Описание крипто-модуля RRIC_COM_OBJECT.Crypto

Общая информация

Крипто-модуль RRIC_COM_OBJECT.Crypto является составной частью COM-объекта RRIC_COM_OBJECT, разработанного ГУП «Республиканский Расчетный Информационный Центр», и предназначен для осуществления функций по созданию электронных документов, управлению электронными документами и электронно-цифровыми подписями.

В крипто-модуле реализован полный перечень крипто-функций, позволяющих добавлять ЭЦП и проверять их подлинность, зашифровывать и расшифровывать электронные документы.

Для работы с крипто-модулем необходимо установить на хост-компьютер COM-объект RRIC_COM_OBJECT и сопутствующие сертификаты клиента и веб-сервиса ГУП «РРИЦ». Также необходимо, чтобы на хост-компьютере был установлен .Net Framework 4.6.

Для операционных систем Windows Server 2003 и XP перед установкой COM-объекта необходимо установить обновление KB938397 (http://thehotfixshare.net/board/index.php?/search/&q=KB938397&quick=1).

Функции

GetLastError

Возвращает описание ошибки для последней выполненной операции

string GetLastError()

Возвращает строку, содержащую описание ошибки.

CreateDocument

Создание нового электронного документа

string CreateDocument()

Возвращает строку, содержащую имя созданного электронного документа.

В случае ошибки возвращает пустую строку.

LoadDocument

Загрузка электронного документа из файла

string LoadDocument(cntPath)

cntPath – Путь к файлу электронного документа.

Возвращает строку, содержащую имя загруженного электронного документа.

В случае ошибки возвращает пустую строку.

UploadDocument

Загрузка электронного документа из потока байтов

string LoadDocument(cntData64, cntName)

cntData64 – base-64 строка. Файловый поток.

cntName - имя электронного документа.

Возвращает строку, содержащую имя созданного электронного документа.

В случае ошибки возвращает пустую строку.

GetDocuments

Получение информации обо всех электронных документах, с которыми ведётся работа в данный момент

string GetDocuments()

Возвращает строку, содержащую xml со следующей структурой:

<root><doc name="имя электронного документа" status="статус" /></root>

В случае ошибки возвращает пустую строку.

GetDocumentContent

Получение текстового содержимого электронного документа

string GetDocumentContent(string cntName)

cntName – Имя электронного документа.

Возвращает строку с содержимым электронного документа.

В случае ошибки возвращает пустую строку.

DeleteDocument

Удаление электронного документа

int DeleteDocument(string cntName)

cntName – Имя электронного документа.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

SetDocInfoParameter

Добавление/редактирование параметра общей информации об электронном документе

int SetDocInfoParameter(string cntName, string parName, string parValue)

cntName – Имя электронного документа.

parName – Имя параметра. Может быть одним из списка, указанного ниже.

parValue – Значение параметра. Должно быть строкой.

Имя параметра	Описание
Наименование	
namemainru	Наименование вышестоящей организации на русском
namemainukr	Наименование вышестоящей организации на украинском
namemainmd	Наименование вышестоящей организации на молдавском
nameru	Наименование организации на русском
nameukr	Наименование организации на украинском
namemd	Наименование организации на молдавском
Справочные данные об организации	
referencedataorg/postaladdress	Почтовый адрес
referencedataorg/phonenumber	Номер телефона
referencedataorg/faxnumber	Номер факса
referencedataorg/telexnumber	Номер телекса
referencedataorg/bankaccount	№ счета в банке
referencedataorg/emailaddress	Адрес электронной почты
referencedataorg/otheraddress	Другие сведения по усмотрению организации
Вид документа	
nameviewdoc	Наименование вида документа
Регистрационные данные	
reg/datareg	Дата документа
reg/regnumber	Регистрационный номер документа
reg/vhodregnumber	Регистрационный номер входящего документа
reg/vhodregdate	Дата входящего документа
Исполнитель	
executor/executorname	Фамилия
executor/executorphone	Номер телефона
Содержание	
header	Заголовок к тексту
content	Текст документа

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

GetDocInfoParameterValue

Извлечение параметра общей информации об электронном документе.

string GetDocInfoParameterValue(string cntName, string parName)

cntName – Имя электронного документа.

parName – Имя параметра. Может быть одним из списка, указанного в описании метода SetDocInfoParameter.

Возвращает строку, содержащую значение параметра.

В случае ошибки возвращает пустую строку.

SetDocInfoLogo

Добавление логотипа организации, выводимого на бланк электронного документа

int SetDocInfoLogo(string cntName, string logo_image64)

cntName – Имя электронного документа.

logo_image64 – base64-строка байтового массива изображения логотипа. В бланке электронного документа поддерживаются только PNG-изображения размером 97x105 пикселей. Если изображение будет меньше или больше стандартных значений, то оно будет соответственно увеличено или уменьшено до приемлемых значений ширины и высоты.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

AddRecipient

Добавление данных получателя электронного документа.

int AddRecipient(string cntName, string org, string branch, string pos, string fio)

cntName – Имя электронного документа.

org – Наименование организации.

branch – Наименование подразделения.

pos – Наименование должности.

fio – ФИО лица.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

RemoveRecipient

Удаление данных указанного получателя электронного документа.

int RemoveRecipient(string cntName, int index)

cntName – Имя электронного документа.

index – Номер по порядку в списке получателей (нумерация начинается с 0).

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

RemoveRecipients

Удаление всех получателей электронного документа.

int RemoveRecipients(string cntName)

cntName – Имя электронного документа.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

GetRecipients

Получение данных о всех получателях электронного документа.

string GetRecipients(string cntName)

cntName – Имя электронного документа.

В случае успеха возвращает xml с данными о лицах, подписавших электронный документ, со следующей структурой:

<root><destination><org>организация</org><branch>подразделение</branch><post>должность</post><fio>ФИО</fio></destination></root>

В случае ошибки возвращает пустую строку.

AddSigner

Добавление данных подписывающего лица организации.

int AddSigner(string cntName, string position, string fio)

cntName – Имя электронного документа.

position – Наименование должности.

fio – ФИО лица.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

RemoveSigner

Удаление данных подписывающего лица организации.

int RemoveSigner(string cntName, int index)

cntName – Имя электронного документа.

index – Номер по порядку в списке подписавших лиц (нумерация начинается с 0).

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

RemoveSigners

Удаление всех данных о лицах, подписавших документ.

int RemoveSigners(string cntName)

cntName – Имя электронного документа.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

GetSigners

Получение данных о всех лицах, подписавших документ.

string GetSigners(string cntName)

cntName – Имя электронного документа.

В случае успеха возвращает xml с данными о лицах, подписавших электронный документ, со следующей структурой:

<root><signaturestextinfo><post>должность</post><fio>ФИО</fio></signaturestextinfo></root>

В случае ошибки возвращает пустую строку.

AddRawData

Добавление сырых данных в электронный документ.

int AddRawData(string cntName, string rawData64)

cntName – Имя электронного документа.

rawData64 – base-64 строка с данными.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

GetRawData

Извлечение сырых данных их электронного документа.

string GetRawData(string cntName)

cntName – Имя электронного документа.

В случае успеха возвращает base-64 строку с сырыми данными электронного документа.

В случае ошибки возвращает пустую строку.

RemoveRawData

Удаление сырых данных из электронного документа.

int RemoveRawData(string cntName)

cntName – Имя электронного документа.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

AddAttachment1

Добавление файла вложения электронного документа, используя путь к файлу вложения.

int AddAttachment1(string cntName, string docPath)

cntName – Имя электронного документа.

docPath – Путь к файлу вложения.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

AddAttachment2

Добавление файла вложения электронного документа, используя содержимое файла вложения.

AddAttachment2(string cntName, string docName, string docData64)

cntName – Имя электронного документа.

docName – Имя файла вложения (включая расширение файла)

docData64 – строка base64. Содержимое файла вложения.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

RemoveAttachment

Удаление файла вложения электронного документа.

string RemoveAttachment (string cntName, string docName)

cntName – Имя электронного документа.

docName – Имя файла вложения.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

GetAttachment

Получение содержимого файла вложения электронного документа.

string GetAttachment (string cntName, string docName)

cntName – Имя электронного документа.

docName – Имя файла вложения.

Возвращает base64 строку содержимого файла вложения.

В случае ошибки возвращает пустую строку.

SaveAttachment

Сохранение файла вложения электронного документа в указанное место на диске.

int SaveAttachment (string cntName, string docName, string docPath)

cntName – Имя электронного документа.

docName – Имя файла вложения.

docPath – Путь к сохраняемому файлу на диске.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

GetAttachmentsCount

Получение количества файлов, вложенных в электронный документ.

int GetAttachmentsCount(string cntName)

cntName – Имя электронного документа.

В случае успеха возвращает количество файлов, вложенных в электронный документ.

В случае ошибки возвращает -1.

GetAttachmentsList

Получение списка файлов, вложенных в электронный документ.

string GetAttachmentsList(string cntName)

cntName – Имя электронного документа.

В случае успеха возвращает base64-строку, содержащую xml со следующей структурой:

<docs><doc name="имя документа" size="размер документа" /></docs>

В случае ошибки возвращает пустую строку.

SignDocument1

Подписание электронного документа электронно-цифровой подписью, используя сертификат безопасности, установленный в хранилище сертификатов.

string SignDocument1(string cntName, string storeLocation, string storeName, int flags, string tsaService)

cntName – Имя электронного документа.

storeLocation – Имя хранилища сертификатов.

storeName – Имя каталога сертификатов.

flags – Флаг режима извлечения сертификата (см. Перечисления->OpenFlags).

tsaService – URL сервера штампов времени (используется для формирования и добавления к подписи штампа времени).

В случае успеха возвращает строку с идентификатором созданной подписи.

В случае ошибки возвращает пустую строку.

SignDocument2

Подписание электронного документа электронно-цифровой подписью, используя сертификат безопасности (в формате pkcs#12) с диска.

string SignDocument2(string cntName, string certPath, string certPass, string tsaService)

cntName – Имя электронного документа.

certPath – Путь к файлу сертификата (файл с расширением *.pfx, *.crt).

certPass – Пароль (ПИН-код) доступа к файлу сертификата.

tsaService – URL сервера штампов времени (используется для формирования и добавления к подписи штампа времени).

В случае успеха возвращает строку с идентификатором созданной подписи.

В случае ошибки возвращает пустую строку.

SignDocument3

Подписание электронного документа электронно-цифровой подписью, используя отпечаток сертификата безопасности.

string SignDocument3(string cntName, string certThumbprint, string tsaService)

cntName – Имя электронного документа.

certThumbprint – Отпечаток сертификата безопасности.

tsaService – URL сервера штампов времени (используется для формирования и добавления к подписи штампа времени).

В случае успеха возвращает строку с идентификатором созданной подписи.

В случае ошибки возвращает пустую строку.

SignSignature1

Подписание электронно-цифровой подписи – визирование, используя сертификат безопасности, установленный в хранилище сертификатов.

string SignSignature1(string cntName, string sigId, string storeLocation, string storeName, int flags, string tsaService)

cntName – Имя электронного документа.

sigId – Идентификатор визируемой электронно-цифровой подписи.

storeLocation – Имя хранилища сертификатов.

storeName – Имя каталога сертификатов.

flags – Флаг режима извлечения сертификата (см. Перечисления->OpenFlags).

tsaService – URL сервера штампов времени (используется для формирования и добавления к подписи штампа времени).

В случае успеха возвращает строку с идентификатором созданной подписи.

В случае ошибки возвращает пустую строку.

SignSignature2

Подписание электронно-цифровой подписи – визирование, используя сертификат безопасности (в формате pkcs#12) с диска.

string SignSignature2(string cntName, string sigId, string certPath, string certPass, string tsaService)

cntName – Имя электронного документа.

sigId – Идентификатор визируемой электронно-цифровой подписи.

certPath – Путь к файлу сертификата (файл с расширением *.pfx, *.crt).

certPass – Пароль (ПИН-код) доступа к файлу сертификата.

tsaService – URL сервера штампов времени (используется для формирования и добавления к подписи штампа времени).

В случае успеха возвращает строку с идентификатором созданной подписи.

В случае ошибки возвращает пустую строку.

SignSignature3

Подписание электронно-цифровой подписи – визирование, используя отпечаток сертификата безопасности.

string SignSignature2(string cntName, string certThumbprint, string tsaService)

cntName – Имя электронного документа.

certThumbprint – Отпечаток сертификата безопасности.

tsaService – URL сервера штампов времени (используется для формирования и добавления к подписи штампа времени).

В случае успеха возвращает строку с идентификатором созданной подписи.

В случае ошибки возвращает пустую строку.

RemoveSignature1

Удаление электронно-цифровой подписи, используя сертификат безопасности, установленный в хранилище сертификатов.

int RemoveSignature1(string cntName, string sigId, string storeLocation, string storeName, int flags)

cntName – Имя электронного документа.

sigId – Идентификатор визируемой электронно-цифровой подписи.

storeLocation – Имя хранилища сертификатов.

storeName – Имя каталога сертификатов.

flags – Флаг режима извлечения сертификата (см. Перечисления->OpenFlags).

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

RemoveSignature2

int RemoveSignature2(string cntName, string sigId, string certPath, string certPass)

cntName – Имя электронного документа..

sigId – Идентификатор визируемой электронно-цифровой подписи.

certPath – Путь к файлу сертификата (файл с расширением *.pfx, *.crt).

certPass – Пароль (ПИН-код) доступа к файлу сертификата.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

GetSigCount

Получение количества ЭЦП в электронном документе.

int GetSigCount(string cntName)

cntName – Имя электронного документа.

В случае успеха возвращает количество ЭЦП в электронном документе.

В случае ошибки возвращает -1.

GetSigList

Получение списка ЭЦП в электронном документе.

string GetSigList(string cntName)

cntName – Имя электронного документа.

В случае успеха возвращает строку, содержащую xml со следующей структурой:

<root><signature id="идентификатор ЭЦП" sign_type="тип ЭЦП" child_id="идентификатор визированной ЭЦП" digest="подпись" createtime="время создания" timestamp="штамп времени" hash_algo="хэш-алгоритм" hash_algo_oid="OID хэш-алгоритма" sig_algo="алгоритм подписи" sig_algo_oid="OID алгоритма подписи" cert="содержимое файла сертификата" /></root>

Где:

sign_type – одно из значений (0 – обычная подпись, 1 – визирующая подпись);

digest, timestamp, cert – строки base64

В случае ошибки возвращает пустую строку.

ValidateSignatures

Проверка всех электронно-цифровых подписей, добавленных в электронный документ - валидация

string ValidateSignatures(string cntName)

cntName – Имя электронного документа.

В случае успеха возвращает строку, содержащую xml со следующей структурой:

<root><signature id="идентификатор ЭЦП" sign_type="тип ЭЦП" validation="результат проверки" /></root>

Где:

sign_type – одно из значений (0 – обычная подпись, 1 – визирующая подпись);

validation – одно из значений ("Valid" – подпись подьверждена, "Invalid" – подпись не подтверждена).

В случае ошибки возвращает пустую строку.

ValidateSignature

Проверка заданной электронно-цифровой подписи, добавленной в электронный документ - валидация

string Validate(string cntName, string sigId)

cntName – Имя электронного документа.

sigId – Идентификатор визируемой электронно-цифровой подписи.

В случае положительной проверки возвращает одно из значений (1 – подпись подтверждена, 0 – подпись не подтверждена).

В случае ошибки возвращает пустую строку.

EncryptDocument1

Зашифрование электронного документа, используя сертификат безопасности, установленный в хранилище сертификатов.

int EncryptDocument1(string cntName, string storeLocation, string storeName, int flags, int multy)

cntName – Имя электронного документа.

storeLocation – Имя хранилища сертификатов.

storeName – Имя каталога сертификатов.

flags – Флаг режима извлечения сертификата (см. Перечисления->OpenFlags).

multy – Режим диалогового окна выбора сертификата (0-диалоговое окно одиночного выбора, 1-диалоговое окно множественного выбора) .

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

EncryptDocument2

Зашифрование электронного документа, используя сертификат безопасности (в формате pkcs#12) с диска.

int EncryptDocument2(string cntName, string certPath, int flag)

cntName – Имя электронного документа.

certPath – Путь к файлу сертификата (файл с расширением *.pfx, *.crt).

flag – должен быть равен 1.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

EncryptDocument3

Зашифрование электронного документа, используя список сертификатов безопасности.

int EncryptDocument(string cntName, string certData, int flag)

cntName – Имя электронного документа.

flag – должен быть равен 2.

certData – данные сертификатов безопасности получателей XML-контейнера в виде xml со следующей структурой:

<root><cert data="содержимое файла сертификата в base64"" /><cert data="содержимое файла сертификата в base64"" /></root>

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

DecryptDocument

Расшифрование электронного документа.

int DecryptDocument(string cntName)

cntName – Имя электронного документа.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

CompressDocument

Сжатие (архивирование) электронного документа.

int CompressDocument(string cntName)

cntName – Имя электронного документа.

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

DecompressDocument

Извлечение (разархивирование) электронного документа.

int DecompressDocument(string cntName)

cntName – Имя электронного документа..

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

GetDocumentStatus

Получение признака текущего состояния электронного документа.

string GetDocumentStatus(string cntName)

cntName – Имя электронного документа.

Возвращает строку с текущим статусом электронного документа (см. Перечисления->CntStat).

В случае ошибки возвращает строку с фразой, содержащую слово «NotDefined».

ClearAll

Очистка временного хранилища электронных документов.

string ClearAll()

В случае успеха возвращает 0.

В случае ошибки возвращает -1.

GetCertificate1

Извлечение сертификата безопасности, установленного в хранилище сертификатов.

string GetCertificate1(string storeLocation, string storeName, int flags)

storeLocation – Имя хранилища сертификатов.

storeName – Имя каталога сертификатов.

flags – Флаг режима извлечения сертификата (см. Перечисления->OpenFlags).

В случае успеха возвращает base-64 строку содержимого сертификата.

В случае ошибки возвращает пустую строку.

GetCertificate2

Извлечение сертификата безопасности из файла в формате pkcs#12 с диска.

string GetCertificate2(string certPath)

certPath – Путь к файлу сертификата (файл с расширением *.pfx, *.crt).

В случае успеха возвращает base-64 строку содержимого сертификата.

В случае ошибки возвращает пустую строку.

GetStampInfo

Извлечение информации о штампе времени

string GetStampInfo(string cntName, string sigId)

cntName – Имя электронного документа.

sigId – идентификатор подписи.

В случае успеха возвращает XML-строку с данными штампа времени со следующей структурой:

```
<root>
        <timestamp serial_number="Серийный номер" created="Дата создания" data_hash="Хэш подписанных данных">
                <!—данные сертификата сервера -->
                <TSA issuer_name="Кем выдан" issuer_serial="Серийный номер" tsa_name="Кому выдан" tsa_from_date="Действует
                с даты" tsa_to_date="Действует до даты" />
        </timestamp>
</root>
```

В случае ошибки возвращает пустую строку.

GetStampDate

Извлечение даты формирования штампа времени

string GetStampDate(string cntName, string sigId)

cntName – Имя электронного документа.

sigId – идентификатор подписи.

В случае успеха возвращает дату штампа времени в формате dd.MM.yyyy hh:mm:ss

В случае ошибки возвращает пустую строку.

SignatureInfo

Извлечение информации о подписи

string SignatureInfo(string cntName, string sigId)

cntName – Имя электронного документа.

sigId – идентификатор подписи.

В случае успеха возвращает XML-строку с данными подписи со следующей структурой:

```
<root>
        <signature>
                <id>идентификатор подписи</id>
                <sign_type>тип подписи</sign_type>
                <sign_value>подпись</sign_value>
                <hasstamp>флаг наличия штампа времени</hasstamp>
                <hash_algo>алгоритм хэширования</hash_algo>
                <hash_algo_oid>OID алгоритма хэширования</hash_algo_oid>
                <sig_algo>алгоритм подписи</sig_algo>
                <sig_algo_oid>OID алгоритма подписи</sig_algo_oid>
                <!—данные сертификата -->
                <certificate>
                        <version>версия</version>
                        <serial>серийный номер</serial>
                        <subject>владелец</subject>
                        <validity_from>действителен с даты</validity_from>
```

```
                <validity_till>действителен до даты</validity_till>
                <issuer>кем выдан</issuer>
                <key_usage>назначение ключа сертификата</key_usage>
            </certificate>
        </signature>
        <!—список заверяющих подписей -->
        <counter_signatures>
            <counter_signature id=" идентификатор подписи" serial="серийный номер сертификата"
            thumbprint="отпечаток сертификата" subject="владелец сертификата" />
        </counter_signatures>
</root>
```

В случае ошибки возвращает пустую строку.

CertificateInfo1

Извлечение информации о сертификате подписи

string SignatureInfo(string cntName, string sigId)

cntName – Имя электронного документа.

sigId – идентификатор подписи.

В случае успеха возвращает XML-строку с данными подписи со следующей структурой:

```
<root>
        <version>версия</version>
        <serial>серийный номер</serial>
        <thumbprint>отпечаток</thumbprint>
        <subject>владелец</subject>
        <validity_from>действителен с даты</validity_from>
        <validity_till>действителен до даты</validity_till>
        <issuer>кем выдан</issuer>
        <key_usage>назначение ключа сертификата</key_usage>
        <clientId>ID владельца</clientId>
        <clientMvdId>ID МВД владельца</clientMvdId>
        <clientBirthDate>дата рождения владельца</clientBirthDate>
        <clientOrgRegNum>регистрационный номер организации владельца</clientOrgRegNum>
        <clientOrgFiscalCode>фискальный код организации владельца</clientOrgFiscalCode>
</root>
```

В случае ошибки возвращает пустую строку.

CertificateInfo2

Извлечение информации о сертификате

string SignatureInfo(string certData64)

certData64 – base-64 строка содержимого сертификата.

В случае успеха возвращает XML-строку с данными подписи со следующей структурой:

```
<root>
        <version>версия</version>
        <serial>серийный номер</serial>
        <thumbprint>отпечаток</thumbprint>
        <subject>владелец</subject>
        <validity_from>действителен с даты</validity_from>
        <validity_till>действителен до даты</validity_till>
        <issuer>кем выдан</issuer>
        <key_usage>назначение ключа сертификата</key_usage>
        <clientId>ID владельца</clientId>
        <clientMvdId>ID МВД владельца</clientMvdId>
        <clientBirthDate>дата рождения владельца</clientBirthDate>
        <clientOrgRegNum>регистрационный номер организации владельца</clientOrgRegNum>
        <clientOrgFiscalCode>фискальный код организации владельца</clientOrgFiscalCode>
```

</root>

В случае ошибки возвращает пустую строку.

ConvertToBase64String

Конвертация в base-64 строку

string ConvertToBase64String(string input)

input – строка данных.

В случае успеха возвращает сконвертированную в base-64 строку данных

В случае ошибки возвращает пустую строку.

ConvertFromBase64String

Конвертация из base-64 строки

string ConvertFromBase64String(string input)

input – base-64 строка данных.

В случае успеха возвращает сконвертированную из base-64 строку данных

В случае ошибки возвращает пустую строку.

Перечисления

OpenFlags

IncludeArchived	8	Open the X.509 certificate store and include archived certificates.
MaxAllowed	2	Open the X.509 certificate store for the highest access allowed.
OpenExistingOnly	4	Opens only existing stores; if no store exists, the Open(OpenFlags) method will not create a new store.
ReadOnly	0	Open the X.509 certificate store for reading only.
ReadWrite	1	Open the X.509 certificate store for both reading and writing.

CntStat

NotDefined	0	Не определено
Clean	1	Без вложений и ЭЦП
Filled	2	Есть вложения
Signed	3	Есть ЭЦП
Crypted	4	Зашифрован
Compressed	5	Сжат

Структура «голых» данных электронных документов

1. Структура суточной заявки на газопотребление

```
<?xml version="1.0" encoding="windows-1251"?>
<root>
	<form_type>Суточная заявка на газопотребление</form_type>  <!-- тип заявки -->
	<org_id></org_id>                    <!-- Код потребителя -->
	<org_name></org_name>        <!-- Наименование потребителя -->
	<dog_num></dog_num>          <!-- Номер договора -->
	<dog_date></dog_date>        <!-- Дата договора -->
	<comorg_id></comorg_id>      <!-- Код подразделения ТТГП -->
	<comorg_name></comorg_name>  <!-- Наименование подразделения ТТГП -->
	<ras_date></ras_date>        <!-- Расчетная дата -->
	<full_value>0</full_value>   <!-- Полный объем газа по заявке -->
	<rows>                       <!-- почасовой объем -->
		<row id="0" start="9:00" end="10:00" value="0" />
		<row id="1" start="10:00" end="11:00" value="0" />
		<row id="2" start="11:00" end="12:00" value="0" />
		<row id="3" start="12:00" end="13:00" value="0" />
		<row id="4" start="13:00" end="14:00" value="0" />
		<row id="5" start="14:00" end="15:00" value="0" />
		<row id="6" start="15:00" end="16:00" value="0" />
		<row id="7" start="16:00" end="17:00" value="0" />
		<row id="8" start="17:00" end="18:00" value="0" />
		<row id="9" start="18:00" end="19:00" value="0" />
		<row id="10" start="19:00" end="20:00" value="0" />
		<row id="11" start="20:00" end="21:00" value="0" />
		<row id="12" start="21:00" end="22:00" value="0" />
		<row id="13" start="22:00" end="23:00" value="0" />
		<row id="14" start="23:00" end="00:00" value="0" />
		<row id="15" start="00:00" end="01:00" value="0" />
		<row id="16" start="01:00" end="02:00" value="0" />
		<row id="17" start="02:00" end="03:00" value="0" />
		<row id="18" start="03:00" end="04:00" value="0" />
		<row id="19" start="04:00" end="05:00" value="0" />
		<row id="20" start="05:00" end="06:00" value="0" />
		<row id="21" start="06:00" end="07:00" value="0" />
		<row id="22" start="07:00" end="08:00" value="0" />
		<row id="23" start="08:00" end="09:00" value="0" />
	</rows>
</root>
```

2. Структура годовой заявки на газопотребление

```
<?xml version="1.0" encoding="windows-1251"?>
<root>
	<form type>Годовая заявка на газопотребление</form type>   <!-- тип заявки -->
	<org_id></org_id>            <!-- Код потребителя -->
	<org_name></org_name>        <!-- Наименование потребителя -->
	<fiscode></fiscode>          <!-- Фискальный код потребителя -->
	<comorg_id></comorg_id>      <!-- Код подразделения ТТГП -->
	<comorg_name></comorg_name>  <!-- Наименование подразделения ТТГП -->
	<ras_adr></ras_adr>          <!-- Адрес расчетной точки -->
	<ras_year></ras_year>        <!-- Расчетный год -->
	<full_value></full_value>    <!-- Полный объем газа по заявке -->
	<rows>                       <!-- почасовой объем -->
		<row id="1" month="январь" value="" />      <!—Код и название месяца, объем -->
		<row id="2" month="февраль" value="" />
		<row id="3" month="март" value="" />
		<row id="4" month="апрель" value="" />
		<row id="5" month="май" value="" />
		<row id="6" month="июнь" value="" />
		<row id="7" month="июль" value="" />
		<row id="8" month="август" value="" />
		<row id="9" month="сентябрь" value="" />
		<row id="10" month="октябрь" value="" />
		<row id="11" month="ноябрь" value="" />
		<row id="12" month="декабрь" value="" />
	</rows>
</root>
```

3. Структура годовой заявки на электропотребление

```
<?xml version="1.0" encoding="windows-1251"?>
<root>
	<form type>Годовая заявка на электропотребление</form type><!-- тип заявки -->
	<liz></liz>                  <!-- Лицевой счёт абонента -->
	<org_id></org_id>            <!-- Код организации абонента -->
	<org_name></org_name>        <!-- Наименование организации абонента -->
	<fiscode></fiscode>          <!-- Фискальный код организации абонента -->
```

```
        <comorg_id></comorg_id>       <!-- Код подразделения ЕРЭС -->
        <comorg name></comorg name>   <!-- Наименование подразделения ЕРЭС -->
        <ras year></ras year>         <!—Год заявки -->
        <plan>                        <!-- Объемы заявки -->
        <rows type="1" full_value="0">       <!-- Помесячный объем электроэнергии по заявке -->
                <row id="1" month="январь" value="" />      <!—Код и название месяца, объем -->
                <row id="2" month="февраль" value="" />
                <row id="3" month="март" value="" />
                <row id="4" month="апрель" value="" />
                <row id="5" month="май" value="" />
                <row id="6" month="июнь" value="" />
                <row id="7" month="июль" value="" />
                <row id="8" month="август" value="" />
                <row id="9" month="сентябрь" value="" />
                <row id="10" month="октябрь" value="" />
                <row id="11" month="ноябрь" value="" />
                <row id="12" month="декабрь" value="" />
        </rows>
        <rows type="2" full value="0">       <!-- Помесячная мощность по заявке -->
                <row id="1" month="январь" value="" />      <!—Код и название месяца, объем -->
                <row id="2" month="февраль" value="" />
                <row id="3" month="март" value="" />
                <row id="4" month="апрель" value="" />
                <row id="5" month="май" value="" />
                <row id="6" month="июнь" value="" />
                <row id="7" month="июль" value="" />
                <row id="8" month="август" value="" />
                <row id="9" month="сентябрь" value="" />
                <row id="10" month="октябрь" value="" />
                <row id="11" month="ноябрь" value="" />
                <row id="12" month="декабрь" value="" />
        </rows>
        </plan>
</root>
```

4. Структура показаний приборов учета

```
<?xml version="1.0" encoding="windows-1251"?>
<root>
        <form_type>Показания приборов учета</form_type>     <!-- тип документа -->
        <liz></liz>                           <!-- Лицевой счет абонента -->
        <comorg_id></comorg_id>               <!-- Код энергоснабжающей организации -->
        <pok_date></pok_date>                 <!-- Дата ввода показаний -->
        <rows>                        <!-- Блоки данных по потребителям -->
                <potr>                <!-- Данные потребителя -->
                        <kod></kod>   <!-- код -->
                        <id></id>     <!-- id -->
                        <name></name> <!-- наименование -->
                        <adr></adr>   <!-- адрес -->
                        <devices>     <!—Данные приборов учета -->
                                <!-- Данные прибора учета (код, тип, номер) -->
                                <device num="" kod="" name="" type="">
                                        <!-- Показания (вид энергии, предыдущие показания,
                                        текущие показания, объем) -->
                                        <pok tip="" prev="" cur="" val=""/>
                                </device>
                        </devices>
                </potr>
        </rows>
        <plan></plan>  <!-- Ожидаемое потребление на следующий месяц (только для ЕРЭС) -->
</root>
```

Примеры

1. Пример вызова некоторых функций крипто-модуля в Visual FoxPro 8+

```
x=CREATEOBJECT("RRIC COM OBJECT.Crypto")
* новый документ
y=x.CreateDocument()
?y
IF !EMPTY(y)
        * заполнение поле общей информации документа
        x.SetDocInfoParameter(y, "nameru", "ООО Ивушка")
        x.SetDocInfoParameter(y, "nameviewdoc", "Письмо")
        x.SetDocInfoParameter(y, "postaladdress", "г.Каменка, ул.Ленина, д.15/7")
        x.SetDocInfoParameter(y, "datareg", "15.01.2021")
        x.SetDocInfoParameter(y, "regnumber", "12/23-3434")
        * заполнение текстового содержимого документа
        x.SetDocInfoParameter(y, "content", "Это текст документа, содержащий какую-то
информацию.")
        * Добавление получателя
```

```
x.AddRecipient(y, "ООО Колокольчик", "", "Директору", "О.И. Зыкиной")
* информация о подписантах
x.AddSigner(y, "Директор", "И.И. Иванов")
x.AddSigner(y, "Заместитель директора", "О.П. Сидоров")
* информация об исполнителе
x.SetDocInfoParameter(y, "executorname", "А.А. Петрова")
x.SetDocInfoParameter(y, "executorphone", "216 22222")
* добавление логотипа организации
*c.SetDocInfoLogo(y, "base64-строка байтового массива PNG-файла")

* добавление вложения
z=x.AddAttachment1(y,"путь к файлу")
* подпись документа
s=x.SignDocument1(y,"CurrentUser","My",4,"")
?s
IF !EMPTY(s)
        * проверка подписи
        z=x.ValidateSignature(y,s)
        ?z
ELSE
        c=x.GetLastError()
        ?c
ENDIF

* если нужно сохранить xml в файл на диске, сохраняем
* извлечение содержимого документа
v=x.GetDocumentContent(y)
* сохранение документа в файл
r=STRCONV(v,9)
gnFileHandle = FCREATE("путь к xml-файлу")
=FWRITE(gnFileHandle, r)
= FCLOSE(gnFileHandle)

* если нужно зашифровать документ
* зашифрование документа
c=x.EncryptDocument1(y,"CurrentUser","My",4,0)
?c
IF c=0
        * сохранение зашифрованного документа в файл
        v=x.GetDocumentContent(y)
        r=STRCONV(v,9)
        gnFileHandle = FCREATE("путь к xml-файлу")
        =FWRITE(gnFileHandle, r)
        = FCLOSE(gnFileHandle)
        * расшифрование документа
        c=x.DecryptDocument(y)
        ?c
        * проверка подпси
        z=x.ValidateSignature(y,s)
        ?z
ELSE
        c=x.GetLastError()
        ?c
ENDIF

* если нужно сохранить документ в базу данных АПК «КРЭДО»
* сжимаем документ
x.CompressDocument(y)
* получаем статус документа
w=x.GetDocumentStatus(y)
* преобразуем текстовое значение статуса в численное согласно списка статусов
st = 0
DO CASE
        CASE w="Clean"
                st=1
        CASE w="Filled"
                st=2
        CASE w="Signed"
                st=3
        CASE w="Crypted"
                st=4
        CASE w="Compressed"
                st=5
        OTHERWISE
                st=0
ENDCASE
* извлечение содержимого документа
v=x.GetDocumentContent(y)
* преобразуем документ в base64
v = x.ConvertToBase64String(v)
```

```
        * подключаемся к каналу обмена
        invObj = CREATEOBJECT("RRIC COM OBJECT.Invoker")
        invObj.ClientCertName = "сертификат клиента"
        invObj.ServiceCertName = "Rric Wcf Service"
        invObj.EndpointUrl = "http://www.rric.org:7742/RRIC_EP_SERVICE/WcfService"
        invObj.MaxMessageSize = 100000000
        channel = invObj.CreateChannel()
        y = invObj.ExecProc(channel, ;
        "Set_URDocument", ;
        "@type=3,@org_out=83599402-48C8-492C-98F2-C8FD0BBA7E38, ;
        @pid_out=1a3f3568-a155-4ba5-8f00-52e7a2025943, ;
        @org_in=472256F0-285E-4934-9C7D-5765925688AC, ;
        @pid in=ae15e09d-67be-4445-b58a-4967038245f0, ;
        @doc stat="+ALLTRIM(STR(st))+",@doc data={base64}"+v)
        * добавляем обработку ошибок
        * закрываем канал
        invObj.CloseChannel(channel)

        * очистка временного хранилища крипто-модуля
        x.ClearAll()
ELSE
        c=x.GetLastError()
        ?c
ENDIF
RELEASE x,y,z,c
```

2. Пример вызова некоторых функций крипто-модуля в JavaScript (только IE8+)

```
x = new ActiveXObject("RRIC_COM_OBJECT.Crypto");
y=x.CreateDocument();
console.log(y);
if(y!="")
{
        z=x.AddAttachment1(y,"экранированный путь к файлу");
        console.log(z);
        if(z==0)
        {
                s=x.SignDocument1(y,"CurrentUser","My",4,"");
                console.log(s);
                if(s!="")
                {
                        z=x.ValidateSignature(y,s);
                        console.log(z);
                        c=x.EncryptDocument1(y,"CurrentUser","My",4,0);
                        console.log(c);
                        if(c==0)
                        {
                                c=x.DecryptDocument(y);
                                console.log(c);
                                z=x.ValidateSignature(y,s);
                                console.log(z);
                        }
                        else
                        {
                                c=x.GetLastError();
                                console.log(c);
                        }
                }
                else
                {
                        c=x.GetLastError();
                        console.log(c);
                }
        }
        else
        {
                c=x.GetLastError();
                console.log(c);
        }
}
else
{
        c=x.GetLastError();
        console.log(c);
}
```

3. Пример получения электронного документа из базы данных АПК «КРЭДО», его распаковки и отображения в браузере (FoxPro 8+).

```
В реальном проекте необходимо добавить обработку ошибок

* подключение к Invoker для взаимодействия с web-сервисом ГУП «РРИЦ»
inv = CREATEOBJECT("RRIC COM OBJECT.Invoker")
inv.ClientCertName = "имя сертификата"
inv.ServiceCertName = "Rric Wcf Service"
inv.EndpointUrl = "http://www.rric.org:7742/RRIC_EP_SERVICE/WcfService"
inv.MaxMessageSize = 1000000000
* открытие канала связи
chn = inv.CreateChannel()
* Вызов процедуры получения данных, содержащих электронный документ из базы данных АПК «КРЭДО»
doc = inv.ExecProc(chn,"Kredo_DocumentData", "@doc_id=D4237372-E041-458C-9EDC-AB3175A757E8")
* закрытие канала
inv.CloseChannel(chn)
RELEASE inv
* данные в курсор
=XMLTOCURSOR(doc, "doc")
* поля курсора в массив
SELECT doc_id, doc_data FROM doc INTO ARRAY doc_arr


* объявление Win32 API-функции для запуска внешней программы
declare long ShellExecute in "shell32.dll" ;
long hwnd, string lpszOp, ;
string lpszFile, string lpszParams, ;
string lpszDir, long nShowCmd


* подключение крипто-модуля
cr=CREATEOBJECT("RRIC_COM_OBJECT.Crypto")
* загрузка документа во временное хранилище крипто-модуля
dn=cr.UploadDocument(doc_arr[1,2], doc_arr[1,1])
* разархивирование документа
cr.DecompressDocument(dn)
* считывание содержимого документа из временного хранилища
content=cr.GetDocumentContent(dn)
* смена кодировки на UTF-8
r=STRCONV(content,9)
* запись электронного документа в файл с расширением xml
gnFileHandle = FCREATE("путь к файлу")
=FWRITE(gnFileHandle, r)
= FCLOSE(gnFileHandle)
* запуск браузера Chrome (поменяйте путь, если используете другой браузер) с передачей ему пути к
файлу xml электронного документа
=ShellExecute(0, "", "C:\Program Files (x86)\Google\Chrome\Application\chrome.exe", "путь к
файлу", 0, 1)

* удаление электронного документа из временного хранилища крипто-модуля
cr.DeleteDocument(dn)

RELEASE cr
```

4. Пример получения электронных документов, содержащих показания приборов учета, из базы данных АПК «КРЭДО», их распаковки и извлечения из них информации о показаниях приборов учета (FoxPro 8+).

```
x = CREATEOBJECT("RRIC_COM_OBJECT.Invoker")
x.ClientCertName = "имя сертификата"
x.ServiceCertName = "Rric Wcf Service"
x.EndpointUrl = "http://www.rric.org:7742/RRIC_EP_SERVICE/WcfService"
x.MaxMessageSize = 1000000000
c = x.CreateChannel()

* Получить все документы для своей организации (на примере ГУП "ЕРЭС")
y = x.ExecProc(c,"Kredo_DocumentsIn", "@comorg=300")
IF AT("error>", y) > 0
        * обработка ошибки
        a = AT("<error>",y,1)+7
        b = (LEN(y)-a)-(LEN(y)-AT("</error>",y,1))
        =MESSAGEBOX(SUBSTR(y, a, b))
ELSE
        * данные в курсор
        =XMLTOCURSOR(y, "acn")
```

```
* извлечь документы ГУП «ЕРЭС» с типом 13 (показания приборов учета)
SELECT doc_id, dstat FROM acn WHERE dtype=13 INTO ARRAY dd
FOR d=1 TO ALEN(dd,1)
        * данные элемента массива в переменные для дальнейшего использования
        doc_id = ALLTRIM(dd[d,1])
        doc_stat = dd[d,2]
        * получить содержимое электронного документа
        y = x.ExecProc(c,"Kredo_DocumentData", "@doc_id="+doc_id)
        * данные в курсор
        =XMLTOCURSOR(y, "doc")
        * извлечь содержимое сжатого электронного документа
        SELECT doc_data FROM doc INTO ARRAY ddd
        doc_data = ALLTRIM(ddd[1,1])
        * подключаем крипто-модуль
        rc=CREATEOBJECT("RRIC_COM_OBJECT.Crypto")
        * загрузить электронный документ во временное хранилище крипто-модуля
        rc.UploadDocument(doc_data,doc_id)
        * проверка на ошибки
        er=rc.GetLastError()
        IF er<>""
                ?er
                RELEASE rc
                x.CloseChannel(c)
                RELEASE x,y,c
        ENDIF
        * если документ сжат
        IF doc_stat=5
                * разархивировать документ
                rc.DecompressDocument(doc_id)
                * проверка на ошибки
                er=rc.GetLastError()
                IF er<>""
                        ?er
                        RELEASE rc
                        x.CloseChannel(c)
                        RELEASE x,y,c
                ENDIF
        ENDIF
        * извлечь из документа "голые" данные, содержащие показания приборов
        doc64 = rc.GetRawData(doc_id)
        * проверка на ошибки
        IF er<>""
                ?er
                RELEASE rc
                x.CloseChannel(c)
                RELEASE x,y,c
        ENDIF
        * конвертация "голых" данных из base64 (на выходе имеем читаемый xml)
        doc = rc.ConvertFromBase64String(doc64)
        IF er<>""
                ?er
                RELEASE rc
                x.CloseChannel(c)
                RELEASE x,y,c
        ENDIF

        * сохранение xml в файл (не обязательно)
        *doc = '<?xml version="1.0" encoding="windows-1251"?>' + doc
        *gnFileHandle = FCREATE("путь к файлу")
        *=FWRITE(gnFileHandle, doc)
        *=FCLOSE(gnFileHandle)

        * обработка xml
        oXML = CreateObject("Microsoft.XMLDOM")
        oXML.async = .F.
        oXML.loadXML(doc)
        boolValue = oXML.hasChildNodes()
        IF boolValue = .F.
                RELEASE oXML
                RELEASE rc
                x.CloseChannel(c)
                RELEASE x,y,c
        ENDIF

        CREATE CURSOR devices(p_kod C(20),id C(50),name C(40),adr C(100),;
                d_kod C(20),d_type N(1,0),d_num C(20),tip_pok N(1,0),;
                prev_pok N(8,2),cur_pok N(8,2),cur_val N(8,2))
        root = oXML.selectSingleNode("root/rows")
        FOR i=0 TO root.childNodes.length-1
                potrKod=root.childNodes[i].selectSingleNode("kod").text
```

```
                        potrId=root.childNodes[i].selectSingleNode("id").text
                        potrName=root.childNodes[i].selectSingleNode("name").text
                        potrAdr=root.childNodes[i].selectSingleNode("adr").text

                        devs = root.childNodes[i].selectSingleNode("devices")
                        FOR j=0 TO devs.childNodes.length-1
                                dev = devs.childNodes[j]
                                devKod=dev.getAttributeNode("kod").text
                                devType=VAL(dev.getAttributeNode("type").text)
                                devNum=dev.getAttributeNode("num").text
                                FOR k=0 TO dev.childNodes.length-1
                                        pok = dev.childNodes[k]
                                        devPokType=VAL(pok.getAttributeNode("tip").text)
                                        prevPok=VAL(pok.getAttributeNode("prev").text)
                                        curPok=VAL(pok.getAttributeNode("cur").text)
                                        devVal=VAL(pok.getAttributeNode("val").text)

                                        INSERT INTO devices (p_kod, id, [name], adr, d_kod,
                                        d_type, d_num, tip_pok, prev_pok, cur_pok, cur_val) ;
                                        VALUES(potrKod, potrId, potrName, potrAdr, ;
                                        devKod, devType, devNum, devPokType, prevPok, ;
                                        curPok, devVal)
                                ENDFOR
                        ENDFOR
                ENDFOR
                BROWSE
        ENDFOR
ENDIF
* Очистка ресурсов
RELEASE rc
x.CloseChannel(c)
RELEASE x,y,c
```

5. Пример получения электронных документов, содержащих заявки на газопотребление, из базы данных АПК «КРЭДО», их распаковки и извлечения из них информации заявок (FoxPro 8+).

```
x = CREATEOBJECT("RRIC_COM_OBJECT.Invoker")
x.ClientCertName = "имя сертификата"
x.ServiceCertName = "Rric Wcf Service"
x.EndpointUrl = "http://www.rric.org:7742/RRIC_EP_SERVICE/WcfService"
x.MaxMessageSize = 1000000000
c = x.CreateChannel()

* Получить все документы для своей организации (на примере ООО "ТТГП")
y = x.ExecProc(c,"Kredo_DocumentsIn", "@comorg=100")
IF AT("error>", y) > 0
        * обработка ошибки
        a = AT("<error>",y,1)+7
        b = (LEN(y)-a)-(LEN(y)-AT("</error>",y,1))
        =MESSAGEBOX(SUBSTR(y, a, b))
ELSE
        * данные в курсор
        =XMLTOCURSOR(y, "acn")
        * извлечь первый документ с типом 21 (суточная заявка на газопотребление)
        SELECT TOP 1 doc_id, dstat FROM acn WHERE dtype=21 ORDER BY 1 INTO ARRAY dd
        * данные элемента массива в переменные для дальнейшего использования
        doc_id = ALLTRIM(dd[1,1])
        doc_stat = dd[1,2]
        * получить содержимое электронного документа
        y = x.ExecProc(c,"Kredo_DocumentData", "@doc_id="+doc_id)
        * данные в курсор
        =XMLTOCURSOR(y, "doc")
        * извлечь содержимое сжатого электронного документа
        SELECT doc_data FROM doc INTO ARRAY ddd
        doc_data = ALLTRIM(ddd[1,1])
        * подключаем крипто-модуль
        rc=CREATEOBJECT("RRIC_COM_OBJECT.Crypto")
        * загрузить электронный документ во временное хранилище крипто-модуля
        rc.UploadDocument(doc_data,doc_id)
        * проверка на ошибки
        er=rc.GetLastError()
        IF er<>""
                ?er
                RELEASE rc
                x.CloseChannel(c)
                RELEASE x,y,c
```

```
        ENDIF
        * если документ сжат
        IF doc stat=5
                * разархивировать документ
                rc.DecompressDocument(doc_id)
                * проверка на ошибки
                er=rc.GetLastError()
                IF er<>""
                        ?er
                        RELEASE rc
                        x.CloseChannel(c)
                        RELEASE x,y,c
                ENDIF
        ENDIF
        * извлечь из документа "голые" данные, содержащие заявку
        doc64 = rc.GetRawData(doc_id)
        * проверка на ошибки
        IF er<>""
                ?er
                RELEASE rc
                x.CloseChannel(c)
                RELEASE x,y,c
        ENDIF
        * конвертация "голых" данных из base64 (на выходе имеем читаемый xml)
        doc = rc.ConvertFromBase64String(doc64)
        IF er<>""
                ?er
                RELEASE rc
                x.CloseChannel(c)
                RELEASE x,y,c
        ENDIF

        * сохранение xml в файл (не обязательно)
        *doc = '<?xml version="1.0" encoding="windows-1251"?>' + doc
        *gnFileHandle = FCREATE("путь к файлу")
        *=FWRITE(gnFileHandle, doc)
        *=FCLOSE(gnFileHandle)

        * обработка xml
        oXML = CreateObject("Microsoft.XMLDOM")
        oXML.async = .F.
        oXML.loadXML(doc)
        boolValue = oXML.hasChildNodes()
        IF boolValue = .F.
                RELEASE oXML
                RELEASE rc
                x.CloseChannel(c)
                RELEASE x,y,c
        ENDIF

        req_date = oXML.selectSingleNode("root/ras_date").text

        CREATE CURSOR req([id] N(2,0),start C(5),end C(5),[val] N(8,0))
        rw = oXML.selectSingleNode("root/rows")
        FOR i=0 TO rw.childNodes.length-1
                row_id=VAL(rw.childNodes[i].getAttributeNode("id").text)
                start_time=rw.childNodes[i].getAttributeNode("start").text
                end_time=rw.childNodes[i].getAttributeNode("end").text
                row_val=VAL(rw.childNodes[i].getAttributeNode("value").text)

                INSERT INTO req([id],start,end,[val]) ;
                VALUES(row_id,start_time,end_time,row_val)
        ENDFOR
        BROWSE
ENDIF
* Очистка ресурсов
RELEASE rc
x.CloseChannel(c)
RELEASE x,y,c
```