).

– String **serial**

Серийный номер сертификата (в виде hex-строки, например, 40:00:00:01).

Класс *RASvcLibrary.interval_t*

Интервал времени.

Состав:

– long **not_before**

Время и дата начала интервала.

– long **not_after**

Время и дата конца интервала.

Класс RASvcLibrary.find_object_param_t

Параметры поиска объектов.

Моделирует UNION языка C.

Состав:

– int **type**

Определяет используемое поле. Значения остальных полей игнорируются.

– int **long_p**

*Используется если **type** равен FOP_LONG.*

– String **string_p**

*Используется если **type** равен FOP_STRING.*

– issuer_and_serial_t **ias_p**

*Используется если **type** равен FOP_IAS.*

– interval_t **interval_p**

*Используется если **type** равен FOP_INTERVAL.*

1.5 Описание функций получения описаний ошибок

String **CARAErrors.CARAGetErrorText** (int code)

Функция получения текстового сообщения по коду ошибки

Возвращаемые значения:

строку с текстовым описанием ошибки.

Аргументы:

– **code** (in) код ошибки;

1.6 Описание функций ЦР

1.6.1 Функции инициализации и деинициализации

void **RASvcLibrary.Detach** ()

Функция отсоединения от ЦР и уничтожения контекста

Примечание - Данную функцию всегда необходимо вызывать по окончании использования объекта основного класса библиотеки (контекста) для очистки выделенных ресурсов. После вызова данной функции повторное использование уничтоженного контекста не разрешается.

1.6.2 Функции проверки соединения

int **RASvcLibrary.RA_PingService** ()

Функция проверки соединения с сервисом ЦР

Соответствует функции RACLI_PingService.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

int **RASvcLibrary.RA_CAPingService** ()

Функция проверки соединения сервиса ЦР с сервисом ЦС

Соответствует функции RACLI_CAPingService.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

1.6.3 Функции обработки запросов на выпуск сертификата

int **RASvcLibrary.RA_REQBlockToREQBlock** (mem_blk_t pInREQBlock, mem_blk_t pOutREQBlock)

Функция обработки запроса на выпуск сертификата

Соответствует функции RACLI_REQBlockToREQBlock.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

Аргументы:

- **pInREQBlock** (in) запрос на выпуск сертификата;
- **pOutREQBlock** (out) подготовленный для ЦС запрос на выпуск сертификата;

Запрос на выпуск сертификата в формате **PKCS#10** должен быть подписан на сертификате Оператора ЦР и передаваться в контейнере **PKCS#7** с присоединенной подписью.

int **RASvcLibrary.RA_XmlBlockMergeWithREQBlock** (mem_blk_t pXmlBlock, mem_blk_t pInREQBlock, mem_blk_t pOutREQBlock, int fXmlFlags)

Функция обработки запроса на выпуск сертификата

с модифицирующим шаблоном

Соответствует функции RACLI_XmlBlockMergeWithREQBlock.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

Аргументы:

- **pXmlBlock** (in) модифицирующий шаблон в формате XML;
- **pInREQBlock** (in) запрос на выпуск сертификата;
- **pOutREQBlock** (out) подготовленный для ЦС запрос на выпуск сертификата;
- **fXmlFlags** (in) флаги обработки модифицирующего шаблона;

Запрос на выпуск сертификата в формате **PKCS#10** должен быть подписан на сертификате Оператора ЦР и передаваться в контейнере **PKCS#7** с присоединенной подписью.

int **RASvcLibrary.RA_X509BlockToX509Block** (mem_blk_t pInBlock, mem_blk_t pOutBlock)

Функция обработки полученного от ЦС блока данных

Соответствует функции RACLI_X509BlockToX509Block.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

Аргументы:

- **pInBlock** (in) полученный от ЦС блок данных, содержащий выпущенный сертификат;
- **pOutBlock** (out) сертификат для выдачи пользователю.

1.6.4 Функции обработки запросов на отзыв сертификата

int **RASvcLibrary.RA_REVREQBlockToREVREQBlock** (mem_blk_t pInREVREQBlock, mem_blk_t pOutREVREQBlock)

Функция обработки запроса на отзыв сертификата

Соответствует функции RACLI_REVREQBlockToREVREQBlock.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

Аргументы:

- **pInREVREQBlock** (in) запрос на отзыв сертификата;
- **pOutREVREQBlock** (out) подготовленный для ЦС запрос на отзыв сертификата.

Запрос на отзыв сертификата должен быть подписан на сертификате Оператора ЦР и передаваться в контейнере **PKCS#7** с присоединенной подписью.

int **RASvcLibrary.RA_XmlBlockToREVREQBlock** (mem_blk_t pXmlBlock, mem_blk_t pREVREQBlock, int fXmlFlags)

Функция обработки запроса на отзыв сертификата в формате XML

Соответствует функции RACLI_XmlBlockToREVREQBlock.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

Аргументы:

- **pXmlBlock** (in) запрос на отзыв сертификата в формате XML;
- **pREVREQBlock** (out) подготовленный для ЦС запрос на отзыв сертификата;
- **fXmlFlags** (in) флаги обработки запроса в формате XML;

Запрос на отзыв сертификата в формате XML должен быть подписан на сертификате Оператора ЦР и передаваться в контейнере **PKCS#7** с присоединенной подписью.

1.6.5 Функции получения объектов из справочника по указанному критерию поиска

int **RASvcLibrary.RA_FindObjects** (int fStoreType, int fObjectType, int fValueType, find_object_param_t pFindParam, int[] pFiltParse, String[] ppszFiltValue, int[] piFiltOffset, mem_blk_array_t ppObjects)

Функция получения объектов системы управления сертификатами (СУС) по указанному критерию поиска

Соответствует функции RACLI_FindObjects.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

Аргументы:

- **fStoreType** (in) справочник, где будет выполняться поиск объектов. Одна из констант **CARALib.CARASEARCH_STORE_XXX**;
- **fObjectType** (in) тип объекта. Одна из констант **CARALib.CARACOMMON_OBJECT_XXX**;
- **fValueType** (in) критерий поиска объектов. Одна из констант **CARALib.CARASEARCH_VALUE_XXX**;
- **pFindParam** (in) параметры поиска, в зависимости от критерия может быть *null*;
- **pFltParse** (in) дополнительный параметр поиска, массив значений констант **CARALib.CARAU_PARSE_NAME_RDN_XXX**, может быть *null*;
- **ppszFltValue** (in) дополнительный параметр поиска, массив строк, может быть *null*;
- **piFltOffset** (in) дополнительный параметр поиска, массив чисел в диапазоне 0-255, может быть *null*;
- **ppObjects** (out) найденные объекты;

Параметр **pFindParam** должен иметь значение *null* если параметр **fValueType** равен **CARALib.CARASEARCH_VALUE_ANY_OBJECT**. Иначе должен быть передан соответствующим образом заполненный объект класса **find_object_param_t**.

Параметры **pFltParse**, **ppszFltValue** и **piFltOffset** должны либо все иметь значение *null*, либо быть массивами одинаковой длины, не превышающей 255.

Для получения всех объектов справочника указанного типа следует использовать **CARALib.CARASEARCH_VALUE_ANY_OBJECT** для параметра **fObjectType** и *null* для параметров **pFindParam**, **pFltParse**, **ppszFltValue** и **piFltOffset**.

1.7 Описание функций ЦР для вызова функций ЦС

В случае настройки соединения сервиса ЦР с сервисом ЦС имеется возможность вызова отдельных функций ЦС посредством вызова соответствующих функций ЦР. При этом ЦР в свою очередь выполняет вызов функций ЦС, используя настроенное соединение и переданные параметры и возвращая полученные результаты.

1.7.1 Функции вызова ЦС для выпуска сертификата

int RASvcLibrary.RA_CAREQBlockToX509Block (mem_blk_t pREQBlock, mem_blk_t pX509Block, int iCertMonths, int iKeyMonths)

Функция обработки подготовленного для ЦС запроса на выпуск сертификата
Соответствует функции **RACLI_CAREQBlockToX509Block**.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

Аргументы:

- **pREQBlock** (in) подготовленный для ЦС запрос на выпуск сертификата;
- **pX509Block** (out) блок данных для обработки ЦР, содержащий выпущенный сертификат;

- **iCertMonths** (in) длительность действия сертификата в месяцах;
- **iKeyMonths** (in) длительность действия закрытого ключа в месяцах;

Полученный в случае успеха блок данных, содержащий выпущенный сертификат, должен быть обработан с помощью функции ЦР **RASvcLibrary.RA_X509BlockToX509Block**.

int **RASvcLibrary.RA_CAXmlBlockMergeWithREQBlockToX509Block** (mem_blk_t pXmlBlock, mem_blk_t pREQBlock, mem_blk_t pX509Block, int fXmlFlags, int iCertMonths, int iKeyMonths)

Функция обработки подготовленного для ЦС запроса

на выпуск сертификата с модифицирующим шаблоном

Соответствует функции RACLI_CAXmlBlockMergeWithREQBlockToX509Block.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

Аргументы:

- **pXmlBlock** (in) модифицирующий шаблон в формате XML;
- **pREQBlock** (in) подготовленный для ЦС запрос на выпуск сертификата;
- **pX509Block** (out) блок данных для обработки ЦР, содержащий выпущенный сертификат;
- **fXmlFlags** (in) флаги обработки модифицирующего шаблона;
- **iCertMonths** (in) длительность действия сертификата в месяцах;
- **iKeyMonths** (in) длительность действия закрытого ключа в месяцах;

Полученный в случае успеха блок данных, содержащий выпущенный сертификат, должен быть обработан с помощью функции ЦР **RASvcLibrary.RA_X509BlockToX509Block**.

1.7.2 Функции вызова ЦС для отзыва сертификата

int **RASvcLibrary.RA_CAProcessREVREQBlock** (mem_blk_t pREVREQBlock)

Функция обработки подготовленного для ЦС запроса на отзыв сертификата

Соответствует функции RACLI_CAProcessREVREQBlock.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

Аргументы:

- **pREVREQBlock** (in) подготовленный для ЦС запрос на отзыв сертификата;

1.8 Описание функций ЦС

1.8.1 Функции инициализации и деинициализации

void **CASvcLibrary.Detach** ()

Функция отсоединения от ЦС и уничтожения контекста

Примечание - Данную функцию всегда необходимо вызывать по окончании использования объекта основного класса библиотеки (контекста) для очистки выделенных ресурсов. После вызова данной функции повторное использование уничтоженного контекста не разрешается.

1.8.2 Функции проверки соединения

int **CASvcLibrary.CA_PingService** ()

Функция проверки соединения с сервисом ЦС

Соответствует функции CACLI_PingService.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

1.8.3 Функции выпуска сертификата

int **CASvcLibrary.CA_REQBlockToX509Block** (mem_blk_t pREQBlock, mem_blk_t pX509Block, int iCertMonths, int iKeyMonths)

Функция обработки подготовленного для ЦС запроса на выпуск сертификата

Соответствует функции CACLI_REQBlockToX509Block.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

Аргументы:

- **pREQBlock** (in) подготовленный для ЦС запрос на выпуск сертификата;
- **pX509Block** (out) блок данных для обработки ЦР, содержащий выпущенный сертификат;
- **iCertMonths** (in) длительность действия сертификата в месяцах;
- **iKeyMonths** (in) длительность действия закрытого ключа в месяцах;

Полученный в случае успеха блок данных, содержащий выпущенный сертификат, должен быть обработан с помощью функции ЦР **RASvcLibrary.RA_X509BlockToX509Block**.

int **CASvcLibrary.CA_XmlBlockMergeWithREQBlockToX509Block** (mem_blk_t pXmlBlock, mem_blk_t pREQBlock, mem_blk_t pX509Block, int fXmlFlags, int iCertMonths, int iKeyMonths)

Функция обработки подготовленного для ЦС запроса

на выпуск сертификата с модифицирующим шаблоном

Соответствует функции CACLI_XmlBlockMergeWithREQBlockToX509Block.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

Аргументы:

- **pXmlBlock** (in) модифицирующий шаблон в формате XML;
- **pREQBlock** (in) подготовленный для ЦС запрос на выпуск сертификата;
- **pX509Block** (out) блок данных для обработки ЦР, содержащий выпущенный сертификат;
- **fXmlFlags** (in) флаги обработки модифицирующего шаблона;
- **iCertMonths** (in) длительность действия сертификата в месяцах;
- **iKeyMonths** (in) длительность действия закрытого ключа в месяцах;

Полученный в случае успеха блок данных, содержащий выпущенный сертификат, должен быть обработан с помощью функции ЦР **RASvcLibrary.RA_X509BlockToX509Block**.

1.8.4 Функции отзыва сертификата

int **CASvcLibrary.CA_ProcessREVREQBlock** (mem_blk_t pREVREQBlock)

Функция обработки подготовленного для ЦС запроса на отзыв сертификата

Соответствует функции CACLI_ProcessREVREQBlock.

Возвращаемые значения:

0 в случае успеха или ненулевой код ошибки.

Аргументы:

– **pREVREQBlock** (in) подготовленный для ЦС запрос на отзыв сертификата;

2 ОПИСАНИЕ ОШИБОЧНЫХ СИТУАЦИЙ

Ниже (Таблица 1) приведено описание возможных ошибочных ситуаций. В левой колонке указано символьное имя ошибки и шестнадцатеричное значение ее кода, в правой колонке приведено детальное описание и причина возникновения ошибки.

Таблица 1 – Описание ошибочных ситуаций

Имя и код ошибки	Описание и причина возникновения ошибки
CARASC_E_NO_MEMORY (0xE0D20001)	Ошибка выделения памяти.
CARASC_E_INVALID_PARAM (0xE0D20002)	Недопустимое значение параметра.
CARASC_E_NOT_IMPLEMENTED (0xE0D20003)	Функция отсутствует.
CARASC_E_INTERNAL_ERROR (0xE0D20004)	Внутренняя ошибка.
CARASC_E_DELETE_FOLDER (0xE0D20005)	Ошибка удаления папки.
CARASC_E_CREATE_FOLDER (0xE0D20006)	Ошибка создания папки.
CARASC_E_LOG_INIT (0xE0D20007)	Ошибка инициализации журналирования.
CARASC_E_INVALID_FOLDER (0xE0D20008)	Недопустимое имя папки.
CARASC_E_CFG_READ (0xE0D20009)	Ошибка чтения настроек.
CARASC_E_REG_ACCESS_DENIED (0xE0D2000A)	Недостаточно прав для операции с ключом/значением реестра.
CARASC_E_CFG_WRITE (0xE0D2000B)	Ошибка записи настроек.
CARASC_E_OPERATION_ - PROHIBITED (0xE0D2000C)	Операция запрещена.
CARASC_E_INVALID_SESSION (0xE0D2000D)	Недопустимый дескриптор сессии работы с сервисом.
CARASC_E_RPC_ERROR (0xE0D2000E)	Ошибка RPC.
CARASC_E_SESSION_ - INITIALIZED (0xE0D20010)	Сессия криптопровайдера уже инициализирована.
CARASC_E_INVALID_CRYPT_ - CTX (0xE0D20011)	Недопустимый дескриптор сессии криптопровайдера.
CARASC_EMSG_AUTO_BAD_OBJ_ - TYPE (0xE0D20012)	Неподдерживаемый тип объекта для авто-обработки.
CARASC_E_AUTO_PLUGIN_ - FAILED (0xE0D20013)	Ошибка вызова подключаемого модуля экспорта.
CARASC_E_SYSTEM_ERROR (0xE0D20015)	Системная ошибка.
CARASC_E_CA_FWD_FAILED (0xE0D20016)	Ошибка вызова функции сервиса ЦС.
CARASC_E_CA_FWD_PING_ - FAILED (0xE0D20017)	Сервис ЦС недоступен.
CARASC_E_LIC_DB_ERROR (0xE0D20018)	Ошибка получения количества сертификатов.
CARASC_E_LIC_BAD (0xE0D20019)	Отсутствует валидная лицензия.

Имя и код ошибки	Описание и причина возникновения ошибки
CARASC_E_LIC_CERT_LIMIT (0xE0D2001A)	Превышен лимит сертификатов.
CARASC_E_REQ_LOG_CREATE (0xE0D2001B)	Ошибка создания таблиц журналов запросов.
CARASC_E_REQ_LOG_FIND_CERT (0xE0D2001C)	Ошибка чтения данных журнала запросов на выпуск сертификата.
CARASC_E_REQ_LOG_FIND_REV (0xE0D2001D)	Ошибка чтения данных журнала запросов на отзыв сертификата.
CARALIB_E_NO_MEMORY (0xE0C40001)	Недостаточно оперативной памяти.
CARALIB_E_INVALID_PARAM (0xE0C40002)	Передан неверный параметр.
CARALIB_E_INVALID_FLAGS (0xE0C40003)	Переданы неверные флаги.
CARALIB_E_INVALID_CONTEXT (0xE0C40004)	Неверный контекст библиотеки.
CARALIB_E_NOT_IMPLEMENTED (0xE0C40005)	Вызов данной функции не разрешен.
CARALIB_E_INTERNAL_ERROR (0xE0C40006)	Произошла внутренняя ошибка.
CARALIB_E_BUFFER_TOO_SMALL (0xE0C40007)	Размер буфера недостаточен.
CARALIB_E_DATA_MISSING (0xE0C40008)	Отсутствуют требуемые данные.
CARALIB_E_CANCELLED_BY_USER (0xE0C40009)	Операция отменена пользователем.
CARALIB_E_BOOLEAN_XML_BLOCK (0xE0C4000A)	Неверный формат блока данных XML.
CARALIB_E_SYSTEM_DATE_TIME (0xE0C4000B)	Ошибка получения системного времени.
CARALIB_E_DATE_TIME_FORMAT (0xE0C4000C)	Неверный формат даты или времени.
CARALIB_E_STRING_ENCODING (0xE0C4000D)	Неверная кодировка строковых данных.
CARALIB_E_INVALID_CONFIG (0xE0C4000E)	Неверная конфигурация библиотеки.
CARALIB_E_CSP_INITIALIZE (0xE0C4000F)	Ошибка инициализации СКЗИ.
CARALIB_E_STRING_TO_HEX (0xE0C40010)	Ошибка конвертации строки в 16-ричный формат.
CARALIB_E_GENERATE_KEYID (0xE0C40011)	Ошибка формирования идентификатора ключа ЭП.
CARALIB_E_CREATE_PRIVATE_KEY (0xE0C40012)	Ошибка формирования ключа ЭП.
CARALIB_E_LOAD_PRIVATE_KEY (0xE0C40013)	Ошибка загрузки ключа ЭП.
CARALIB_E_GET_PUBLIC_KEY (0xE0C40014)	Ошибка получения ключа проверки ЭП.
CARALIB_E_DUPLICATE_PUBLIC_KEY (0xE0C40015)	Сертификат с таким ключом проверки ЭП уже существует.
CARALIB_E_CREATE_NEW_STORE (0xE0C40016)	Ошибка создания нового хранилища.
CARALIB_E_OPEN_EXISTANT_STORE (0xE0C40017)	Ошибка открытия существующего хранилища.
CARALIB_E_ADD_STORE_SIGNER (0xE0C40018)	Ошибка вычисления ЭП хранилища.
CARALIB_E_STORE_ADD_CERT (0xE0C40019)	Ошибка добавления сертификата в хранилище.
CARALIB_E_STORE_ADD_CRL (0xE0C4001A)	Ошибка добавления САС в хранилище.

Имя и код ошибки	Описание и причина возникновения ошибки
CARALIB_E_STORE_ADD_REQ (0xE0C4001B)	Ошибка добавления запроса PKCS#10 в хранилище.
CARALIB_E_STORE_ADD_REVREQ (0xE0C4001C)	Ошибка добавления запроса на аннулирование в хранилище.
XCARALIB_E_STORE_NOT_SIGNED (0xE0C4001D)	Хранилище не подписано.
CARALIB_E_INVALID_STORE_URI (0xE0C4001E)	Неверный идентификатор хранилища.
CARALIB_E_INVALID_STORE_USAGE (0xE0C4001F)	Неверное использование хранилища.
CARALIB_E_STORE_ADD_FAILED (0xE0C40020)	Ошибка функции добавления объекта в хранилище.
CARALIB_E_STORE_MODIFY_FAILED (0xE0C40021)	Ошибка функции модификации объекта в хранилище.
CARALIB_E_STORE_FIND_FAILED (0xE0C40022)	Ошибка функции поиска объекта в хранилище.
CARALIB_E_STORE_CTRL_FAILED (0xE0C40023)	Ошибка функции управления хранилищем.
CARALIB_E_STORE_DELETE_FAILED (0xE0C40024)	Ошибка функции удаления объекта из хранилища.
CARALIB_E_STORE_EMPTY (0xE0C40025)	В хранилище отсутствуют объекты.
CARALIB_E_CERT_NOT_FOUND (0xE0C40026)	Сертификат не найден.
CARALIB_E_CRL_NOT_FOUND (0xE0C40027)	CAC не найден.
CARALIB_E_REQ_NOT_FOUND (0xE0C40028)	Запрос PKCS#10 не найден.
CARALIB_E_REVREQ_NOT_FOUND (0xE0C40029)	Запрос на аннулирование не найден.
CARALIB_E_CALCULATE_KEY (0xE0C4002A)	Ошибка вычисления ключа объекта.
CARALIB_E_KEY_NOT_FOUND (0xE0C4002B)	Контекст библиотеки был создан для проверки ЭП.
CARALIB_E_WRITE_FILE (0xE0C4002C)	Произошла ошибка при записи выходного файла.
CARALIB_E_PRINT_OBJECT (0xE0C4002D)	Произошла ошибка при печати объекта.
CARAASN_E_ASN1_CERT_ENCODE (0xE0C50001)	Ошибка кодирования сертификата X.509.
CARAASN_E_ASN1_CERT_DECODE (0xE0C50002)	Ошибка декодирования сертификата X.509.
CARAASN_E_ASN1_CRL_ENCODE (0xE0C50003)	Ошибка кодирования CAC X.509.
CARAASN_E_ASN1_CRL_DECODE (0xE0C50004)	Ошибка декодирования CAC X.509.
CARAASN_E_ASN1_REQ_ENCODE (0xE0C50005)	Ошибка кодирования запроса PKCS#10.
CARAASN_E_ASN1_REQ_DECODE (0xE0C50006)	Ошибка декодирования запроса PKCS#10.
CARAASN_E_ASN1_REVREQ_ENCODE (0xE0C50007)	Ошибка кодирования запроса на аннулирование.
CARAASN_E_ASN1_REVREQ_DECODE (0xE0C50008)	Ошибка декодирования запроса на аннулирование.

Имя и код ошибки	Описание и причина возникновения ошибки
CARAASN_E_ASN1_PKCS7_- ENCODE (0xE0C50009)	Ошибка закодирования сообщения CMS/PKCS#7.
CARAASN_E_ASN1_PKCS7_- DECODE (0xE0C5000A)	Ошибка раскодирования сообщения CMS/PKCS#7.
CARAASN_E_INVALID_PKCS7_- TYPE (0xE0C5000B)	Неверный тип сообщения CMS/PKCS#7.
CARAASN_E_ASN1_X509V3_- ENCODE (0xE0C5000C)	Ошибка закодирования расширения X.509v3.
CARAASN_E_ASN1_X509V3_- DECODE (0xE0C5000D)	Ошибка раскодирования расширения X.509v3.
CARAPKI_E_X500_NAME_CREATE (0xE0C60001)	Ошибка создания X.500 имени владельца.
CARAPKI_E_X509V3_PREV_- SUBJECT (0xE0C60002)	Ошибка расширения "Предыдущее имя владельца".
CARAPKI_E_X509V3_ALT_NAME (0xE0C60003)	Ошибка расширения "Альтернативное имя владельца".
CARAPKI_E_X509V3_KEY_USAGE (0xE0C60004)	Ошибка расширения "Использование ключа".
CARAPKI_E_X509V3_BASIC_- CONSTR (0xE0C60005)	Ошибка расширения "Базовые ограничения".
CARAPKI_E_X509V3_KEY_- USAGE_PERIOD (0xE0C60006)	Ошибка расширения "Период использования ключа".
CARAPKI_E_X509V3_EXT_KEY_- USAGE (0xE0C60007)	Ошибка расширения "Расширенное использование ключа".
CARAPKI_E_X509V3_CERT_- POLICY (0xE0C60008)	Ошибка расширения "Регламенты сертификата".
CARAPKI_E_X509V3_CUSTOM_- EXTENSION (0xE0C60009)	Ошибка расширения "Собственное расширение".
CARAPKI_E_X509V3_RA_- CERTIFICATE (0xE0C6000A)	Ошибка расширения "Сертификат Центра Регистрации".
CARAPKI_E_X509V3_PRIVATE_- KEYID (0xE0C6000B)	Ошибка расширения "Идентификатор ключа ЭП".
CARAPKI_E_X509V3_SUBJECT_- KEYID (0xE0C6000C)	Ошибка расширения "Идентификатор ключа владельца".
CARAPKI_E_X509V3_- AUTHORITY_KEYID (0xE0C6000D)	Ошибка расширения "Идентификатор ключа издателя".
CARAPKI_E_KEY_NOT_YET_- VALID (0xE0C6000E)	Ключ ЭП еще не действителен.
CARAPKI_E_KEY_HAS_EXPIRED (0xE0C6000F)	Ключ ЭП уже истек.
CARAPKI_E_KEY_USAGE_PERIOD (0xE0C60010)	Ошибка периода использования ключа.
CARAPKI_E_CERT_NOT_YET_- VALID (0xE0C60011)	Сертификат еще не действителен.
CARAPKI_E_CERT_HAS_EXPIRED (0xE0C60012)	Сертификат уже истек.
CARAPKI_E_CERT_USAGE_- PERIOD (0xE0C60013)	Ошибка периода использования сертификата.

Имя и код ошибки	Описание и причина возникновения ошибки
CARAPKI_E_CERT_IS_DAMAGED (0xE0C60014)	Сертификат поврежден или искажен.
CARAPKI_E_CERT_IS_MISSING (0xE0C60015)	Отсутствует сертификат издателя.
CARAPKI_W_CERT_IS_ON_HOLD (0xA0C60016)	Действие сертификата приостановлено.
CARAPKI_E_CERT_IS_UNTRUSTED (0xE0C60017)	Сертификат не является доверенным.
CARAPKI_E_CRL_NOT_YET_VALID (0xE0C60018)	CAC еще не действителен.
CARAPKI_E_CRL_HAS_EXPIRED (0xE0C60019)	CAC уже истек.
CARAPKI_E_CRL_IS_DAMAGED (0xE0C6001A)	CAC поврежден или искажен.
CARAPKI_E_CRL_IS_MISSING (0xE0C6001B)	Отсутствует CAC издателя.
CARAPKI_E_INVALID_USAGE (0xE0C6001C)	Неверное использование ключа или сертификата.
CARAPKI_E_INVALID_SIGNATURE (0xE0C6001D)	Произошла ошибка проверки ЭП.
CARAPKI_E_INVALID_SIGNERS_COUNT (0xE0C6001E)	Неверное количество подписантов.
CARAPKI_E_CHAIN_TOO_LONG (0xE0C6001F)	Цепочка сертификации слишком длинная.
CARAPKI_E_BROKEN_CONSTRAINT (0xE0C60020)	Нарушены базовые ограничения.
CARAPKI_E_BROKEN_HIERARCHY (0xE0C60021)	Нарушены ограничения иерархии.
CARAPKI_E_INVALID_CA_CERT (0xE0C60022)	Неверный сертификат Центра Сертификации.
CARAPKI_E_INVALID_RA_CERT (0xE0C60023)	Неверный сертификат Центра Регистрации.
CARAPKI_E_INVALID_CERT_TYPE (0xE0C60024)	Неверный тип сертификата.
CARAPKI_E_CERT_SIGNING_FAILED (0xE0C60025)	Произошла ошибка вычисления ЭП сертификата.
CARAPKI_E_CRL_SIGNING_FAILED (0xE0C60026)	Произошла ошибка вычисления ЭП CAC.
CARAPKI_E_REQ_SIGNING_FAILED (0xE0C60027)	Произошла ошибка вычисления ЭП запроса PKCS#10.
CARAPKI_E_REVREQ_SIGNING_FAILED (0xE0C60028)	Произошла ошибка вычисления ЭП запроса на аннулирование.
CARAPKI_E_DIGEST_FAILED (0xE0C60029)	Произошла ошибка вычисления хэш-значения.
CARAPKI_E_SIGNING_FAILED (0xE0C6002A)	Произошла общая ошибка вычисления ЭП.
CARAPKI_E_INVALID_ISSUER (0xE0C6002B)	Неверное имя издателя.
CARAPKI_E_SUBJECT_MISSING (0xE0C6002C)	Имя владельца отсутствует.
CARAPKI_E_INVALID_SUBJECT (0xE0C6002D)	Неверное имя владельца.
CARAPKI_E_PUBLIC_KEY_MISSING (0xE0C6002E)	Отсутствует ключ проверки ЭП.

Имя и код ошибки	Описание и причина возникновения ошибки
CARAPKI_E_INVALID_PUBLIC_- KEY (0xE0C6002F)	Неверный ключ проверки ЭП.
CARAPKI_E_EXTENSION_- MISSING (0xE0C60030)	Отсутствует расширение X.509v3.
CARAPKI_E_INVALID_- EXTENSION (0xE0C60031)	Неверное расширение X.509v3.
CARAPKI_E_ATTRIBUTE_- MISSING. (0xE0C60032)	Отсутствует требуемый атрибут.
CARAPKI_E_REVOCATION_- MISSING (0xE0C60033)	Отсутствуют данные аннулирования.
CARAPKI_E_INVALID_- REVOCATION (0xE0C60034)	Неверные данные аннулирования.
CARAPKI_E_HANDLING_- PROHIBITED (0xE0C60035)	Обработка запрещена.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

ОС	Операционная система (Operating System)
ПО	Программное обеспечение
САС	Список аннулированных сертификатов
СУС	Система управления сертификатами
ЦР	Центр регистрации
ЦС	Центр сертификации
ЭП	Электронная подпись (Digital Signature)

[illegible][illegible]