Перв. применение

Справ. №

Программное обеспечение «Портал банка спецификаций по схемам трансграничного взаимодействия с электронной подписью с целью предоставления участникам ОСЖД возможности выбора и тестирования технических решений в области трансграничного взаимодействия»

ОБЩЕЕ ОПИСАНИЕ

Подп. и дата

Инв. № дубл.

Взам. инв. №

Подп. и дата

Инв. № подл.

		№ докум.	Подп.	Дата			
Разраб.					**Портал Банка Спецификаций**	Лит.	Лист / Листов
Пров.							
Н. контр.							
Утв.							

СОДЕРЖАНИЕ

1 Термины и определения

Термин	Определение
Электронная подпись	информация в электронной форме, которая присоединена к другой информации в электронной форме (подписываемой информации) или иным образом связана с такой информацией и которая используется для определения лица, подписывающего информацию
Удостоверяющий центр	юридическое лицо или индивидуальный предприниматель, осуществляющие функции по созданию и выдаче сертификатов ключей проверки электронных подписей, а также иные функции, предусмотренные законом
Доверенная третья сторона	организация, наделенная правом в соответствии с законодательством государства каждой из Сторон осуществлять деятельность по проверке электронной подписи в электронных документах в фиксированный момент времени в отношении составителя и (или) адресата электронного документа.

2 Назначение системы

1.1 Портал банка спецификаций трансграничного взаимодействия предназначен для агрегации существующих технологических и программных решений и нормативно-распорядительных документов с целью их использования при разработке, тестировании и эксплуатации систем взаимного признания электронной подписи для обеспечения юридической значимости трансграничного электронного документооборота между администрациями железных дорог – членами Организации сотрудничества железных дорог (ОСЖД) при организации международных грузовых железнодорожных перевозок.

1.2 Портал банка спецификаций трансграничного взаимодействия выполняет следующие функции:

1.2.1 Предоставление справочной, нормативной, организационно-распорядительной и технической информации по технологиям ДТС и используемым схемам трансграничного взаимодействия.

1.2.2 Онлайн-тестирование внешних служб ДТС.

1.2.3 Предоставление сервисов для разработки и тестирования компонентов информационных систем ДТС.

1.2.4 Предоставление API для внешних систем по разбору и проверке структур данных протоколов ДТС.

3 Описание системы

Портал Банка Спецификаций представляет из себя Web-портал, содержащий нормативно справочную информацию по тематике трансграничного взаимодействия, а также предоставляющий специализированные ресурсы для технических специалистов, позволяющие ускорить процессы разработки и тестирования собственных служб ДТС для подключения к инфраструктуре доверия ОАО «РЖД» и других участников ЭДО.

Портал Банка Спецификаций является закрытым ресурсом и для доступа к функционалу Портала необходима регистрация.

Авторизация на портале осуществляется по логину-паролю, выдаваемому организацией-владельцем Портала при регистрации.

3.1 Структура Портала банка спецификаций

Схема Портала банка спецификаций трансграничного взаимодействия представлена на Рисунок 3-1 и состоит из следующих компонентов:

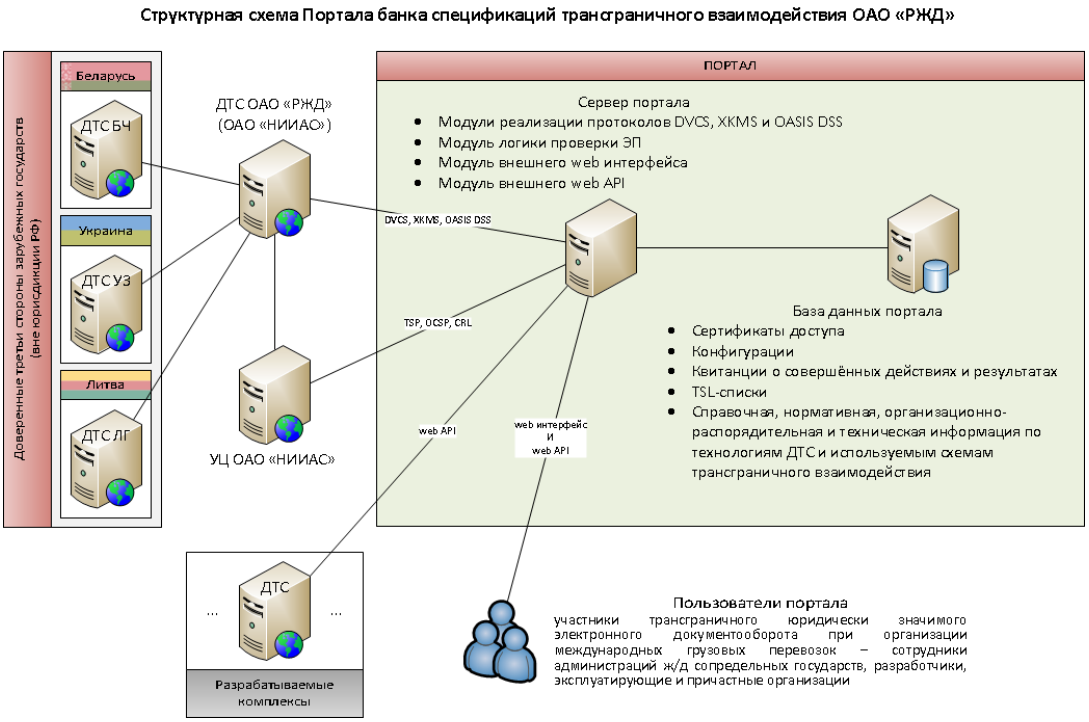

Рисунок 3-1 Структурная схема Портала Банка Спецификаций

1. База данных портала.
2. Сервер портала, состоящий из:
   - 2.1. Модуль реализации протокола DVCS.
   - 2.2. Модуль реализации протокола XKMS.
   - 2.3. Модуль реализации протокола OASIS DSS.

2.4. Модуль внешнего Web-интерфейса.

2.5. Модуль внешнего Web-API.

2.6. Модуль логики проверки ЭП.

3.1.1. База данных портала предназначена для хранения информации по схемам трансграничного взаимодействия, конфигурациям, сертификатам доступа, справочным, нормативным и организационно-распорядительным документам, TSL-спискам, квитанциям и другой информации, необходимой для работы пользователей и функционирования портала.

3.1.2. Сервер портала взаимодействует со всеми компонентами системы и предоставляет возможность выбора и тестирования технических решений в области трансграничного взаимодействия, выполняя функции заполнения базы данных портала, взаимодействия по протоколам и т.п.

Модуль реализации протокола DVCS предоставляет следующий функционал:

- формирование не подписанных VSD-запросов на проверку ЭП;
- подпись VSD-запросов;
- отправка VSD-запросов на тестовый DVCS-сервер;
- разбор, проверка и визуализация VSD-ответов.

Модуль реализации протокола XKMS предоставляет следующий функционал:

- формирование не подписанных XKMS-запросов на проверку СКПЭП;
- подпись XKMS-запросов;
- отправка XKMS-запросов на тестовый XKMS-сервер;
- разбор, проверка и визуализация XKMS-ответов.

Модуль реализации протокола OASIS DSS предоставляет следующий функционал:

- формирование не подписанных OASIS DSS-запросов на проверку СКПЭП и ЭП;
- подпись OASIS DSS-запросов;
- отправка OASIS DSS-запросов на тестовый OASIS DSS-сервер;
- разбор, проверка и визуализация OASIS DSS-ответов.

Модуль внешнего web-интерфейса обеспечивает взаимодействие пользователей с порталом через Web-интерфейс, в том числе обработку запросов, генерацию Web-страниц и отправку ответов.

Модуль внешнего Web-API предоставляет прикладной программный интерфейс по протоколу SOAP для программного взаимодействия со средствами портала.

Модуль логики проверки ЭП обеспечивает корректную проверку ЭП под проверяемыми документами в соответствии с зарегистрированными на сервере Портала криптографическими

провайдерами и корневыми цепочками доверенных Удостоверяющих центров.

3.2 Сведения по используемым спецификациям и алгоритмам

В работе Портала банка спецификаций используются следующие спецификации и алгоритмы:

3.2.1. При взаимодействии со службой DVCS:

- RFC 3029 – Internet X.509 Public Key Infrastructure Data Validation and Certification Server Protocols.

3.2.2. При взаимодействии со службой XKMS:

- W3C Recommendation - XML Key Management Specification Version 2.0 (XKMS 2.0) 28 June 2005.

3.2.3. При взаимодействии со службой OASIS DSS:

- OASIS Digital Signature Services (DSS) TC – Digital Signature Service CoreProtocols, Elements, and Bindings.

3.2.4. При подписи запросов:

- PKCS #7: Cryptographic Message Syntax.
- Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1.

3.2.5. При проверке сертификатов:

- RFC 2560 – X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP.
- RFC 5912 – New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX).
- RFC 6277 – Online Certificate Status Protocol Algorithm Agility.
- RFC 6960 – X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP.
- RFC 3161 – Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP).

3.3 Архитектура системы

Сервер портала содержит программные модули реализации протоколов DVCS, XKMS и OASIS DSS. Через них происходит взаимодействие внешних пользователей и систем через веб-интерфейс и веб-API с функциями портала.

Ключевые компоненты портала представлены на рисунке 3-2. Доступ через веб интерфейс осуществляется путём взаимодействия с html-страницами портала. Каждый из модулей реализует взаимодействие через страницу конструктора протокола, проверок подписи или сертификата и тестирования сторонней службы, реализующей данный протокол. Каждый из модулей осуществляет взаимодействие с базой данных для проверки прав пользователей на доступ к тому или иному функционалу. Страницы, содержащие документы и спецификации обращаются к базе данных для их получения.

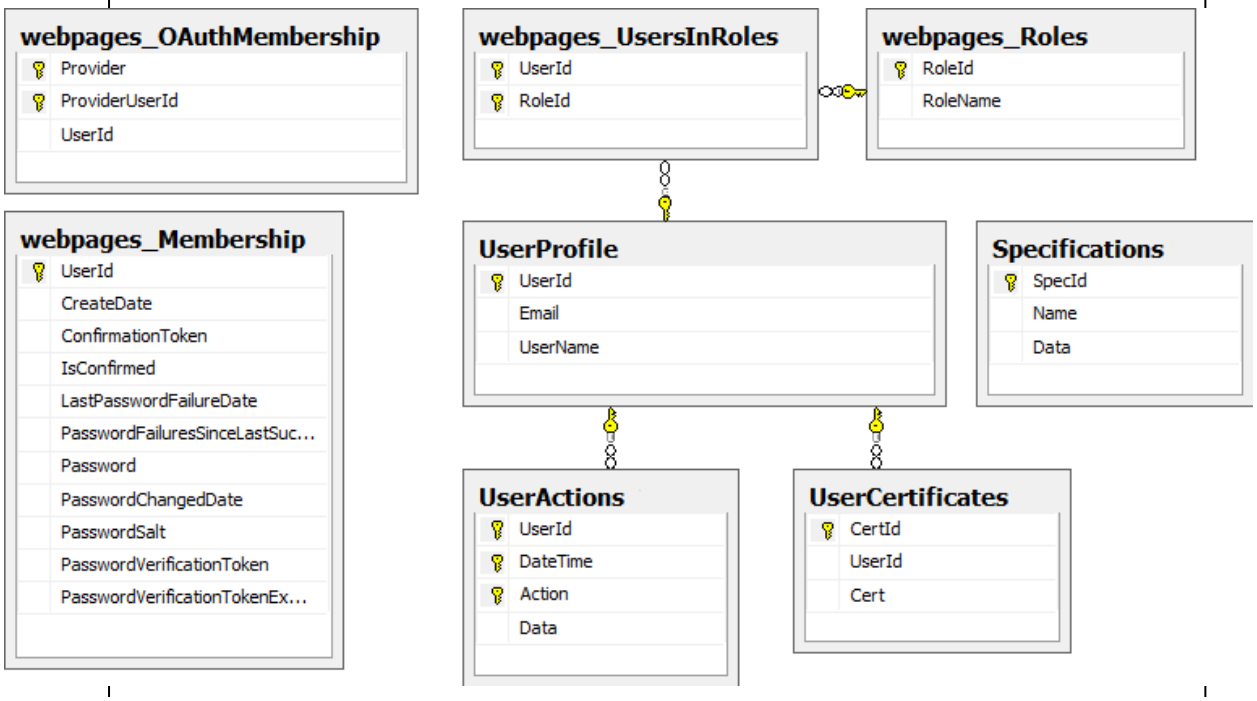

Рисунок 3-2. Схема базы данных портала

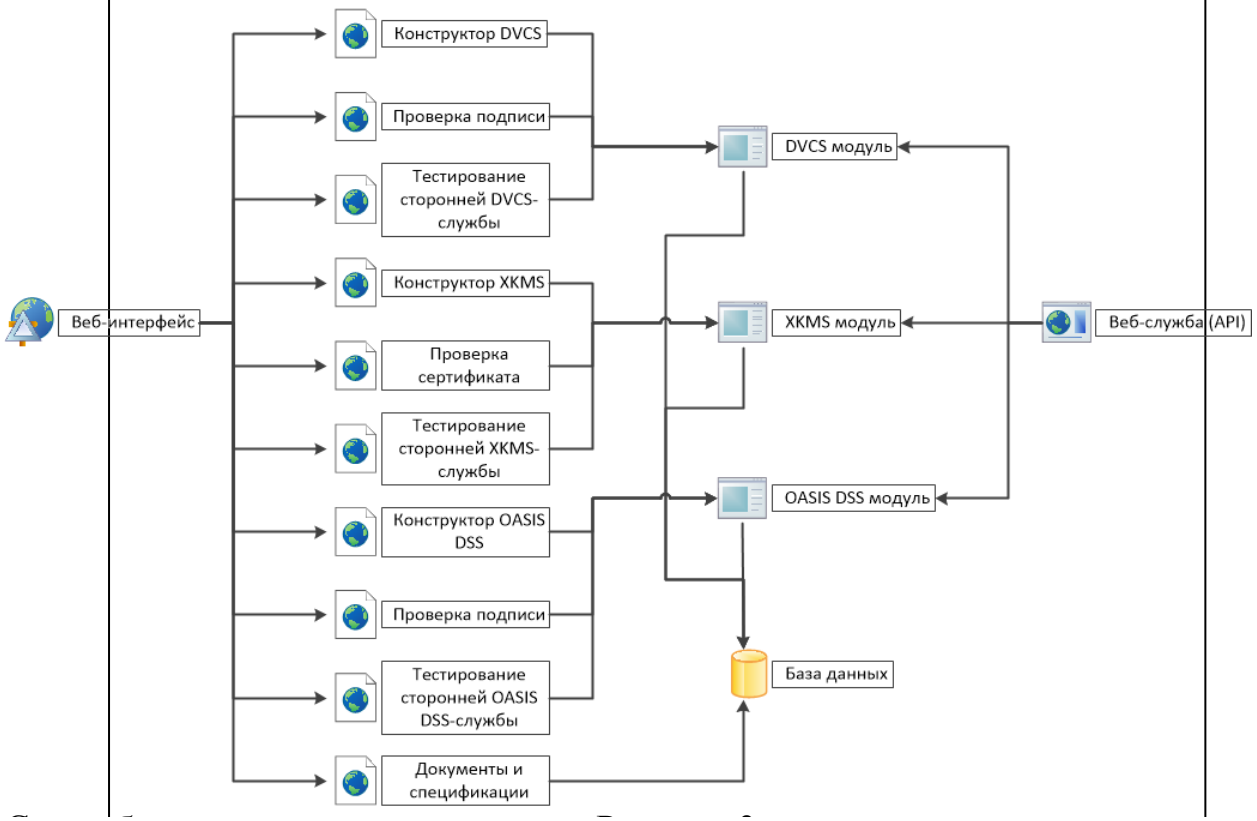

Рисунок 3-3 Ключевые компоненты портала

База данных портала хранит информацию о пользователях системы, в том числе логины, пароли, роли, информацию о сертификатах доступа, квитанции о совершённых действиях и результатах, справочная, нормативная, организационно-распорядительная и техническая информация по технологиям ДТС и используемым схемам трансграничного взаимодействия.

Таблицы, имеющие префикс webpages_ реализуют функционал SimpleMembership – модуля ASP.NET MVC 4, который отвечает за создание и управление ролями и пользователями. Это позволяет достаточно просто управлять пользователями и ролями с помощью класса WebSecurity. К примеру, для создания пользователя достаточно вызвать метод WebSecurity.addUser() и передать параметром имя создаваемого пользователя.

Таблица UserProfile содержит информацию о пользователях портала – уникальный идентификатор, адрес электронной почты и имя пользователя. Таблица UserCertificates хранит сертификаты доступа, ограничивающие возможность автоматизированного тестирования через API, таблица UserAction – квитанции и результаты действий пользователя (например, сформированный запрос для действия формирования запроса), таблица Specifications – документы и спецификации, использующиеся для отображения в соответствующих разделах портала.

3.4 DVCS модуль

Data Validation and Certification Server (DVCS) – это сервис инфраструктуры открытых ключей, предоставляющий:

- проверку данных;
- проверку правильность цифровой подписи документов;
- проверку действительности открытых ключей сертификатов;
- подтверждение о владении или о существовании данных.

Описание сервиса и протоколов взаимодействия содержится в RFC 3029 «Internet X.509 Public Key Infrastructure Data Validation and Certification Server Protocols».

DVCS – это доверенная третья сторона (ДТС, Trusted Third Party – TTP), которая может быть использована в качестве одного из компонентов при построении надёжных сервисов неотказуемости. В результате проверки, DVCS генерирует сертификат (свидетельство) проверки данных (Data Validation Certificate – DVC), который может быть использован для построения доказательств неотказуемости в отношении правильности и корректности утверждаемого лица к обладанию данными, правильность и достоверность открытого ключа лица, правильность и корректность цифровой подписи документа. Он не является заменой традиционным спискам отзыва (CRL) и протоколу получения статуса сертификата в реальном времени (Online Certificate Status Protocol – OCSP). DVCS следует скорее использовать для поддержки неотказуемости или в дополнение к более традиционным услугам, связанным с электронным

документооборотом. Наличие сертификата проверки данных поддерживает безотказность, предоставляя доказательства того, что цифровая подпись документа или сертификат открытого ключа были действительны на дату, указанную в DVC.

RFC3029 определяет 4 типа услуг проверки и подтверждения (validation and certification services):

- Certification of Possession of Data (cpd);
- Certification of Claim of Possession of Data (ccpd);
- Validation of Digitally Signed Document (vsd);
- Validation of Public Key Certificates (vpkc).

CPD – служба подтверждения владения данными свидетельствует, что запрашивающий обладал данными в указанное время и что настоящие данные были представлены серверу поверки данных.

CCPD – служба подтверждения претензии на владение данными аналогична предыдущей, за исключение того, что запрашивающий не предоставляет сами данные, а лишь значение хэш-функции.

VSD – служба проверка цифровой подписи документа, используется когда необходимо подтверждение действительности подписи документа. DVCS проверяет все подписи, присоединенные к подписанному документу, используя все необходимые сведения о состоянии и сертификаты открытых ключей. DVCS проверяет математическую корректность всех подписей, прикрепленных к документу, а также проверяет можно ли доверять подписывающим лицам, например, проверяется

полный путь сертификации от подписывающего лица к доверенной точки (например, DVCS's CA, или корневому CA в иерархии). DVCS будет выполнять проверку всех подписей, прикрепленных к подписанному документу. Провал проверки одной из подписей не обязательно приводит к отказу всей проверки, и наоборот, глобальная ошибка может произойти, если документ имеет недостаточное количество подписей.

VPKC – служба проверки сертификатов открытых ключей используется для проверки и утверждения достоверности (в соответствии с RFC 2459) одного или более сертификатов открытых ключей в указанное время. При проверке сертификата открытого ключа, DVCS подтверждает, что сертификат, включенный в запрос, – действительный сертификат, и определяет его статус отзыва в заданное время. DVCS проверяет полный путь сертификации от издателя сертификата до доверенной точки. Опять же, DVCS может полагаться на внешнюю информацию (CRL, OCSP, DVCS).

Портал Банка Спецификаций реализует VSD службу DVCS для проверки электронной подписи документа.

В соответствии с этим, модуль предоставляет три основных функциональных элемента:

- Проверка подписи по протоколу DVCS;
- Работу по протоколу DVCS с VSD службой в режиме конструктора
- Прикладной программный интерфейс для внешних систем (API)

Проверка подписи документа предполагает работу по следующему алгоритму:

1. Загрузка подписанного документа на сервер;
2. Формирование запроса со стандартными параметрами;
3. Отправка запроса на службу DVCS ДТС ОАО «РЖД» и получение ответа;
4. Разбор ответа и вывод результата проверки.

После загрузки сертификата на сервер осуществляется попытка разбора и преобразования в соответствующую программную структуру. Если попытка провалилась, то пользователю выводится соответствующее сообщение об ошибке.

При формировании запроса стандартными параметрами является включение в структуру requesterPolicy значения GeneralNames, содержащего Common Name сертификата подписанта.

Отправляемый запрос подписываемся ключом портала для обеспечения безопасности и исключения подмены данных, передаваемых на сервер. Если подпись запроса не пройдёт верификацию (проверку), то сервер вернёт ошибку.

Режим конструктора предполагает работу по следующему алгоритму:

1. Загрузка документа, подписанного документа, запроса, подписанного запроса или ответа на сервер;
2. Разбор, преобразование в программную структуру и переход на соответствующий шаг;

3. Подпись документа;
4. Формирование запроса;
5. Подпись запроса;
6. Отправка запроса и получение ответа.

Каждый из шагов 2-5 предполагает вывод пользователю текущего запроса/ответа и другой вспомогательной информации.

На первом шаге происходит загрузка файла выбранного типа на сервер, после чего осуществляется попытка разбора и преобразования в соответствующие программные структуры (второй шаг). Если попытка провалилась, то пользователю выводится соответствующее сообщение об ошибке, иначе происходит переход на соответствующий шаг.

На третьем шаге документ подписывается ключом портала и возвращается пользователю.

На четвёртом шаге для подписанного документа формируется запрос с необходимыми параметрами, который и возвращается пользователю.

На пятом шаге запрос подписывается ключом портала, подпись помещается в ASN.1 структуру запроса согласно стандарту PKCS-7.

На шестом шаге подписанный запрос отправляется на службу DVCS ДТС ОАО «РЖД», полученный ответ обрабатывается и выводится пользователю. Если во время отправки запроса или обработки запроса на ДТС возникли ошибки, то пользователю выводится соответствующее сообщение об ошибке.

API предоставляет доступ к функциям конструктора, а именно:

- Подпись документа
- Формирование запроса
- Подпись запроса
- Отправка запроса

Входными данными для всех функций является массив байт или base-64 строка, которые разбираются и преобразуются в соответствующие программные структуры.

Возвращаемым значением является ответ с результатом действия и сопутствующие сообщения (информационные, предупреждения, ошибки). Если действие не удалось, то пользователю выводится соответствующее сообщение об ошибке.

3.5 XKMS модуль

XML Key Management Specification (XKMS 2.0) – спецификация XML управления ключами – это документ, определяющий протоколы для распространения и регистрации открытых ключей, подходящие для использования в сочетании со стандартом для XML-подписи (XML Signature), определённым World Wide Web Consortium (W3C) и Internet Engineering Task Force (IETF), и сопутствующим стандартом XML-шифрования (XML Encryption). Основная цель разработки протокола – сведение к минимуму сложности приложений, использующих XML подпись. Став клиентом службы XKMS, приложение освобождается от сложностей и синтаксиса основной PKI, используемой, чтобы установить доверительные отношения, которые могут быть основаны на различных спецификациях, таких как X.509/PKIX, SPKI или PGP. Является Рекомендацией W3C и состоит из двух частей:

- XML Key Information Service Specification (X-KISS) – спецификация XML службы информации о ключах – описывает протокол делегации приложением к службе обработки ключевой информации, связанной с XML-подписью, XML-шифрованием, или с иным использованием элемента <ds:KeyInfo> XML-подписи;
- XML Key Registration Service Specification (X-KRSS) – спецификация XML службы регистрации ключей – описывает

протокол регистрации ключевой пары с целью дальнейшего её использования в сочетании с X-KISS или PKI (такой, как X.509).

Описываемые протоколы не требуют какой-либо конкретной базовой PKI (например, X509), но разработаны совместимыми с такими инфраструктурами.

Обмен данными по протоколу XKMS состоит из последовательности одной или двух пар запрос-ответ. Сообщения протокола используют общий формат, что позволяет транслировать их через различные протоколы. В целях совместимости рекомендуется реализация с поддержкой SOAP через HTTP.

Основная цель X-KISS – помочь разработчикам приложений получить информацию, связанную с криптографическим ключом, присоединённым к подписанному и/или зашифрованному XML документу.

X-KISS описывает три уровня (tier) взаимодействия –. <ds:RetrievalMethod> Processing (tier 0), Locate Service (tier 1) и Validate Service (tier 2).

Для <ds:RetrievalMethod> Processing , основной элемент которого включает в себя <ds:KeyInfo>, доверенная служба не требуется, т.к. это средство передачи информации, доступной из удаленного местоположения. Например, лицо, подписавшее документ, захочет указать проверяющих (verifiers) для цепочки сертификатов без прикрепления их к документу. В этом случае, элемент <ds:RetrievalMethod> будет состоять из ссылки на

местоположение в сети, из которой цепочка сертификатов может быть восстановлена.

Locate Service разрешает (определяет) элемент <ds:KeyInfo>, но от службы не требуется утверждение достоверности связывания (validity of the binding) между данными в <ds:KeyInfo>. Примером использования службы может быть необходимость получения требуемых параметров шифрования открытого ключа получателя при отправке ему зашифрованного XML документа.

Validate Service делает всё то же, что и Locate Service, и, кроме того, клиент может получить утверждение, определяющее статус связывание между открытым ключом и другими данных, например, именем или набором расширенных атрибутов. Более того, служба заявляет, что статус каждого из элементов возвращаемых данных действует, и что все они связаны с одним и тем же открытым ключом. Клиент посылает службе прототип, содержащий некоторые или все из элементов, для которых требуется привязка статуса доверия. Если информация в прототипе является неполной, служба может получить необходимые дополнительные данные от базовой службы PKI. После того, как правильность привязки к ключу была определена, сервис возвращает статуса результата клиенту.

При запросе информация о сертификате ключа помещается в элемент <X509Certificate>, вложенный в элемент <X509Data>, помещаемый в элемент <QueryKeyBinding>. Кроме того, в элемент <KeyUsage> запроса помещается информация об одной

или более целей использования (всего их три – шифрование, подпись и обмен).

Портал Банка Спецификаций реализует Validate Service согласно X-KISS для проверки сертификата электронной подписи.

В соответствии с этим, модуль предоставляет три основных функциональных элемента:

- Проверка сертификата по протоколу XKMS;
- Работу по протоколу XKMS с Validate Service в режиме конструктора
- API

Проверка сертификата предполагает работу по следующему алгоритму:

5. Загрузка сертификата на сервер;
6. Формирование запроса со стандартными параметрами;
7. Отправка запроса на службу XKMS ДТС ОАО «РЖД» и получение ответа;
8. Разбор ответа и вывод результата проверки.

После загрузки сертификата на сервер осуществляется попытка разбора и преобразования в соответствующую программную структуру. Если попытка провалилась, то пользователю выводится соответствующее сообщение об ошибке.

При формировании запроса стандартными параметрами является включение элемента <RespondWith> со значением

X509CRL (http://www.w3.org/2002/03/xkms#X509CRL) – указанием на возврат информации о проверке по спискам отзыва сертификатов в XKMS-ответе.

Отправляемый запрос подписываемся ключом портала для обеспечения безопасности и исключения подмены данных, передаваемых на сервер. Если подпись запроса не пройдёт верификацию (проверку), то сервер вернёт ошибку.

Режим конструктора предполагает работу по следующему алгоритму:

1. Загрузка сертификата, запроса, подписанного запроса или ответа на сервер;
2. Разбор, преобразование в программную структуру и переход на соответствующий шаг;
3. Формирование запроса;
4. Валидация запроса;
5. Подпись запроса;
6. Отправка запроса и получение ответа.

Каждый из шагов 2-6 предполагает вывод пользователю текущего запроса/ответа и другой вспомогательной информации.

На первом шаге происходит загрузка файла выбранного типа на сервер, после чего осуществляется попытка разбора и преобразования в соответствующие программные структуры (второй шаг). Если попытка провалилась, то пользователю выводится соответствующее сообщение об ошибке, иначе происходит переход на соответствующий шаг.

На третьем шаге для сертификата формируется запрос с необходимыми параметрами, результатом является xml-документ, который и возвращается пользователю.

На четвёртом шаге происходит валидация (проверка на соответствие xsd-схеме) xml-документа запроса. Если документ не соответствует схеме, то пользователю выводится соответствующее сообщение об ошибке.

На пятом шаге запрос подписывается ключом портала, подпись помещается в xml-документ в тело запроса как элемент <ds:Signature>.

На шестом шаге подписанный запрос отправляется на службу XKMS ДТС ОАО «РЖД», полученный ответ обрабатывается и выводится пользователю. Если во время отправки запроса или обработки запроса на ДТС возникли ошибки, то пользователю выводится соответствующее сообщение об ошибке.

API предоставляет доступ к функциям конструктора, а именно:

Формирование запроса

Валидация запроса

Подпись запроса

Отправка запроса

Входными данными для всех функций является массив байт или base-64 строка, которые разбираются и преобразуются в соответствующие программные структуры.

Возвращаемым значением является ответ с результатом действия и сопутствующие сообщения (информационные, предупреждения, ошибки). Если действие не удалось, то пользователю выводится соответствующее сообщение об ошибке.

3.6 OASIS DSS модуль

Organization for the Advancement of Structured Information Standards (OASIS) – организация по развитию стандартов структурированной информации является глобальным консорциумом, который работает над разработкой, сближением и принятием стандартов в области электронного бизнеса и веб служб.

Digital Signature Services (DSS) – OASIS стандарт. Его спецификация описывает два XML-based протокола типа запрос-ответ – протокол подписи и протокол верификации. С помощью этих протоколов клиент может отправить документ на сервер и получить обратно подпись на документ; или отправить документ и подпись серверу и получить обратно ответ с результатом проверки подписи документа

Стандарт состоит из основного модуля (DSS Core) и возможных профилей (DSS Profiles). Основной модуль предоставляет базовый протокол и элементы, которые приспособлены для поддержки конкретных случаев использования в профилях. Протокол подписи содержит два основных элемента, отображающих параметры запроса и ответа в XML – <SignRequest> и <SignResponse>.

Элемент <SignRequest>:

- Может содержать атрибут RequestID, соотносящий запрос с ответом. Если он есть в запросе, то сервер обязан вернуть его в ответе.

- Может содержать атрибут Profile, указывающий конкретный профиль DSS. Он может быть использован, чтобы выбрать профиль, если сервер поддерживает несколько профилей, или как проверка, что сервер реализует профиль, запрашиваемый клиентом.
- Может содержать элемент <OptionalInputs>, который содержит дополнительные данные, добавленные к запросу. Используется в профилях.
- Обязан содержать элемент <InputDocuments>, в котором содержится документ для подписи.

Элемент <SignResponse>:

- Может содержать атрибут RequestID, соотносящий запрос с ответом, если он есть в запросе.
- Может содержать атрибут Profile, указывающий конкретный профиль DSS, используемый сервером. Он может быть использован клиентом для целей ведения журнала или чтобы убедиться, что сервер реализует профиль, который ожидает клиент.
- Обязан содержать элемент <Result> — код, представляющий состояние запроса.
- Может содержать элемент <OptionalOutputs>, который содержит дополнительные данные, добавленные к ответу. Используется в профилях.
- Может содержать элемент <SignatureObject>, который содержит результат подписи или временной штамп, или, в случае, когда подпись содержится в выходном документе (<OutputDocument>

– один из элементов в <OptionalOutputs>), указатель на подпись (используется в профилях).

При запросе на подпись в запрос помещаются необходимые документы и дополнительная информация (например, идентификатор запроса), после чего запрос отправляется на сервер. Сервер подписывает документы и отправляет результаты работы обратно клиенту. Сами запросы при этом подписываются сертификатом открытого ключа клиента.

Протокол проверки (как и протокол подписи) содержит два основных элемента запроса и ответа – <VerifyRequest> и <VerifyResponse>. Элемент <VerifyRequest>, как и <SignRequest>, может содержать атрибуты RequestID и Profile, и элементы <OptionalInputs> и <InputDocuments> (причём последний не является обязательным). Дополнительным элементом является <SignatureObject>, в случае отсутствия которого налагается условие наличия одного документа в <InputDocuments>, в котором сервер будет искать подпись для проверки.

Элемент <VerifyResponse>, как и <SignResponse>, может содержать атрибуты RequestID и Profile, и элементы <OptionalOutputs> и <InputDocuments>, а обязательно должен содержать элемент <Result>.

Портал Банка Спецификаций реализует сервис для проверки электронной подписи документа по протоколу OASIS DSS.

В соответствии с этим, модуль предоставляет три основных функциональных элемента:

- Проверка электронной подписи документа по протоколу OASIS DSS;
- Работу по протоколу OASIS DSS со службой проверки ЭП в режиме конструктора;
- API.

Проверка ЭП предполагает работу по следующему алгоритму:

1. Загрузка подписанного документа на сервер;
2. Формирование запроса со стандартными параметрами;
3. Отправка запроса на службу OASIS DSS ДТС ОАО «РЖД» и получение ответа;
4. Разбор ответа и вывод результата проверки.

После загрузки сертификата на сервер осуществляется попытка разбора и преобразования в соответствующую программную структуру. Если попытка провалилась, то пользователю выводится соответствующее сообщение об ошибке.

Отправляемый запрос подписываемся ключом портала для обеспечения безопасности и исключения подмены данных, передаваемых на сервер. Если подпись запроса не пройдёт верификацию (проверку), то сервер вернёт ошибку.

Режим конструктора предполагает работу по следующему алгоритму:

1. Загрузка документа, подписанного документа, запроса, подписанного запроса или ответа на сервер;

2. Разбор, преобразование в программную структуру и переход на соответствующий шаг;
3. Формирование запроса;
4. Валидация запроса;
5. Подпись запроса;
6. Отправка запроса и получение ответа.

Каждый из шагов 2-6 предполагает вывод пользователю текущего запроса/ответа и другой вспомогательной информации.

На первом шаге происходит загрузка файла выбранного типа на сервер, после чего осуществляется попытка разбора и преобразования в соответствующие программные структуры (второй шаг). Если попытка провалилась, то пользователю выводится соответствующее сообщение об ошибке, иначе происходит переход на соответствующий шаг.

На третьем шаге для подписанного документа формируется запрос с необходимыми параметрами, результатом является xml-документ, который и возвращается пользователю.

На четвёртом шаге происходит валидация (проверка на соответствие xsd-схеме) xml-документа запроса. Если документ не соответствует схеме, то пользователю выводится соответствующее сообщение об ошибке.

На пятом шаге запрос подписывается ключом портала, подпись помещается в xml-документ в тело запроса как элемент <ds:Signature>.

На шестом шаге подписанный запрос отправляется на службу OASIS DSS ДТС ОАО «РЖД», полученный ответ

обрабатывается и выводится пользователю. Если во время отправки запроса или обработки запроса на ДТС возникли ошибки, то пользователю выводится соответствующее сообщение об ошибке.

API предоставляет доступ к функциям конструктора, а именно:

- Подпись документа;
- Формирование запроса;
- Валидация запроса;
- Подпись запроса;
- Отправка запроса.

Входными данными для всех функций является массив байт или base-64 строка, которые разбираются и преобразуются в соответствующие программные структуры.

Возвращаемым значением является ответ с результатом действия и сопутствующие сообщения (информационные, предупреждения, ошибки). Если действие не удалось, то пользователю выводится соответствующее сообщение об ошибке.

3.7 Программная реализация

3.7.1. Программная платформа

К программной платформе для Портала банка спецификаций предъявлялись следующие функциональные требования:

- работа с ASN.1 структурами данных;
- работа с XML структурами данных;
- работа с X.509 сертификатами, открытыми и закрытыми ключами, доступ к криптографическим функциям;
- поддержка веб-служб, в частности с реализацией взаимодействия по протоколу SOAP;
- возможность создания динамических веб-страниц;
- поддержка разработки внешних модулей и библиотек;
- работа с БД.

Данным требованиям удовлетворяют различные библиотеки и фреймворки, написанные на Java, C#, Python и др. Однако, в условиях существующих стандартов разработки в организации (.Net Framework) программная платформа получилась следующей:

- ASP.NET MVC 4 Framework
- Библиотека BouncyCastle
- База данных Microsoft SQL Server 2008 R2

3.7.2. Программные модули

В данном разделе описана реализация модулей портала.

3.7.3. DVCS

DVCS модуль портала базируется на классах, реализующих сами структуры протокола DVCS и модель взаимодействия пользователей с порталом.

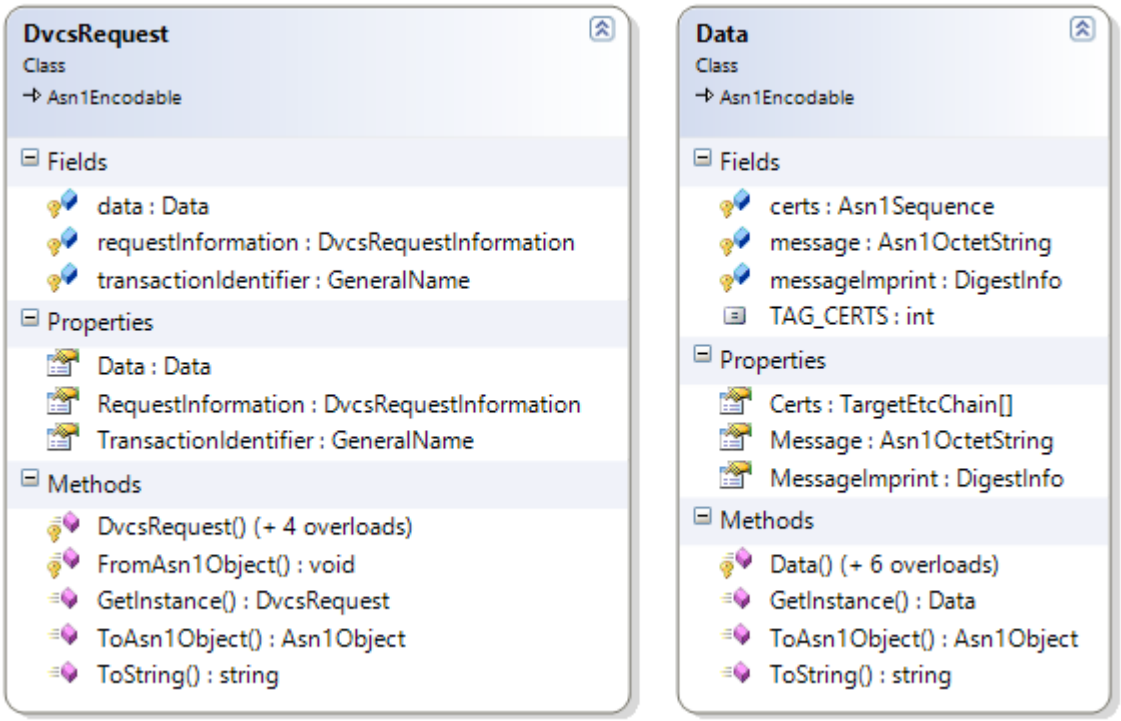

Рисунок 3-2. Классы DvcsRequest (слева) и Data (справа)

Иерархия классов повторяет ASN.1 структуры из спецификации. Так, например, класс DvcsRequest (Рисунок 3-2, слева) наследуется от класса Asn1Encodable и имеет три защищённых (protected) поля:

- requestInformation типа DvcsRequestInformation;
- data типа Data;
- transactionIdentirier типа X509.GeneralName.

Это соответствует следующей ASN.1 структуре:

DVCSRequest ::= SEQUENCE {

```
requestInformation DVCSRequestInformation,
data               Data,
transactionIdentifier  GeneralName OPTIONAL
}
```

А класс Data (Рисунок 3-2, справа), так же наследуемый от класса Asn1Encodable и имеющий поля:

- message типа Asn1OctetString;
- messageImprint типа DigestInfo;
- certs типа Asn1Sequence

соответствует структуре

```
Data ::= CHOICE {
        message          OCTET STRING ,
        messageImprint   DigestInfo,
        certs            SEQUENCE SIZE (1..MAX) OF
TargetEtcChain
}
```

Поскольку формирование запроса происходит к службе VSD, то в экземпляре класса Data заполняется только поле message, куда в виде Asn1OctetString, имеющего конструктор для массива байт, помещается подписанный документ. Другие поля для данного типа запроса не используются, но реализованы в соответствии со спецификацией.

Методы классов предоставляют конструкторы с различными сигнатурами. Метод ToAsn1Object() наследуется от родительского класса и является переопределённым, метод GetInstance() хоть и не является обязательным, но его реализация для наследников Asn1Object (базовый класс всех Asn1 объектов) является хорошим тоном и часто используется.

Класс DvcsRequestInformation (Рисунок 3-3), наследуемый от класса Asn1Encodable, содержит информацию о текущем запросе, в том числе версию (по умолчанию 1), тип запрашиваемой службы (VSD) и опциональное поля: nonce (одноразовый код, выбранный случайным образом и используемый для безопасной передачи основного пароля, предотвращая атаку повторного воспроизведения), общие имена (GeneralNames) отправителя запроса, политики отправителя (по умолчанию в качестве политик указывается CommonName сертификата отправителя), время формирования запроса.

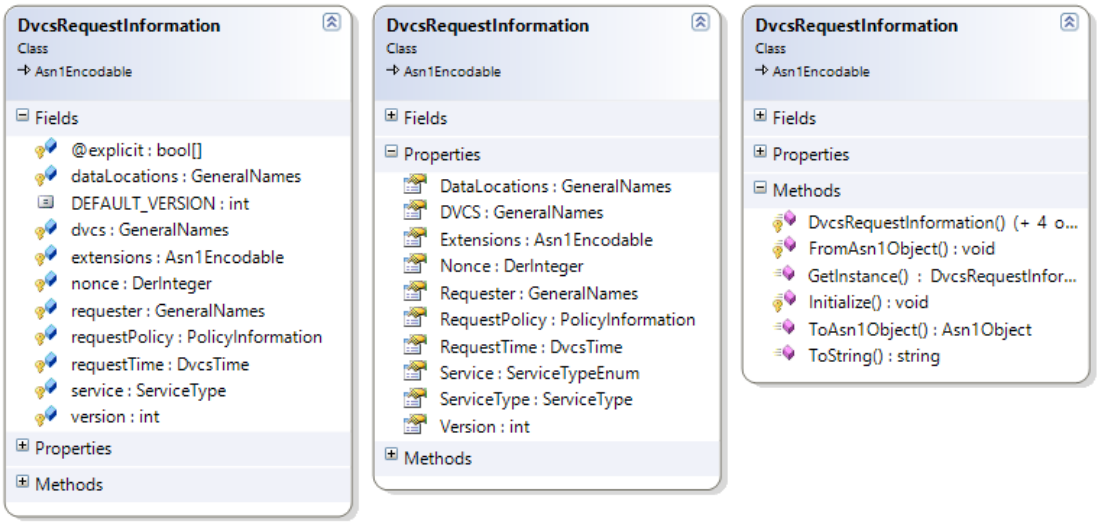

Рисунок 3-3. Класс DvcsRequestInformation

Из методов стоит отметить Initialize(), который создаёт экземпляр со значениями по умолчанию, например, записывает текущее значение времени, генерирует nonce.

Время формирования запроса отражено в классе DvcsTime (Рисунок 3-4, слева), основная особенность которого – возможность сохранять временной штамп.

Класс DvcsResponse (Рисунок 3-4, справа) отражает структуру ответа, наследуется от Asn1Encodable, и содержит два поля: dvCertInfo типа DvcsCertInfo и dvErrorNote типа DvcsErrorNotice.

Класс DvcsCertInfo содержит в себе информацию о полученном запросе (поле dvReqInfo типа DvcsRequestInformation, поле reqSignature, содержащее информацию о подписях в запросе, и поле messageImprint, содержащее подписанные данные из поля Data запроса), результатах проверки (поле dvStatus) и информации о самом ответе (версия, время ответа, серийный номер).

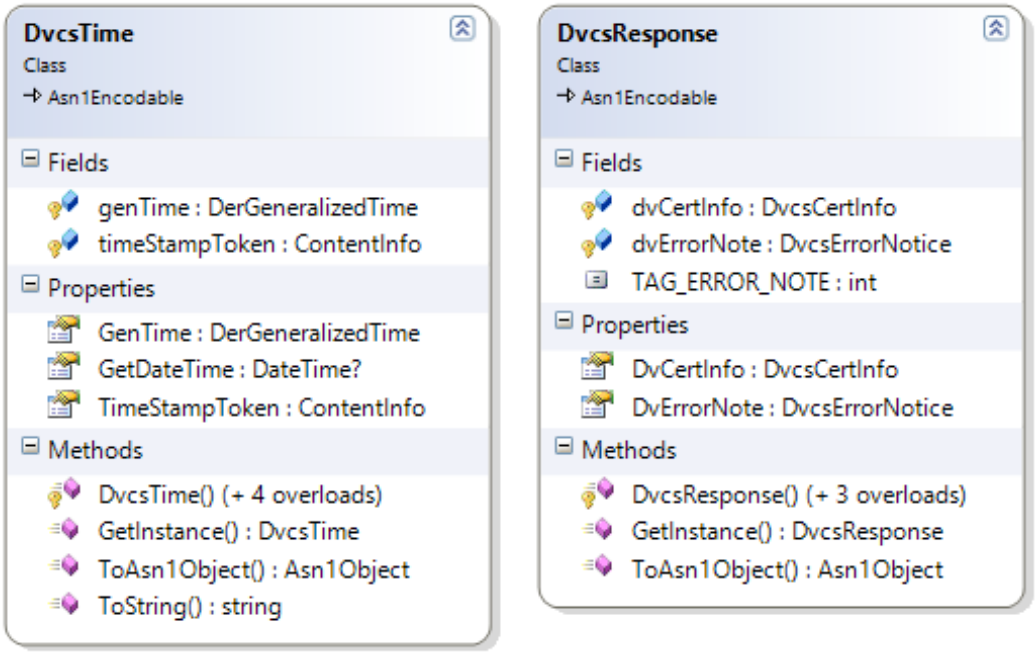

Рисунок 3-4. Классы DvcsTime (слева) и DvcsResponse (справа)

Класс DvcsErrorNotice содержит информацию о критической ошибке, произошедшей во время обработки запроса (например, не получилось разобрать запрос, не те объектные идентификаторы и т.д.) и состоит из двух полей: идентификатор транзакции, который соответствует идентификатору транзакции из запроса, и статус транзакции, который содержит сведения об ошибке.

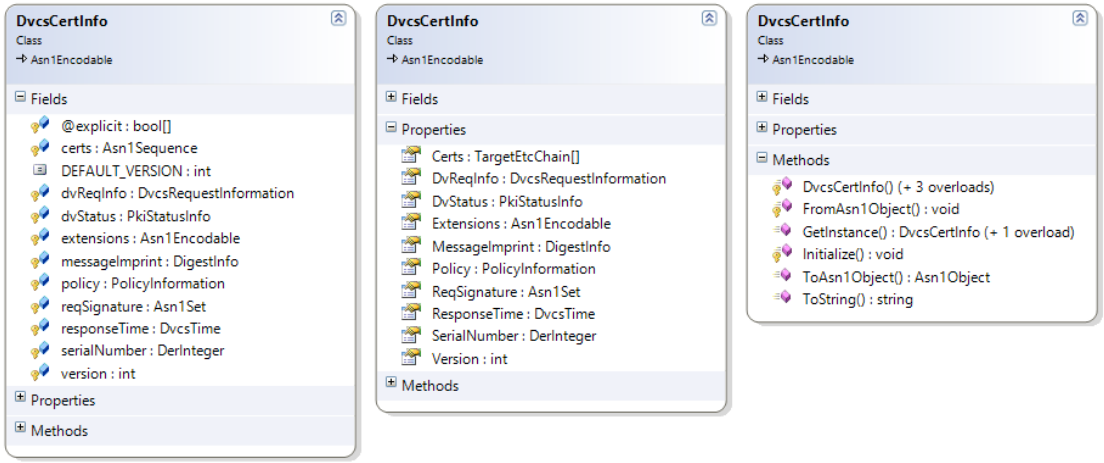

Рисунок 3-5. Класс DvcsCertInfo

3.7.4.XKMS

XKMS модуль портала базируется на классах, реализующих сами структуры протокола XKMS и модель взаимодействия пользователей с порталом.

Класс ValidateRequest (Рисунок 3-6, слева) реализует в себе функционал формирования запроса, его подписи и отправки. Поле x509cert содержит сертификат для подписи запроса, serviceAddress – адрес, на который будет отправлен запрос, isSigned – булева переменная, показывающая подписан запрос

или нет, xmlDoc – xml-документ, содержащий запрос, reqId – уникальный идентификатор запроса, respondWith – список, содержащий значения URI, в соответствии с которыми будет формироваться ответ, timeInstant – время формирования запроса, x509certs – список сертификатов, отправляемых на проверку.

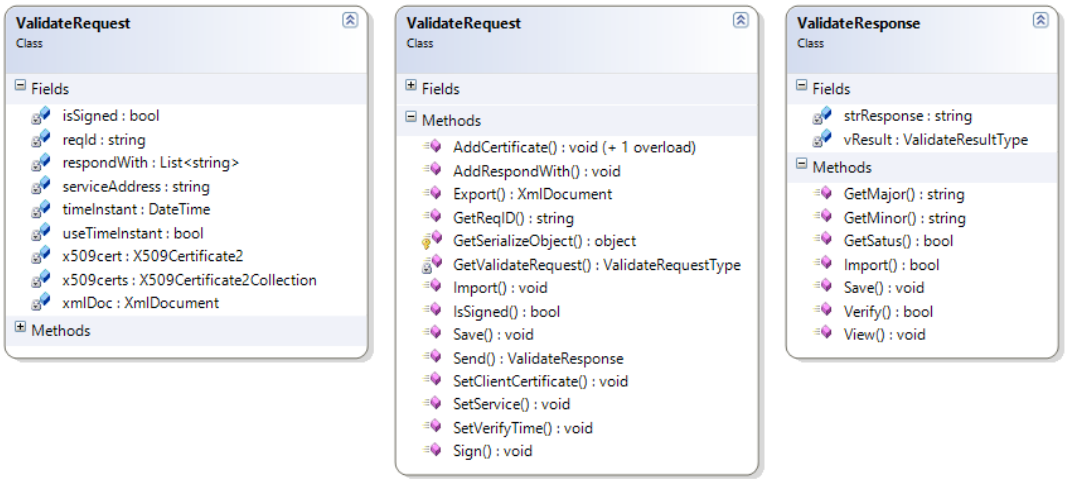

Рисунок 3-6. Классы ValidateRequest (слева) и ValidateResponse (справа)

Методы Import() и Export() позволяют импортировать или экспортировать текущий запрос. При импорте принимается xml-строка, которая десериализуется в струку ValidateRequestType, которая, вместе с такими классами, как QueryKeyBindingType, TimeInstantType, и другими отражают структуру запроса, используются для сериализации-десериализации, но никаким другим функционалом не обладают, поэтому их описание приведено не будет. При экспортировании возвращается xml-документ, содержащий запрос, типа XmlDocument из пространства имён System.Xml.

Метод AddCertificate() добавляет сертификат в коллекцию x509certs для проверки. Входным значением может быть как отпечаток (thumbprint) сертификата, так и экземпляр класса X509Certificate2 пространства имён System.Security.Cryptography.X509Certificates. При передаче значения отпечатка, поиск выполняется в хранилище личное (StroreName.My) сертификатов пользователя (StoreLocation.CurrentUser).

Метод AddRespondWith() добавляет одноимённый параметр к запросу. Сам параметр является uri, стандартные значения которого описаны в классе Constants, среди которых как обычные xkms-константы (RESPOND_WITH_OCSP, RESPOND_WITH_VALIDATION_DETAILS), так и константы из расширения для работы с Pan-European Public Procurement Online – PEPPOL (RESPOND_WITH_PEPPOL_EXTENDED).

Метод GetReqID() возвращает значение уникального идентификатора текущего запроса. Методы GetSerializeObject() и GetValidateRequest() испольсуются при сериализации и десириализации.

Метод SetClientCertificate() принимает экземпляр класса X509Certificate2 и устанавливает его в качестве сертификата для подписи (поле x509cert). При этом делается проверка на наличие закрытого ключа.

Метод SetService() принимает в качестве параметра строку с адресом XKMS службы, на которую будет отправлен запрос, а метод SetTimeInstant() принимает экземпляр класса DateTime

пространства имён System и устанавливает его для поля timeInstant.

Метод Sign() подписывает запрос закрытым ключом сертификата, указанного в x509cert. При этом выполняется так называемая канонизация xml-документа, создание подписи, её проверка и помещение в запрос в качестве элемента <ds:Signature>.

Метод Send() производит отправку подписанного запроса на адрес службы, указанный в serviceAddress. Полученный ответ преобразуется к классу ValidateResponse, экземпляр которого и возвращает метод.

Класс ValidateResponse (Рисунок 3-6, справа) содержит два поля: ответ, преобразованный в xml-строку strResponse типа string и vResult типаValidateResultType, являющийся результатом десериализации этой строки. Кроме того, содержит метод Import(), принимающий массив байт, который преобразуется, разбирается, а результат заполняет поля экземпляра.

Методы GetStatus(), GetMajor() и GetMinor() позволяют получить статус ответа, и результат проверки. Результат состоит из двух значений – старшего кода (ResultMajor) и дополнительного младшего кода (ResultMinor). Старшие и младшие коды выражаются через XML-тип anyURI. Спецификация использует нотацию ResultMajor.ResultMinor, чтобы указать код результата. Например, Sender.Failure означает, что причина ошибки относится к отправителю (например, запрос не прошёл проверку на соответствие схеме). Классы ResultMajor

и ResultMinor являются перечислениями (enum), содержащими стандартные uri в соответствии со спецификацией.

3.7.5. OASIS DSS

OASIS DSS модуль портала базируется на классах, реализующих сами структуры протокола OASIS DSS и модель взаимодействия пользователей с порталом.

Класс VerifyRequest (Рисунок 3-7, слева) реализует в себе функционал формирования запроса, его подписи и отправки. Поле x509cert содержит сертификат для подписи запроса, serviceAddress – адрес, на который будет отправлен запрос, isSigned – булева переменная, показывающая подписан запрос или нет, xmlDoc – xml-документ, содержащий запрос, reqId – уникальный идентификатор запроса, docList – список подписанных документов, отправляемых на проверку.

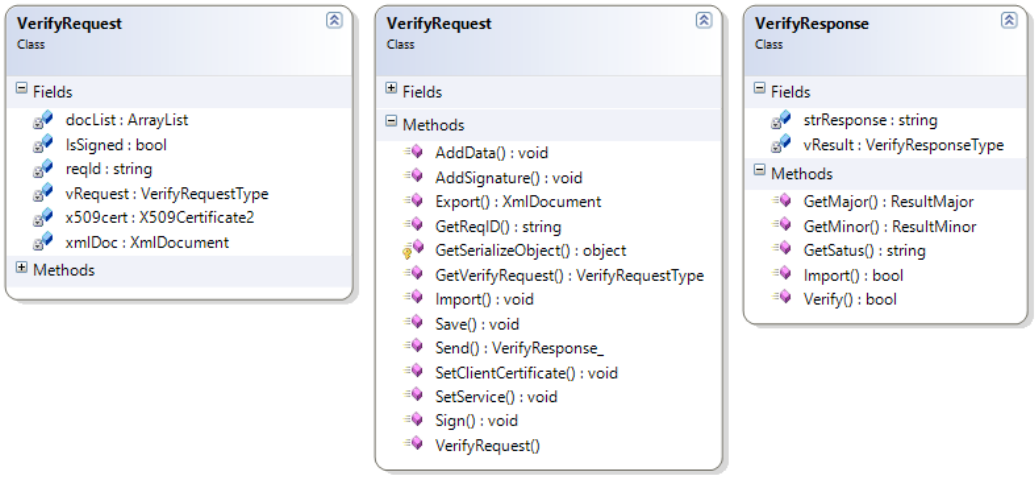

Рисунок 3-7. Классы VerifyRequest (слева) и VerifyResponse (справа)

Методы Import() и Export() позволяют импортировать или экспортировать текущий запрос. При импорте принимается xml-

строка, которая десериализуется в струку VerifyRequestType, которая, вместе с такими классами, как AnyType, InputDocuments и SignatureObject отражают структуру запроса, используются для сериализации-десериализации. При экспортировании возвращается xml-документ, содержащий запрос, типа XmlDocument из пространства имён System.Xml.

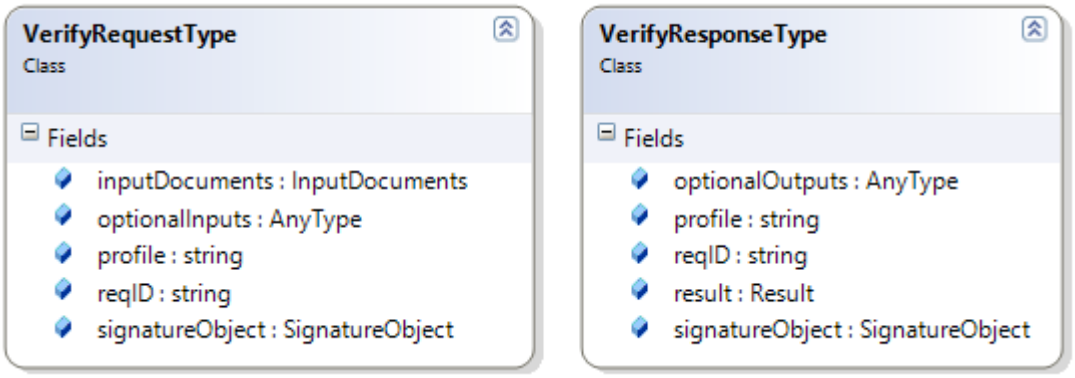

Рисунок 8. Классы VerifyRequestType (слева) и VerifyResponseType (справа)

Класс VerifyRequestType (Рисунок 8, слева) содержит поле optionalInputs типа AnyType, inputDocuments типа InputDocuments, reqID типа string и profile типа string и signatureObject типа SignatureObject. Данные поля полностью соответствуют полям запроса в соответствии со спецификацией.

Метод AddData() добавляет подписанный документ в коллекцию docList для проверки. Входным значением может быть как массив байт (byte[]), так и строка, содержащая путь к файлу.

Метод GetReqID() возвращает значение уникального идентификатора текущего запроса. Методы GetSerializeObject() и

GetVerifyRequest() используются при сериализации и десириализации.

Метод SetClientCertificate() принимает экземпляр класса X509Certificate2 и устанавливает его в качестве сертификата для подписи (поле x509cert). При этом делается проверка на наличие закрытого ключа.

Метод SetService() принимает в качестве параметра строку с адресом OASIS DSS службы, на которую будет отправлен запрос.

Метод Sign() подписывает запрос закрытым ключом сертификата, указанного в x509cert. При этом выполняется так называемая канонизация xml-документа, создание подписи, её проверка и помещение в запрос в качестве элемента <ds:Signature>.

Метод Send() производит отправку подписанного запроса на адрес службы, указанный в serviceAddress. Полученный ответ преобразуется к классу VerifyResponse, экземпляр которого и возвращает метод.

Класс VerifyResponse (Рисунок 3-7, справа) содержит два поля: ответ, преобразованный в xml-строку strResponse типа string и vResult типа VerifyResponseType (Рисунок 8, справа), являющийся результатом десериализации этой строки. Кроме того, содержит метод Import(), принимающий массив байт, который преобразуется, разбирается, а результат заполняет поля экземпляра.

Метод GetStatus(), GetMajor() и GetMinor() позволяют получить статус ответа, и результат проверки. Результат состоит

из двух значений – старшего кода (ResultMajor) и дополнительного младшего кода (ResultMinor). Старшие и младшие коды выражаются через XML-тип anyURI. Спецификация использует нотацию ResultMajor.ResultMinor, чтобы указать код результата. Например, Success.ValidSignatureOnAllDocuments означает, что проверка подписи всех подписанных документов прошла успешно. Классы ResultMajor и ResultMinor являются перечислениями (enum), содержащими стандартные uri в соответствии со спецификацией.

4 Взаимодействие Портала Банка Спецификаций с другими системами

Портал Банка Спецификаций в процессе функционирования взаимодействует с ДТС РЖД ОАО «НИИАС» по протоколам DVCS, XKMS и OASIS DSS.

4.1. Взаимодействие со службой DVCS.

4.1.1. Формирование не подписанных VSD-запросов на проверку ЭП/ЭЦП.

4.1.2. Подпись VSD-запросов.

4.1.3. Отправка VSD-запросов на тестовый DVCS-сервер.

4.1.4. Разбор, проверка и визуализация VSD-ответов.

4.2. Взаимодействие со службой XKMS.

4.2.1. Формирование не подписанных XKMS-запросов (единичных и комплексных) на проверку сертификатов.

4.2.2. Подпись XKMS-запросов.

4.2.3. Отправка XKMS-запросов на тестовый XKMS-сервер.

4.2.4. Разбор, проверка и визуализация XKMS-ответов.

4.3. Взаимодействие со службой OASIS DSS.

4.3.1. Формирование не подписанных OASIS DSS-запросов (единичных и комплексных) на проверку сертификатов.

4.3.2. Подпись OASIS DSS-запросов.

4.3.3. Отправка OASIS DSS-запросов на тестовый OASIS DSS-сервер.

4.3.4. Разбор, проверка и визуализация OASIS DSS ответов.

5 Перечень сокращений

АС	Автоматизированная система
ДТС	Доверенная третья сторона
ДУЦ	Доверенный удостоверяющий центр
RFC	
TSL	
API	
Web	
SOAP	
XML	
XSD	
ИОК (PKI)	Инфраструктура открытых ключей
УЦ	Удостоверяющий центр
ЭД	Электронный документ
(К)ЭП	(Квалифицированная) Электронная подпись
ЭЦП	Электронная цифровая подпись
ЭДО	Электронный документооборот
СКПЭП	Сертификат ключа проверки электронной подписи