Protocol for Data Exchange Between Databases and Information Systems

Requirements for Information Systems and Adapter Servers

Specification
Version: 9.5
06.02.2014
60 pages
Y-597-2
## Version History

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>01 July 2010</td>
<td>0.1</td>
<td>First draft</td>
</tr>
</tbody>
</table>
| 10 June 2011    | 8.0     | Unification of version numbers, incorporation of changes from interim editions, mainly:  
• new metaquery xrd.logOnly, and more metaquery examples (listConsumers, listGroups, getProducerData, getConsumerData, getGroupData, getProducerACL, getServiceACL, logOnly);  
• support for sub-offices in query names;  
• new field ski (SubjectKeyIdentifier) in xrd.getConsumerData and xrd.getProducerData queries;  
• new, English-language namespace for Document/Literal Wrapped queries. |
| 28 May 2013     | 8.7     |  
• Document/Literal Wrapped style WSDL sample added in 7.2 ("Sample descriptions of database services").  
• In 3.2, note added about official field being not used any more.  
• MIME attachments subclause moved to clause 3, a subclause describing HTTP-message headers added.  
• Subclauses of clause 5 rearranged, the getMethods metaservice subclause added.  
• Namespace http://x-rad.net/xsd/xroad.xsd replaced by http://x-road.ee/xsd/x-road.xsd.  
• Database namespace http://name.ee.x-rad.net/producer/ replaced by http://name.x-road.ee/producer/. |
| 27 June 2013    | 9.3     | RPC/encoded style related parts removed. |
| 24 September 2013| 9.4     | Corrected sequence of elements in 3.4.1 |
| 6 February 2014 | 9.5     | Corrected copy/paste typo in the description of meta service getServiceACL. |
Table of Contents

1 Introduction.................................................................................................................6
  1.1 Purpose of the document and intended audience.................................................6
  1.2 Operation of X-road from developer’s viewpoint...............................................6
  1.3 Technical Aspects of Connecting to X-Road.......................................................7

2 Definitions and Abbreviations.................................................................................8
  2.1 Standards used........................................................................................................8
    2.1.1 SOAP............................................................................................................8
    2.1.2 WSDL...........................................................................................................8
    2.1.3 Bindings.........................................................................................................8
    2.1.4 Namespaces....................................................................................................8
    2.1.5 Services provided..........................................................................................8

3 Structure of Messages in Data Exchange Protocol..............................................10
  3.1 Division of Data Between Components...............................................................10
  3.2 Elements of THE header COMPONENT............................................................10
  3.3 Message presentation in SOAP............................................................................11
    3.3.1 Restrictions..................................................................................................12
    3.3.2 Service request.............................................................................................12
    3.3.3 Service response.........................................................................................12
    3.3.4 Asynchronous messaging...........................................................................13
  3.4 Error messages......................................................................................................13
    3.4.1 Standard SOAP error message..................................................................13
    3.4.2 Non-technical SOAP error message..........................................................14
  3.5 Messages with MIME attachments......................................................................14
  3.6 HTTP MESSAGE HEADERS...............................................................................15

4 Format of Data Services.........................................................................................16
  4.1 Names of services and parameters.......................................................................16
  4.2 WSDL format........................................................................................................16
  4.3 Publication of Service Descriptions.....................................................................18

5 Metaservices...........................................................................................................20
  5.1 Overview...............................................................................................................20
  5.2 Description of metaservices..................................................................................21
    5.2.1 Service for listing allowed methods (allowedMethods).............................21
    5.2.2 Service for querying the next unsent asynchronous message (asyncNext).................................................................................................................21
    5.2.3 Service for querying the last successfully sent asynchronous message

Protocol for Data Exchange Between Databases and Information Systems

28.05.2013
5.2.4 Service for listing databases (listProducers) ................................................. 22
5.2.5 Service listConsumers .................................................................................. 22
5.2.6 Service listGroups ..................................................................................... 23
5.2.7 Service getProducerData ........................................................................... 23
5.2.8 Service for retrieving organization's certificate fingerprints
                (getConsumerData) ................................................................................ 24
5.2.9 Service for retrieving group members (getGroupData) ............................ 24
5.2.10 Service for retrieving database ACLs (getProducerACL) ....................... 25
5.2.11 Service for retrieving ACL records (getServiceACL) ............................... 25
5.2.12 Service for making a log entry (logOnly) ................................................ 26
5.2.13 Service for listing adapter server's methods (listMethods) ....................... 26
5.2.14 Service for monitoring an adapter server (testSystem) ............................ 27
5.2.15 Service for monitoring the database (getState) .......................................... 27
5.2.16 Service for listing methods supported by database (getMethod) ............ 27
5.2.17 Service for determining service costs (getCharge) .................................. 28
5.2.18 Service for loading a classifier (loadClassificator) .................................. 28
5.2.19 Service for entering into a legacy system (legacyXYZ) ............................ 29

6 References ........................................................................................................ 31

7 Examples ........................................................................................................... 32
7.1 Examples of services ...................................................................................... 32
5.2.12 Service: asyncLast ................................................................................... 51
5.2.13 Service: listProducers .............................................................................. 52
5.2.14 Service: listGroups ................................................................................... 52
5.2.15 Service: getProducerData ........................................................................... 52
5.2.16 Service: getConsumerData ...................................................................... 52
5.2.17 Service: getGroupData .............................................................................. 52

Protocol for Data Exchange Between Databases and Information Systems 9.5
28.05.2013 4/60
7.1.18 Service: getProducerACL.................................................................44
7.1.19 Service: getServiceACL.................................................................45
7.1.20 Service: getMethods.................................................................46
7.1.21 Service: logOnly.........................................................................47
7.2 Sample description of database services (WSDL).............................47
7.2.1 WSDL in document/literal wrapped style........................................47
1 Introduction

1.1 Purpose of the document and intended audience

The purpose of this document is describing the X-Road protocol on a level sufficient for connecting databases and information systems to X-Road. Connecting to X-Road means either making your services available via X-Road, or using services provided by X-Road in your information system. The document explains what restrictions X-Road imposes on the standard protocols used, what types of services can be used/implemented via X-Road, and how services should be described.

The document is intended for information system and database developers who will implement the X-Road services as users or providers of these services.

The reader is expected to be familiar with the SOAP protocol. The purpose of this document is not to describe the SOAP protocol, as there are plenty of other sources of information and X-Road is not tied to any particular implementation of those protocols.

1.2 Operation of X-road from developer's viewpoint

The data exchange protocol of X-Road controls the data exchange between an information system and the adapter server of a database. The information system of an institution provides service to end users by means of platforms and protocols independent of X-Road. The X-Road related communication of both the information system and the adapter server occurs only via corresponding security servers. The security server of an institution makes a connection to the security server of the database and forwards the service invocation (query) received from the information system. The adapter server of the database modifies queries arriving from X-Road to a form that can be processed by the database's data server (which is independent of X-Road), and returns the data server's response in a form suitable for X-Road. Figure 1 illustrates the concept described above.

Illustration 1. The concept of X-Road data exchange protocol

The communication between the institution's information system and its security server, as well as between the database adapter server and its security server is based on the SOAP protocol.

The X-Road infrastructure contains many other components, besides the components described above. Those other components will not be described in this document because they are not necessary for implementing X-Road services. The information systems and
databases that connect to X-Road will only communicate with their own security server on X-Road.

1.3 Technical Aspects of Connecting to X-Road

Adapter servers and information systems can be implemented by any software developer. The only technical requirements presented by X-Road are:

- the software has to communicate with the security server via the standardized X-Road protocol;
- for being connected to X-Road, the institution's information system must have a sufficient security level.

Implementing an adapter server means creating a SOAP server and publishing through it the necessary data services, as well as metaservices obligatory for the adapter servers. The parameters of the adapter server are stored in the database's security server which will access the adapter server.

To provide the interface between institution's information system and X-Road, a SOAP client has to be created for forwarding service invocations. The information system will then pass all service invocation requests via http or https to an URL located in the institution's security server: /cgi-bin/consumer_proxy. Any suitable libraries can be used for creating SOAP servers and clients.
2 Definitions and Abbreviations

2.1 Standards used

2.1.1 SOAP
SOAP [SOAP] is an XML-based data exchange protocol. Information systems and adapter servers can communicate with security servers via protocol that conforms to the SOAP 1.1 specification, with restrictions that will be described in this document.

2.1.2 WSDL
WSDL [WSDL] is a language for describing Web services. X-Road service descriptions conform to the WSDL 1.1 specification, with restrictions that are based on X-Road needs and will be described in this document.

2.1.3 Bindings
WSDL binds a web service with the SOAP messaging protocol. In X-Road binding style Document/Literal Wrapped is used. See [WSDL-STYLE].

2.1.4 Namespaces
The document uses the following namespace prefixes for references.
- xrd – refers to namespace http://x-road.ee/xsd/x-road.xsd with its description on the same address
- xsd – refers to namespace http://www.w3.org/2001/XMLSchema

2.1.5 Services provided
The SOAP methods executed on X-Road form X-Road services. An X-Road service is a two-stage data exchange between an information system and a database, initiated by the information system by sending a query (a message with service input), which is then processed by the database to return query results (a message with service output). From a technical viewpoint, it is irrelevant for X-Road whether the aim of the service is to use the stored data in a database to generate a report based on conditions defined in input, to store the received data in the database, or something else.

There are two types of services: data services and metaservices.

Data services are services specific to a particular database and usually created individually for that database. To give access to these services is the main goal of X-Road. The input and output of data services of a database are specified in the database's documentation. Data services belong to the namespace of their corresponding database: http://name.x-road.ee/producer/, where name is the name of a database.

Metaservices are auxiliary services for obtaining information necessary to perform data
services. The input, output and semantics of metaservices are standardized and will be described in this document. They are similar in all servers providing metaservices. Metaservices belong to X-Road namespace http://x-road.ee/xsd/x-road.xsd.
3 Structure of Messages in Data Exchange Protocol

Note. All messages exchanged over X-Road are encoded as UTF-8.

3.1 Division of Data Between Components

Data exchanged through X-Road are divided into three components, namely: the header, query, and reply components, the exact presentation of which depends on the protocol used. As a rule, a component contains sub-elements, but can as well contain scalar values.

Service input messages, i.e. messages requesting a service, contain the header and query components.

Service output messages, i.e. messages responding to service requests, contain the header, the query component and the reply component. In such case, the input and output headers are identical (but see the note above for a special case). Also, the output query and input query are identical, which means that the input header is copied to the output header, and the input query component is copied to the output query component (some exceptions are in case of request with attachments).

Fields of the header component are presented inside the SOAP envelope header. The query and reply components are presented as elements `<request>` and `<response>` immediately below the root element of the SOAP envelope body.

3.2 Elements of THE header COMPONENT

The header consists of the following elements, which must be immediate child tags of the `<SOAPENV:Header>` tag (see Table 1).

Table 1. Header elements

<table>
<thead>
<tr>
<th>Tag</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>consumer</td>
<td>string</td>
<td>DNS-name of the institution</td>
</tr>
<tr>
<td>producer</td>
<td>string</td>
<td>DNS-name of the database</td>
</tr>
<tr>
<td>userId</td>
<td>string</td>
<td>ID code of the person invoking the service, preceded by a two-letter country code</td>
</tr>
<tr>
<td>id</td>
<td>string</td>
<td>Service invocation nonce (unique identifier)</td>
</tr>
<tr>
<td>service</td>
<td>string</td>
<td>Name of the service to be invoked</td>
</tr>
<tr>
<td>issue</td>
<td>string</td>
<td>Name of file or document related to the service invocation</td>
</tr>
</tbody>
</table>

Since the security server version 5.0 (incl.), two-level (hierarachical) organization and database names can be used, so that for every organization that has joined X-Road, one or more sub-units could be created. In the context of X-Road, such sub-units are actually independent organizations or databases that now can have a structured namespace. More precisely, a name can now contain two parts: an attribute and a registry code (for example, `db.70006317`), or an attribute and a short name of the database (for example, `department.carregistry`).
The service invocation nonce is an unique identifier, which can contain numbers and Latin letters ((azAZ) only. It is generated by the information system of the organization using the service; this system must ensure the global uniqueness of the identifier, for example, by using a sufficiently big random number, or combining a random number with the name of the institution and the checksum of the query. An information system could use the service identifier to associate a query with the data it was based on (e.g., a person’s application).

The service name in the header must match the name of the service being called. It is duplicated to facilitate the processing of signed queries.

In addition to the elements listed above, the header of a query can also contain other header elements described in the X-Road namespace (see Table 2).

Table 2. Additional elements in the query header

<table>
<thead>
<tr>
<th>Tag</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unit</td>
<td>string</td>
<td>Registration code of the institution or its unit on whose behalf the service is used (applied in the legal entity portal)</td>
</tr>
<tr>
<td>position</td>
<td>string</td>
<td>Organizational position or role of the person invoking the service</td>
</tr>
<tr>
<td>userName</td>
<td>string</td>
<td>Name of the person invoking the service</td>
</tr>
<tr>
<td>async</td>
<td>boolean</td>
<td>Specifies asynchronous service. If the value is &quot;true&quot;, then the security server performs the service call asynchronously.</td>
</tr>
</tbody>
</table>
| authenticator | string | Authentication method, one of the following:  
|             |          | • ID-CARD – with a certificate of identity  
|             |          | • CERT – with another certificate  
|             |          | • EXTERNAL – through a third-party service  
|             |          | • PASSWORD – with user ID and a password  
|             |          | Details of the authentication (e.g., the identification of a bank for external authentication) can be given in brackets after the authentication method. |
| paid        | string    | The amount of money paid for invoking the service |
| encrypt     | string    | If an organization has got the right from the X-Road Center to hide queries, with the database agreeing to hide the query, the occurrence of this tag in the query header makes the database security server to encrypt the query log, using the encryption key of the X-Road Center |
| encryptCert | base64    | Authentication certificate of the query invoker's ID Card, in the base64-encoded DER format. Occurrence of this tag in the query header represents the wish to encrypt the query log in the organization's security server, using authentication key of the query invoker's ID Card. This field is used in the Citizen Query Portal only. |
| encrypted   | string    | If the query header contains the encrypt tag and the query log has been successfully encrypted, an empty encrypted tag will be inserted in the reply header. |
| encryptedCert | string | If the query header contains the encryptCert tag and the query log has been successfully encrypted, an empty encryptedCert tag will accordingly be inserted in the reply header. |

Required elements of the query header are listed in the WSDL file. All the tags mentioned above belong to the X-Road namespace.

3.3 Message presentation in SOAP

Note: In this section and the sections that follow, the elements that contain values specific to the service are printed in **boldface**. Texts describing context for which there is no syntax
3.3.1 Restrictions
SOAP is presented using the Document/Literal Wrapped style [WSDL-STYLE].

In X-Road message headers and metaservices, all elements must be presented in the single-reference form. Polymorphic elements cannot be used, that is, all parameters in a message must be accessible via only one path that defines the parameter uniquely.

The namespace prefixes are not used for elements in the query and reply components. (All these elements belong to the service namespace which is defined by a single schema. If the schema referees to an element in some other namespace, the query description schema itself defines this element in its own namespace.)

With the Document/Literal Wrapped style the encodingStyle attribute is not used.

3.3.2 Service request
The service request has the following structure.

```
  <SOAP-ENV:Header>
    Header content
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <m:service xmlns:m="URI">
      <request>
        Request content
      </request>
    </m:service>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The **header** and **request content** denote the SOAP representation of component's all sub-parameters (if the component is a structured parameter) or the component's value (if the component is a scalar parameter).

Elements can have other attributes besides those listed here.

The element `<request>` can be omitted if the service is a metaservice and does not contain any parameters.

3.3.3 Service response
The service response has the following structure.

```
  <SOAP-ENV:Header>
    Header content
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <m:serviceResponse xmlns:m="URI">
      <request>
        Request content
      </request>
      <response>
        Response content
      </response>
    </m:serviceResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```
The header, request, and response content denote the SOAP representation of a component's all sub-parameters (if the component is a structured parameter) or the component's value (if the component is a scalar parameter).

Elements can have other attributes besides those listed here.

The element <request> (and its content) can be omitted if the service does not contain any parameters, or if it is a metaservice implemented in a security server.

### 3.3.4 Asynchronous messaging

A message is considered asynchronous if in the header the field "async" is set to "true". When the message is sent to the database, the response is given by its security server. On success, the server returns a receipt (i.e., acknowledgement), which is an empty response without elements (<response/>), as the response is with asynchronous messaging not a query reply.

### 3.4 Error messages

The output of a service can be an error message. Error messages can be presented in two ways, depending on the content of the message.

- To notify about service failure unrelated to the user, a standard SOAP error message is used.
- To report an input error (for example, the user entered incorrect data or the query returned no results), the element <response> should contain a structure that presents the error string in the elements faultCode and faultString.

#### 3.4.1 Standard SOAP error message

If possible, a SOAP error message should also contain a header, but it is not obligatory. Error codes correspond to codes in the SOAP specification. Codes belonging to class Client refer to errors in the service input composed by the information system. Codes belonging to class Server refer to errors not related to service input. Error codes generated by security servers are listed in the document [SERR].

The parameter faultactor is used to store the identity of the component that created the error message, if possible.

```xml
  <SOAP-ENV:Header>
    Header content
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>fault code</faultcode>
      <faultstring>fault string</faultstring>
      <faultactor>fault actor</faultactor>
      <detail>
        fault details
      </detail>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

---

**Protocol for Data Exchange Between Databases and Information Systems**

28.05.2013

Page 13/60
Under the header component is given a SOAP representation of all sub-parameters of the corresponding component.

### 3.4.2 Non-technical SOAP error message

```xml
  <SOAP-ENV:Header>
    <Header content />
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <m:serviceResponse xmlns:m="URI">
      <request>
        <Request content />
      </request>
      <response>
        <faultCode>fault code</faultCode>
        <faultString>fault string</faultString>
      </response>
    </m:serviceResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The `header` and `request content` denote the SOAP representation of a component's all sub-parameters (if the component is a structured parameter) or a component's value (if the component is a scalar parameter).

### 3.5 Messages with MIME attachments

Messages with SOAP protocol are allowed to contain MIME attachments, if the wire format of the message corresponds to specification [SA] and service description corresponds to specification [WSDL] and following constraints are in use:

- MIME container header field `Content-Type` contains parameter `type` with value of `text/xml` or `application/xop+xml`;
- MIME container header field `Content-Type` contains parameter `boundary` with value of the boundary string used to separate parts of the MIME message;
- the first part of a MIME container is always a SOAP envelope, which contains the header and body;
- the `Content-Transfer-Encoding` of the part containing the SOAP envelope is `8bit`;
- the SOAP envelope contains references to all attachments in the message, by using the value of `Content-Id` in the attachment header.
- the message will be encoded as MIME container if the WSDL describes the MIME binding for the method;
- if the message is encoded as MIME container then values of all scalar elements of the input with type of either `xsd:base64Binary` or `xsd:hexBinary` will be sent as attachments;
- if the service query contains links to MIME attachments, then the response message contains SHA-512 hash values of the attached data.

Attachments are the preferred way of transporting huge data, because transmitting large attachments requires less security server resources than transmitting the same data within a SOAP message without attachments.
3.6 HTTP MESSAGE HEADERS

In plain queries the value of the HTTP header field `Content-Type` has to be `text/xml`. In case of queries with attachments the value of the field `Content-Type` is `multipart/related`. In SOAP queries the `SOAPAction` field contains no value.
4 Format of Data Services

4.1 Names of services and parameters
A service is identified by its name. The name of a service consists of three components separated by dots, in the form database.method.version (for example, landcadastre.query1.v1), where:

- database is the name of the database. The name is fixed upon its joining X-Road;
- method is short name of the service, unique among this database's services;
- version is version of the service in the form vN, where N is a positive integer denoting the version number.

All three components must conform to the character set allowed in DNS (i.e. Latin letters, numbers, and hyphen). The short name of a data service must not match the name of any metaservice.

Any individual version of a service is fixed and must not change over time. Any change in the description of a service requires creation of either a new version of the service, or, if the semantics of the service changed, creation of a new service. The semantics of a service must be the same across all versions of the service.

The WSDL must describe only one, the newest, version of the service. However, the database must be able to reply to messages that are in the old format (i.e., described with the earlier WSDL version).

Names of the service parameters can consist only of Latin letters, numbers, underscores, and dots. The name of a parameter cannot begin with an underscore or a number.

4.2 WSDL format

Note. All descriptions in this subclause are given for document/literal wrapped style only.

The WSDL format of service description conforms to the WSDL specification, with the restrictions listed below.

The combination binding style/use must be document/literal (that is, binding style="document"; use="literal").

Query and response parameters are described as an XML schema [XSD].

Also, the following elements have been added to WSDL to store information necessary for X-Road (see table 3).

<table>
<thead>
<tr>
<th>Element location (XPath)</th>
<th>Value/Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/definitions/service/port/xrd:address</td>
<td>No value, but the element has the attribute producer, the value of which is the name of the database.</td>
</tr>
<tr>
<td>/definitions/service/port/xrd:title</td>
<td>Title of the database (for displaying to users)</td>
</tr>
<tr>
<td>/definitions/binding/operation/xrd:version</td>
<td>Service version</td>
</tr>
<tr>
<td>/definitions/binding/operation/xrd:charge</td>
<td>Payment options (if money is charged for)</td>
</tr>
<tr>
<td>Element location (XPath)</td>
<td>Value/Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>/definitions/binding/operation/xrd:charge/xrd:account</td>
<td>Bank account of the payee</td>
</tr>
<tr>
<td>/definitions/binding/operation/xrd:charge/xrd:receiverName</td>
<td>Name of the payee</td>
</tr>
<tr>
<td>/definitions/binding/operation/xrd:charge/xrd:message</td>
<td>Description of payment</td>
</tr>
<tr>
<td>/definitions/binding/operation/xrd:charge/xrd:amount</td>
<td>Amount payable. The element can have the attribute chargeType, the value of which is the type of payee to whom the sum applies.</td>
</tr>
<tr>
<td>/definitions/binding/operation/xrd:noContent</td>
<td>Handling of parameter elements with no content.</td>
</tr>
<tr>
<td></td>
<td><strong>null</strong> – no content means that the corresponding parameter has no value; <strong>empty</strong> – no content means that the corresponding parameter is null (this is the default)</td>
</tr>
<tr>
<td>/definitions/binding/operation/xrd:requireContent</td>
<td>Requirements for values in mandatory fields:</td>
</tr>
<tr>
<td></td>
<td><strong>true</strong> – mandatory fields must contain values; <strong>false</strong> – mandatory field can have null value.</td>
</tr>
<tr>
<td></td>
<td>A field is mandatory if the value of the attribute minOccurs &gt; 0.</td>
</tr>
<tr>
<td>/definitions/portType/operation/documentation/xrd:title</td>
<td>Service title (for displaying to users)</td>
</tr>
<tr>
<td>/definitions/portType/operation/documentation/xrd:notes</td>
<td>Service comments (for displaying to users)</td>
</tr>
<tr>
<td>/definitions/portType/operation/documentation/xrd:techNotes</td>
<td>Service comments (for displaying to developers)</td>
</tr>
<tr>
<td>/definitions/portType/operation/documentation/xrd:actionTitle</td>
<td>Title for the submit action (for example, <strong>Save</strong>)</td>
</tr>
<tr>
<td>/annotation/appinfo/xrd:title</td>
<td>Parameter title (for displaying to users)</td>
</tr>
<tr>
<td>/annotation/appinfo/xrd:notes</td>
<td>Parameter comments (for displaying to users)</td>
</tr>
<tr>
<td>/annotation/appinfo/xrd:techNotes</td>
<td>Parameter comments (for displaying to developers)</td>
</tr>
<tr>
<td>/annotation/appinfo/xrd:fieldType</td>
<td>Parameter type, one of the following: <strong>textarea</strong> – corresponds to the HTML &lt;textarea&gt; element; <strong>comment</strong> – text displayed to the user (the value is a constant set in the WSDL or in a complex query).</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: For the database to be able to assume that the user has not changed the value of a parameter, the attribute default is required.</td>
</tr>
<tr>
<td>/annotation/appinfo/xrd:fieldRows</td>
<td>Recommended number of rows on screen for the input field (the rows attribute in the HTML element &lt;textarea rows=&quot;...&quot; cols=&quot;...&quot;&gt; &lt;textarea&gt;)</td>
</tr>
<tr>
<td>/annotation/appinfo/xrd:fieldCols</td>
<td>Recommended number of columns on screen for the input field (the cols attribute in the HTML element &lt;textarea rows=&quot;...&quot; cols=&quot;...&quot;&gt; &lt;textarea&gt;)</td>
</tr>
<tr>
<td>/annotation/appinfo/xrd:fieldSize</td>
<td>The length of an input field on the screen (the size attribute in the HTML element &lt;input type=&quot;text&quot; size=&quot;...&quot;&gt;)</td>
</tr>
<tr>
<td>/annotation/appinfo/xrd:Wildcard</td>
<td>List of metasymbols allowed</td>
</tr>
<tr>
<td>/annotation/appinfo/xrd:ref</td>
<td>Name of the parameter to which the current parameter is related.</td>
</tr>
</tbody>
</table>
If the element `<xrd:address>` is present, then the service is considered accessible via the X-Road protocol.

In describing X-Road services, the following elements are obligatory: name and title of the database; name, heading, and version number of the service. It is recommended to specify a name for each parameter.

In the elements `<xrd:title>`, `<xrd:notes>` and `<xrd:techNotes>`, the attribute `xml:lang` can be used to store the language in which the value of the element is presented. If the attribute is missing, the default value "en" (English) is presumed.

When invoking services, metasymbols can be used in parameter values if the metasymbols are listed in the description of the service. In service description, all metasymbols allowed in parameters should be listed in the element `<xrd: wildcard>`. The following symbols can be used (see Table 4).

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>Can be used to replace any number of arbitrary symbols</td>
</tr>
<tr>
<td>?</td>
<td>Can be used to replace one arbitrary symbol</td>
</tr>
<tr>
<td>-</td>
<td>Can be used to denote an interval</td>
</tr>
<tr>
<td>P</td>
<td>Value is considered a prefix</td>
</tr>
<tr>
<td>S</td>
<td>Value is considered a substring</td>
</tr>
</tbody>
</table>

To avoid the possibility that an error causes the absence of service output and another error causes the absence of the corresponding error message, it is recommended to design the output of a data service so that the response of the query contains at least one non-empty scalar parameter, independent of how the service was invoked. This parameter could be a non-technical error message.

A default or fixed value of a scalar parameter is described as a value of the `default` or `fixed` attribute, respectively.

In addition to describing the default or fixed value as a constant, it may also be given by a reference to a special variable defined in the information system. In that case, the information system uses the value of the variable as a default or fixed value. A default or fixed value is referencing a variable if the value of the attribute starts with the prefix string 'xrdvar:'. The rest of the value is considered to be the name of the variable, which may contain only Latin characters ([a-zA-Z]), numbers, and underscores.

Service requests with no output description (for which the WSDL element `/binding/operation/output` is missing) will be forwarded asynchronously.

### 4.3 Publication of Service Descriptions

Database administrator publishes descriptions of the database's data services as a WSDL-format file in the database's adapter server. All data services therein should be described in a single common WSDL file, the URL of which is set in the database's security server. If necessary, the file might link to other files either on public Internet or in the same folder as the main WSDL file, provided that the location is accessible to the security server.

If the WSDL files are on a Web server accessible to the institution's information system, then the service descriptions can be downloaded with ordinary HTTP GET queries. If the Web server is not accessible, then the service descriptions can be downloaded only via the institution's security server over the URL: `http://secureproxy/cgi-bin/uriproxy?uri=URI`, where:
• secureproxy – security server’s address
• URI – address of the WSDL or schema (absolute URI valid for the public Internet)

Similarly, it is possible to download a WSDL file that describes the services by accessing the security server through the URL: http://secureproxy/cgi-bin/uriproxy?producer=name, where:

• secureproxy – security server’s address,
• name – name of the database.
5 Metaservices

5.1 Overview

An institution's information system can access the following metaservices:

- allowedMethods
- asyncNext
- asyncLast
- listProducers
- listConsumers
- listGroups
- getProducerData
- getConsumerData
- getGroupData
- getProducerACL
- getServiceACL
- getState
- getMethods
- logOnly

An institution's information system may also have access to some of the following metaservices on some databases:

- getCharge
- loadClassificator
- legacyXYZ (where XYZ is an arbitrary text characteristic to that query)

An adapter server is required to implement the following metaservice:

- listMethods

An adapter server may be required to implement the following metaservice, depending on agreements:

- testSystem

An adapter server may implement the following metaservices:

- getCharge
- loadClassificator
- legacyXYZ
5.2 Description of metaservices

5.2.1 Service for listing allowed methods (allowedMethods)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>producer.allowedMethods (where producer is the name of a database)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns</td>
<td>List of methods available to the calling organization. Method names are in the format database.method.version</td>
</tr>
<tr>
<td>Implemented in</td>
<td>Database's security server</td>
</tr>
<tr>
<td>Description</td>
<td>The service is called by the organization's information system.</td>
</tr>
</tbody>
</table>

Method call:

**Header:** Required

**Input Request component:** Not used

**Output Request component:** Not used

**Output Response component:**

Array

String -- Name of service in the format database.method.version

5.2.2 Service for querying the next unsent asynchronous message (asyncNext)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>xrd.asyncNext</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns</td>
<td>The nonce (ID) of the oldest message in the queue of asynchronous messages</td>
</tr>
<tr>
<td>Implemented in</td>
<td>Organization's security server</td>
</tr>
<tr>
<td>Description</td>
<td>According to the query parameter, asynchronous messages are chosen. If a name is specified in the request, the specified database's queue will be inspected. If the name is empty, the queue with the oldest first message (that is, one that arrived at the security server first) will be inspected. Output response is a string (possibly empty) that contains the ID (i.e., nonce) of the first message in the queue being inspected. If the queue was not found or it is empty, the output response will be an empty string. There is a different queue for each database, because security server is allowed to send messages to different databases independently from each other, as long as the order of messages within a database is maintained.</td>
</tr>
</tbody>
</table>

Method call:

**Header:** Not used

**Input Request component:**

String -- name of a database or an empty string

**Output Request component:** Not used

**Output Response component:**

String -- The ID of the oldest unsent message in the queue

5.2.3 Service for querying the last successfully sent asynchronous message (asyncLast)

Properties:
Name | xrd.asyncLast
---|---
**Returns** | Nonce (ID) of the last successfully sent asynchronous message
**Implemented in** | Organization's security server
**Description** | If input string is not empty then it is the name of the database whose queue should be inspected. Otherwise the last successful asynchronous message will be reported no matter which database it was sent to. Output response is a string (possibly empty) that contains the ID (i.e., nonce) of the last successfully sent message in the queue being inspected. If no asynchronous message has been sent successfully then the output response will be an empty string.

Method call:
**Header:** Not used
**Input Request component:**
String -- Name of a database or an empty string

**Output Request component:** Not used
**Output Response component:**
String -- The ID of the last successfully sent asynchronous message in the queue

### 5.2.4 Service for listing databases (listProducers)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>xrd.listProducers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Returns</strong></td>
<td>List of available databases</td>
</tr>
<tr>
<td><strong>Implemented in</strong></td>
<td>Organization's security server</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>The service allows automatic retrieval of database names. With such a list, it is later possible to retrieve a list of all available services through the xrd.allowedMethods metaservice.</td>
</tr>
</tbody>
</table>

Method call:
**Header:** Not used
**Input Request component:** Not used
**Output Request component:** Not used
**Output Response component:**
Array
  Struct
    String name -- Short name or prefix of the X-Road database
    String description -- Full name of the database

### 5.2.5 Service listConsumers

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>xrd.listConsumers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Returns</strong></td>
<td>List of organizations registered with X-Road</td>
</tr>
<tr>
<td><strong>Implemented in</strong></td>
<td>Organization's security server</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>The service’s purpose is to provide information about organizations that use X-Road services. The output Response is a sequence of structures. Each structure corresponds to one organization.</td>
</tr>
</tbody>
</table>

Method call:
5.2.6 Service listGroups

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>xrd.listGroups</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns</td>
<td>List of groups registered with X-Road</td>
</tr>
<tr>
<td>Implemented in</td>
<td>Organization's security server</td>
</tr>
<tr>
<td>Description</td>
<td>The service's purpose is to provide information about registered groups. The output Response is a sequence of structures. Each structure corresponds to one group.</td>
</tr>
</tbody>
</table>

Method call:

Header: Not used

Input Request component: Not used

Output Request component: Not used

Output Response component:

Array

Struct

String name -- The organization's prefix, i.e., its X-Road short name

String description -- The organization's full name

5.2.7 Service getProducerData

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>xrd.getProducerData</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns</td>
<td>List of the database's certificate fingerprints and SKIs (Subject Key Identifiers).</td>
</tr>
<tr>
<td>Implemented in</td>
<td>Organization's security server</td>
</tr>
<tr>
<td>Description</td>
<td>The input Request component is the name of the database whose certificate fingerprints are needed. The output Response is a sequence of structures. Each structure corresponds to one database certificate.</td>
</tr>
</tbody>
</table>

Method call:

Header: Not used

Input Request component:

String -- Name of the database to be queried

Output Request component: Not used

Output Response component:

Array
Struct
  String certHash -- Database's certificate fingerprint
  String ski -- Database's certificate Subject Key Identifier

5.2.8 Service for retrieving organization's certificate fingerprints (getConsumerData)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>xrd.getConsumerData</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns</td>
<td>List of the organization's certificate fingerprints and SKIs (Subject Key Identifiers).</td>
</tr>
<tr>
<td>Implemented in</td>
<td>Organization's security server</td>
</tr>
<tr>
<td>Description</td>
<td>The input Request component is the name of the organization whose certificate fingerprints are needed. The output Response is a sequence of structures. Each structure corresponds to one organization certificate.</td>
</tr>
</tbody>
</table>

Method call:
Header: Not used
Input Request component:
String -- Organization's name

Output Request component: Not used
Output Response component:
Array
  Struct
    String certHash -- Organization's certificate fingerprint
    String ski -- Organization's certificate Subject Key Identifier

5.2.9 Service for retrieving group members (getGroupData)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>xrd.getGroupData</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns</td>
<td>List of organizations belonging to a given group</td>
</tr>
<tr>
<td>Implemented in</td>
<td>Organization's security server</td>
</tr>
<tr>
<td>Description</td>
<td>The input Request is a string containing the name of the group in question. The output Response is a sequence of structures. Each structure corresponds to one group member, and the only field in the structure is name (the organization's prefix, or short name).</td>
</tr>
</tbody>
</table>

Method call:
Header: Not used
Input Request component:
String -- Group name

Output Request component: Not used
Output Response component:
Array
  Struct
    String name -- Organization's prefix, i.e., its X-Road short name
5.2.10 Service for retrieving database ACLs (getProducerACL)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>producer.getProducerACL</th>
</tr>
</thead>
<tbody>
<tr>
<td>(where producer is the name of the given database)</td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td>Complete list of the given database’s ACL records</td>
</tr>
<tr>
<td>Implemented in</td>
<td>Database’s security server</td>
</tr>
<tr>
<td>Description</td>
<td>The output Response is a sequence of structures. Each structure represents one ACL record and contains as members the following fields:</td>
</tr>
<tr>
<td></td>
<td>• service – name of the service</td>
</tr>
<tr>
<td></td>
<td>• party – name of the organization or group</td>
</tr>
<tr>
<td></td>
<td>• type – ACL record type, one of:</td>
</tr>
<tr>
<td></td>
<td>• consumer – the permitted party is an organization;</td>
</tr>
<tr>
<td></td>
<td>• group – the permitted party is a group;</td>
</tr>
<tr>
<td></td>
<td>• secure – the permitted party is an organization with a permission to encrypt queries.</td>
</tr>
</tbody>
</table>

Method call:

**Header:** Required

**Input Request component:** Not used

**Output Request component:** Not used

**Output Response component:**

Array

Struct

String service -- Service name in the form database.query
String party -- The permitted party
String type -- Type of ACL record

5.2.11 Service for retrieving ACL records (getServiceACL)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>producer.getServiceACL</th>
</tr>
</thead>
<tbody>
<tr>
<td>(where producer is the name of the given database)</td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td>Complete list of the given service’s ACL records</td>
</tr>
<tr>
<td>Implemented in</td>
<td>Database’s security server</td>
</tr>
<tr>
<td>Description</td>
<td>The input Request is a String containing the name of the service whose ACL is needed.</td>
</tr>
<tr>
<td></td>
<td>The output Response contains a sequence of structures. Each structure represents one ACL record and contains as members the fields &quot;party&quot; (name of the organization or group) and &quot;type&quot; (ACL record type: one of consumer, group, or secure; the latter meaning an organization with a permission to encrypt queries).</td>
</tr>
</tbody>
</table>

Method call:

**Header:** Required

**Input Request component:**

String -- Service name in the form database.query or database.query.version

**Output Request component:** Not used

**Output Response component:**

Array

Struct

String party -- The organization in question
5.2.12 Service for making a log entry (logOnly)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>xrd.logOnly</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>(A call to this service is only logged to the security server’s query log (sslog) and not forwarded to any database)</td>
</tr>
<tr>
<td>Implemented in</td>
<td>Organization’s security server</td>
</tr>
<tr>
<td>Description</td>
<td>The service enables information systems to use X-Road secure logging for special purposes.</td>
</tr>
</tbody>
</table>

Method call:

**Header:** Required; value of the xrd:producer field must be the string "xrd"

**Input Request component:**

Struct -- Data to be logged

**Output Request component:** Not used

**Output Response component:** Not used

5.2.13 Service for listing adapter server's methods (listMethods)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>system.listMethods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns</td>
<td>List of all methods implemented in the adapter server</td>
</tr>
<tr>
<td>Implemented in</td>
<td>Adapter server</td>
</tr>
<tr>
<td>Description</td>
<td>The service is meant to be called only by the database’s own security server.</td>
</tr>
</tbody>
</table>

Method call:

**Header:** Not used

**Input Request component:** Not used

**Output Request component:** Not used

**Output Response component:**

Array

String -- Name of service in the form database.query.version

5.2.14 Service for monitoring an adapter server (testSystem)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>system.testSystem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns</td>
<td>None</td>
</tr>
<tr>
<td>Implemented in</td>
<td>Adapter server</td>
</tr>
<tr>
<td>Description</td>
<td>The service is used to verify that the adapter server and the database are operating correctly. The service can be used only in the local security server of the database. If the adapter server fails, it must generate a standard error message (see chapter &quot;Standard SOAP error messages&quot;).</td>
</tr>
</tbody>
</table>

Method call:

---

Protocol for Data Exchange Between Databases and Information Systems

28.05.2013

9.5 26/60
5.2.15 Service for monitoring the database (getState)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>producer.getState</th>
</tr>
</thead>
<tbody>
<tr>
<td>(where producer is the name of a database)</td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td>Status of the database</td>
</tr>
<tr>
<td>Implemented in</td>
<td>Database's security server</td>
</tr>
<tr>
<td>Description</td>
<td>The service returns the status that the security server has previously received from the adapter server through a testSystem call.</td>
</tr>
</tbody>
</table>

Method call:

Header: Required
Input Request component: Not used
Output Request component: Not used
Output Response component:

Int -- Database status (0 - unknown; 1 - OK; 2 - error)

5.2.16 Service for listing methods supported by database (getMethods)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>producer.getMethods</th>
</tr>
</thead>
<tbody>
<tr>
<td>(where producer is the name of a database)</td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td>List of all methods supported by the database, the names being in format database.query.version.</td>
</tr>
<tr>
<td>Implemented in</td>
<td>Database's security server</td>
</tr>
<tr>
<td>Description</td>
<td>The service is meant to be called from the information system</td>
</tr>
</tbody>
</table>

Method call:

Header: Required
Input Request component: Not used
Output Request component: Not used
Output Response component:

Array

String -- Name of service in the form database.query.version

5.2.17 Service for determining service costs (getCharge)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>producer.getCharge</th>
</tr>
</thead>
<tbody>
<tr>
<td>(where producer is the name of a database)</td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td>Amount of fee or charge for invoking a particular service (identified by name) for the user whose ID code is inside the request header.</td>
</tr>
</tbody>
</table>
Method call:

**Header:** Required

**Input Request component:**

String  -- Name of the service in the form database.query.version

**Output Request component:** the same as input Request component

**Output Response component:**

Struct

Int amount  -- Amount of fee

5.2.18  Service for loading a classifier (loadClassificator)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>producer.loadClassificator</th>
</tr>
</thead>
<tbody>
<tr>
<td>(where producer is the name of a database)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>One of the following: a list of classifiers, the contents of a classifier, or a subset of the contents of a classifier.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implemented in</td>
<td>Adapter server</td>
</tr>
</tbody>
</table>

**Description**

If the <request> component is an empty string, then a list of classifier names is returned.

A subset of a classifier is defined by the Date parameter from, which sets the earliest date for modifications to the classificator specified, for example, using an LDAP identifier. The output Response’s structure corresponds to the LDAP tree structure of response data. A LDAP record is presented as a structure with sub-parameters containing fields of that record and lists of sub-records. Elements of lists in subrecords are anonymous; names of lists are values of the objectClass field of the LDAP records corresponding to elements. If a field of a LDAP record has several values, then the field is presented as a list of anonymous values, and the name of the list is the name of the field.

Method call (for retrieving a list of classifiers):

**Header:** Required

**Input Request component:**

String  -- Empty string

**Output Request component:** the same as the input Request component

**Output Request component:**

Struct

Array classifierNames  -- A sequence of classifiers available from this database

String  -- An agreed name of the classifier

Method call (for retrieving the contents of a specific classifier):

**Header:** Required

**Input Request component:**
Struct  -- Description of the data requested
String name  -- Name of classifier
String subset  -- Definition of a subset of the classifier's contents, such as the distinguished name of a branch from a classifier tree
Date from  -- Include changes to the classifier since this date (inclusive)
String max  -- Include changes to the classifier up to this date (inclusive)

Output Request component: the same as input Request component
Output Response component:
Struct  -- Matching part of the classifier

5.2.19 Service for entering into a legacy system (legacyXYZ)

Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>producer.legacyXYZ</th>
</tr>
</thead>
<tbody>
<tr>
<td>(where producer is the name of a database and XYZ is arbitrary text characteristic to that service)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implemented in</td>
<td>Adapter server</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The services are used for logging in to legacy systems. This type of service allows to use the authentication and authorization methods of an information system integrated with X-Road in other information systems. The information system, when creating the query, adds all names of legacy system services allowed for that person in the given database to the input Request of the query. The security server of the database checks, whether the institution is allowed to invoke all the services listed. If the list contains services that the institution has no right to invoke, then the security server will respond with a fault message and will refuse to invoke the service. The URL in the service response is a unique URL created by adapter server and/or legacy system's server that uniquely identifies the session, for example: <a href="https://legacy.example.com/cgi-bin/legacy.cgi?id=6d82fba82752264cabf9e11">https://legacy.example.com/cgi-bin/legacy.cgi?id=6d82fba82752264cabf9e11</a> The URL in the service response should be available over HTTPS (not HTTP). The service must be implemented so that the id is unique and not possible to guess. The information system servicing the URL must guarantee the expiration of the URL.</td>
</tr>
</tbody>
</table>

Method call:

Header: Required

Input Request component:

Array

String  -- List of services the person (i.e., the official) is allowed to access, in the form database.query or database.query.version

Output Request component: same as input Request component

Output Response component:

String url  -- URL of the service
6 References

[WSDL] Web Services Description Language (WSDL) 1.1
http://www.w3.org/TR/wsdl

[WSDL-STYLE] Which style of WSDL should I use?

[XSD] XML Schema Part 2: Datatypes
http://www.w3.org/TR/xmlschema-2/

[SOAP] Simple Object Access Protocol (SOAP) 1.1
http://www.w3.org/TR/SOAP/

[SA] SOAP Messages with Attachments
http://www.w3.org/TR/SOAP-attachments

[SERR] SOAP error codes used in X-Road
soap_veakoodid 0.4 Y-560-1. (Document is in estonian)
7 Examples

7.1 Examples of services

7.1.1 Service: listProducers

Query:

```xml
<?xml version="1.0" encoding="utf-8"?>
    xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
    <SOAP-ENV:Body>
        <xrd:listProducers xmlns:xrd="http://x-road.ee/xsd/x-road.xsd"/>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Response:

```xml
<?xml version="1.0" encoding="utf-8"?>
    xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
    <SOAP-ENV:Body>
        <xrd:listProducersResponse xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
            <response>
                <item>
                    <name>car-registry</name>
                    <description>National Traffic Registry</description>
                </item>
                <item>
                    <name>pr</name>
                    <description>Population Registry</description>
                </item>
                <item>
                    <name>land-cadastre</name>
                    <description>Land Cadastre</description>
                </item>
            </response>
        </xrd:listProducersResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

7.1.2 Service: allowedMethods

Query:

```xml
<?xml version="1.0" encoding="utf-8"?>
    xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
    <SOAP-ENV:Header>
        <xrd:consumer>10239452</xrd:consumer>
        <xrd:producer>land-cadastre</xrd:producer>
        <xrd:userId>EE:PIN:abc4567</xrd:userId>
        <xrd:id>411d675661409fed365ad8135f8210be07613da</xrd:id>
        <xrd:service>land-cadastre.allowedMethods</xrd:service>
        <xrd:issue/>
    </SOAP-ENV:Header>
    <SOAP-ENV:Body>
        <xrd:allowedMethods/>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```
For public use

Response:

<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
<SOAP-ENV:Header>
  <xrd:consumer>10239452</xrd:consumer>
  <xrd:producer>land-cadastre</xrd:producer>
  <xrd:userId>EE:PIN:abc4567</xrd:userId>
  <xrd:id>411d6755661409fed365ad8135f8210be07613da</xrd:id>
  <xrd:service>land-cadastre.allowedMethods</xrd:service>
  <xrd:issue/>
</SOAP-ENV:Header>
<SOAP-ENV:Body>
  <xrd:allowedMethodsResponse>
    <response>
      <item>land-cadastre.cu.v1</item>
      <item>land-cadastre.cuAddres.v1</item>
    </response>
  </xrd:allowedMethodsResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

7.1.3 Service: listMethods

Query:

<?xml version="1.0" encoding="utf-8"?>
xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
<SOAP-ENV:Body>
  <xrd:listMethods xmlns:xrd="http://x-road.ee/xsd/x-road.xsd"/>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Response:

<?xml version="1.0" encoding="utf-8"?>
xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
<SOAP-ENV:Body>
  <xrd:listMethodsResponse xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
    <response>
      <item>land-cadastre.cu.v1</item>
      <item>land-cadastre.cuAddres.v1</item>
      <item>land-cadastre.legacy1.v1</item>
    </response>
  </xrd:listMethodsResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

7.1.4 Service: testSystem

Query:

<?xml version="1.0" encoding="utf-8"?>
xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
<SOAP-ENV:Body>
  <xrd:testSystem xmlns:xrd="http://x-road.ee/xsd/x-road.xsd"/>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
7.1.5 Service: getState

Query:

```xml
<?xml version="1.0" encoding="utf-8"?>
  <SOAP-ENV:Header>
    <xrd:consumer>10239452</xrd:consumer>
    <xrd:producer>land-cadastre</xrd:producer>
    <xrd:userId>EE30101010007</xrd:userId>
    <xrd:id>411d6755661409fed365ad8135f8210be07613da</xrd:id>
    <xrd:service>land-cadastre.getState</xrd:service>
    <xrd:issue/>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <xrd:getState/>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Response:

```xml
<?xml version="1.0" encoding="utf-8"?>
  <SOAP-ENV:Header>
    <xrd:consumer>10239452</xrd:consumer>
    <xrd:producer>land-cadastre</xrd:producer>
    <xrd:userId>EE30101010007</xrd:userId>
    <xrd:id>411d6755661409fed365ad8135f8210be07613da</xrd:id>
    <xrd:service>land-cadastre.getState</xrd:service>
    <xrd:issue/>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <xrd:getStateResponse/>
    <response>1</response>
    <xrd:getStateResponse/>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

7.1.6 Data service

Query:

```xml
<?xml version="1.0" encoding="utf-8"?>
  <SOAP-ENV:Header xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
    <xrd:consumer>10239452</xrd:consumer>
    <xrd:producer>land-cadastre</xrd:producer>
    <xrd:userId>EE30101010007</xrd:userId>
    <xrd:id>3aed1ae3813eb7fbed9396fda70ca1215d3f3fe1</xrd:id>
    <xrd:service>land-cadastre.cuAddres.v1</xrd:service>
    <xrd:issue/>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <xrd:getStateResponse/>
    <response>1</response>
    <xrd:getStateResponse/>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```
<SOAP-ENV:Body>
  <ns4:cuAddres
    xmlns:ns4="http://land-cadastre.ee.x-road.ee/producer/">
    <request>
      <province>0037</province>
      <localGovernment>0784</localGovernment>
      <cuOfficialName>cross road square*</cuOfficialName>
      <cuMax>100</cuMax>
    </request>
  </ns4:cuAddres>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Response:
<?xml version="1.0" encoding="utf-8"?>
  xmlns:ns4="http://assert.x-road.ee/producer/"
  xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Header>
    <xrd:consumer>10239452</xrd:consumer>
    <xrd:producer>land-cadastre</xrd:producer>
    <xrd:userId>EE30101010007</xrd:userId>
    <xrd:id>3aed1ae3813eb7fbed9396fda70ca1215d3f3fe1</xrd:id>
    <xrd:service>land-cadastre.cuAddres.v1</xrd:service>
    <xrd:issue/>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <ns4:cuAddresResponse
      xmlns:ns4="http://land-cadastre.ee.x-rd.net/producer/">
      <request>
        <province>0037</province>
        <localGovernment>0784</localGovernment>
        <cuOfficialName>cross road square*</cuOfficialName>
        <cuMax>100</cuMax>
      </request>
      <response>
        <item>
          <cadastralUnit>012:345:67</cadastralUnit>
          <cuLocation>Cross Rode Square 1A</cuLocation>
          <province>0037</province>
          <localGovernment>0784</localGovernment>
          <registered>1999-09-19T19:09:10</registered>
          <purpose>001</purpose>
        </item>
        <item>
          <cadastralUnit>012:345:68</cadastralUnit>
          <cuLocation>Cross Rode Square 1B</cuLocation>
          <province>0037</province>
          <localGovernment>0784</localGovernment>
          <registered>1999-09-19T19:09:10</registered>
          <purpose>001</purpose>
        </item>
      </response>
    </ns4:cuAddresResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

7.1.7 Standard error with header
<?xml version="1.0" encoding="utf-8"?>
  xmlns:ns4="http://assert.x-road.ee/producer/"
  xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Header>
    <xrd:consumer>abc</xrd:consumer>
  </SOAP-ENV:Header>
7.1.8  **Service: loadClassificator**

Query that returns a list of classifiers:

```xml
<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Header>
    <xrd:consumer>10239452</xrd:consumer>
    <xrd:producer>assert</xrd:producer>
    <xrd:userId>EE30101010007</xrd:userId>
    <xrd:id>3e47be25da43528deb639f5965d58b4d21bbf173</xrd:id>
    <xrd:service>assert.loadClassificator</xrd:service>
    <xrd:issue/>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <xrd:loadClassificator>
      <request/>
    </xrd:loadClassificator>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Response with the list of classifiers:

```xml
<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Header>
    <xrd:consumer>10239452</xrd:consumer>
    <xrd:producer>assert</xrd:producer>
    <xrd:userId>EE30101010007</xrd:userId>
    <xrd:id>3e47be25da43528deb639f5965d58b4d21bbf173</xrd:id>
    <xrd:service>assert.loadClassificator</xrd:service>
    <xrd:issue/>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <xrd:loadClassificatorResponse>
      <request/>
      <response>
        <classifierNames>
          <item>EHAK</item>
          <item>abc</item>
        </classifierNames>
      </response>
    </xrd:loadClassificatorResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```
Query that loads a classifier:

<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
    <SOAP-ENV:Header>
        <xrd:consumer>10239452</xrd:consumer>
        <xrd:producer>assert</xrd:producer>
        <xrd:userId>EE30101010007</xrd:userId>
        <xrd:id>00f340cc5da4d3102c0859cc5f2ae2547b07b2c3</xrd:id>
        <xrd:service>assert.loadClassificator</xrd:service>
    </SOAP-ENV:Header>
    <SOAP-ENV:Body>
        <xrd:loadClassificator>
            <request>
                <name>EHAK</name>
                <subset/>
                <from/>
                <max/>
            </request>
        </xrd:loadClassificator>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Response with the content of the classifier:

<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
    <SOAP-ENV:Header>
        <xrd:consumer>10239452</xrd:consumer>
        <xrd:producer>assert</xrd:producer>
        <xrd:userId>EE30101010007</xrd:userId>
        <xrd:id>00f340cc5da4d3102c0859cc5f2ae2547b07b2c3</xrd:id>
        <xrd:service>assert.loadClassificator</xrd:service>
    </SOAP-ENV:Header>
    <SOAP-ENV:Body>
        <xrd:loadClassificatorResponse>
            <request>
                <name>EHAK</name>
                <subset/>
                <from/>
                <max/>
            </request>
            <response>
                <classificators>
                    <item>
                        <l>37</l>
                        <objectclass>xrdLocality</objectclass>
                        <cn>Harju</cn>
                        <ehakType>0</ehakType>
                        <description>province</description>
                        <subclassificators>
                            <item>
                                <l>112</l>
                                <objectclass>xrdLocality</objectclass>
                                <cn>Aegviidu</cn>
                                <ehakType>1</ehakType>
                            </item>
                        </subclassificators>
                    </item>
                </classificators>
            </response>
        </xrd:loadClassificatorResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
7.1.9 Service: legacyXYZ

Query:

```xml
<?xml version="1.0" encoding="utf-8"?>
  <SOAP-ENV:Header xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
    <xrd:consumer>10239452</xrd:consumer>
    <xrd:producer>assert</xrd:producer>
    <xrd:userId>EE30101010007</xrd:userId>
    <xrd:id>3d0a5efdb4093dc04a8be68cd3227f009bd5c35d</xrd:id>
    <xrd:service>assert.legacy1.v1</xrd:service>
    <xrd:issue/>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <ns4:legacy1 xmlns:ns4="http://assert.ee.x-road.ee/producer/">
      <request>
        <item>assert.aspirant</item>
        <item>assert.houses</item>
        <item>assert.legacyQuery1</item>
        <item>assert.legacyQuery2</item>
        <item>assert.business</item>
      </request>
    </ns4:legacy1>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Response:

```xml
<?xml version="1.0" encoding="utf-8"?>
  <SOAP-ENV:Header xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
    <xrd:consumer>10239452</xrd:consumer>
    <xrd:producer>assert</xrd:producer>
    <xrd:userId>EE30101010007</xrd:userId>
    <xrd:id>3d0a5efdb4093dc04a8be68cd3227f009bd5c35d</xrd:id>
    <xrd:service>assert.legacy1.v1</xrd:service>
    <xrd:issue/>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <ns4:legacy1Response xmlns:ns4="http://assert.ee.x-road.ee/producer/">
      <request>
        <item>assert.aspirant</item>
        <item>assert.houses</item>
        <item>assert.legacyQuery1</item>
        <item>assert.legacyQuery2</item>
        <item>assert.business</item>
      </request>
    </ns4:legacy1Response>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```
7.1.10 Service with a MIME-attachment

Query:
POST /cgi-bin/consumer_proxy HTTP/1.0
User-Agent: PEAR-SOAP 0.7.5-devel
Host: 195.50.218.64
Content-Type: multipart/related; type=text/xml;
boundary="_b5a8d09e0ebe91be29def84633d6f6fc"
Content-Length: 1894
SOAPAction: ""

--_b5a8d09e0ebe91be29def84633d6f6fc
Content-Type:text/xml; charset="UTF-8"
Content-Transfer-Encoding:8bit

<?xml version="1.0" encoding="utf-8"?>
  <SOAP-ENV:Header xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
    <xrd:consumer>assert</xrd:consumer>
    <xrd:producer>land-cadastre</xrd:producer>
    <xrd:userId>EE30101010007</xrd:userId>
    <xrd:id>6cae248568b3db7e97ff784633d6f6fc</xrd:id>
    <xrd:service>land-cadastre.uploadMime.v1</xrd:service>
    <xrd:issue>Y-305-1</xrd:issue>
    <xrd:position>engineer</xrd:position>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <ns4:uploadMime xmlns:ns4="http://assert.x-road.ee/producer/>
      <request href="cid:b8f6da41bf27ba3095a2d1b7be6d802"/>
      <ns4:uploadMime/>
    </ns4:uploadMime>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Response:
HTTP/1.1 200 OK
Date: Tue, 02 Dec 2003 11:29:18 GMT

For public use
  <SOAP-ENV:Header xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
    <xrd:consumer>assert</xrd:consumer>
    <xrd:producer>land-cadastre</xrd:producer>
    <xrd:userId>EE:PIN:ABC4567</xrd:userId>
    <xrd:id>6cae248568b3db7e97ff784673a4d38c5906bee0</xrd:id>
    <xrd:service>land-cadastre.uploadMime.v1</xrd:service>
    <xrd:issue>Y-305-1</xrd:issue>
    <xrd:position>engineer</xrd:position>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <ns4:uploadMimeResponse xmlns:ns4="http://assert.ee.x-rd.net/producer/">
      <request>52ed0ffbf27fc34759dce05d0e7bed2302876ce</request>
      <response>
        <answer>-1</answer>
        <otherInfo href="cid:793340a8216da081f3d785bcc74c0f74"/>
      </response>
    </ns4:uploadMimeResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
<SOAP-ENV:Body>
   <xrd:syncNext>
      <request></request>
   </xrd:syncNext>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Response:
<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
   <SOAP-ENV:Body>
      <xrd:syncNextResponse>
         <response>1234567890</response>
      </xrd:syncNextResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Response, empty body:
<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
   <SOAP-ENV:Body>
      <xrd:syncNextResponse>
         <response></response>
      </xrd:syncNextResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

7.1.12 Service: asyncLAST

Query:
<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
   <SOAP-ENV:Body>
      <xrd:asyncLast>
         <request>database</request>
      </xrd:asyncLast>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Query, empty body:
<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
   <SOAP-ENV:Body>
      <xrd:asyncLast>
         <request></request>
      </xrd:asyncLast>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Response:
<?xml version="1.0" encoding="utf-8"?>
Response, empty body:

<?xml version="1.0" encoding="utf-8"?><SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Body>
    <xrd:asyncLastResponse>
      <response></response>
    </xrd:asyncLastResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

7.1.13 Service: listConsumers

Query:

<?xml version="1.0" encoding="utf-8"?><SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Body>
    <xrd:listConsumers/>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Response:

<?xml version="1.0" encoding="utf-8"?><SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Body>
    <xrd:listConsumersResponse>
      <response>
        <item>
          <name>it</name>
          <description>IT service</description>
        </item>
        <item>
          <name>institution</name>
          <description>Institution</description>
        </item>
      </response>
    </xrd:listConsumersResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

7.1.14 Service: listGroups

Query:

<?xml version="1.0" encoding="utf-8"?><SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
Response:

<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
  xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Body>
    <xrd:listGroupsResponse>
      <response>
        <item>
          <name>g1</name>
          <description>Test group 1</description>
        </item>
      </response>
    </xrd:listGroupsResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

7.1.15 Service: getProducerData

Query:

<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
  xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Body>
    <xrd:getProducerData>
      <request>database</request>
    </xrd:getProducerData>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Response:

<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
  xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Body>
    <xrd:getProducerDataResponse>
      <response>
        <item>
        </item>
      </response>
    </xrd:getProducerDataResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

7.1.16 Service: getConsumerData

Query:

<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
  xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
For public use

<SOAP-ENV:Body>
  <xrd:getConsumerData>
    <request>institution</request>
  </xrd:getConsumerData>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Response:

<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Body>
    <xrd:getConsumerDataResponse>
      <response>
        <item>
        </item>
      </response>
    </xrd:getConsumerDataResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

7.1.17 Service: getGroupData

Query:

<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Body>
    <xrd:getGroupData>
      <request>group</request>
    </xrd:getGroupData>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Response:

<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Body>
    <xrd:getGroupDataResponse>
      <response>
        <item>
          <name>carregister</name>
        </item>
        <item>
          <name>rr</name>
        </item>
      </response>
    </xrd:getGroupDataResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

7.1.18 Service: getProducerACL

Query:

<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
   <SOAP-ENV:Header>
      <xrd:consumer>institution</xrd:consumer>
      <xrd:producer>database</xrd:producer>
      <xrd:userId>EE37702211234</xrd:userId>
      <xrd:id>1234567890</xrd:id>
      <xrd:service>database.getProducerACL</xrd:service>
   </SOAP-ENV:Header>
   <SOAP-ENV:Body>
      <xrd:getProducerACL/>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Response:

<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
   <SOAP-ENV:Header>
      <xrd:consumer>institution</xrd:consumer>
      <xrd:producer>database</xrd:producer>
      <xrd:userId>EE37702211234</xrd:userId>
      <xrd:id>1234567890</xrd:id>
      <xrd:service>database.getProducerACL</xrd:service>
      <xrd:issue/>
   </SOAP-ENV:Header>
   <SOAP-ENV:Body>
      <xrd:getProducerACLResponse>
         <response>
            <item>
               <service>database.query1</service>
               <party>consumer1</party>
               <type>consumer</type>
            </item>
            <item>
               <service>database.query2</service>
               <party>consumer2</party>
               <type>consumer</type>
            </item>
         </response>
      </xrd:getProducerACLResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

7.1.19 Service: getServiceACL

Query:

<?xml version="1.0" encoding="utf-8" ?>
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
   <SOAP-ENV:Header>
      <xrd:consumer>institution</xrd:consumer>
      <xrd:producer>database</xrd:producer>
      <xrd:userId>EE37702211234</xrd:userId>
      <xrd:id>1234567890</xrd:id>
      <xrd:service>database.getServiceACL</xrd:service>
   </SOAP-ENV:Header>
   <SOAP-ENV:Body>
      <xrd:getServiceACL>
         <request>database.query1</request>
      </xrd:getServiceACL>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
Response:

<?xml version="1.0" encoding="utf-8"?>
                    xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Header>
    <xrd:consumer>institution</xrd:consumer>
    <xrd:producer>database</xrd:producer>
    <xrd:userId>EE37702211234</xrd:userId>
    <xrd:id>1234567890</xrd:id>
    <xrd:service>database.getServiceACL</xrd:service>
    <xrd:issue/>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <xrd:getServiceACLResponse>
      <response>
        <item>
          <party>institution1</party>
          <type>consumer</type>
        </item>
        <item>
          <party>institution2</party>
          <type>consumer</type>
        </item>
      </response>
    </xrd:getServiceACLResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

7.1.20 Service: getMethods

Query:

<?xml version="1.0" encoding="utf-8"?>
                    xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Header>
    <xrd:consumer>10239452</xrd:consumer>
    <xrd:producer>land-cadastre</xrd:producer>
    <xrd:userId>EE30101010007</xrd:userId>
    <xrd:id>411d675561409fed365ad8135f8210be07613da</xrd:id>
    <xrd:service>land-cadastre.getMethods</xrd:service>
    <xrd:issue/>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <xrd:getMethods/>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Response:

<?xml version="1.0" encoding="utf-8"?>
                    xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
  <SOAP-ENV:Header>
    <xrd:consumer>10239452</xrd:consumer>
    <xrd:producer>land-cadastre</xrd:producer>
    <xrd:userId>EE30101010007</xrd:userId>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <xrd:getMethods/>  
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
7.1.21 Service: logOnly

Query:
```xml
<?xml version="1.0" encoding="utf-8"?>
 xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
 <SOAP-ENV:Header>
  <xrd:consumer>institution</xrd:consumer>
  <xrd:producer>xrd</xrd:producer>
  <xrd:userId>EE37702211234</xrd:userId>
  <xrd:id>1234567890</xrd:id>
  <xrd:service>xrd.logOnly</xrd:service>
  <xrd:issue></xrd:issue>
 </SOAP-ENV:Header>
 <SOAP-ENV:Body>
  <xrd:logOnly>
   <request>1</request>
  </xrd:logOnly>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Response:
```xml
<?xml version="1.0" encoding="utf-8"?>
 xmlns:xrd="http://x-road.ee/xsd/x-road.xsd">
 <SOAP-ENV:Header>
  <xrd:consumer>institution</xrd:consumer>
  <xrd:producer>xrd</xrd:producer>
  <xrd:userId>EE37702211234</xrd:userId>
  <xrd:id>1234567890</xrd:id>
  <xrd:service>xrd.logOnly</xrd:service>
  <xrd:issue></xrd:issue>
 </SOAP-ENV:Header>
 <SOAP-ENV:Body>
  <xrd:logOnlyResponse/>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

7.2 Sample description of database services (WSDL)

7.2.1 WSDL in document/literal wrapped style

```xml
<?xml version="1.0" encoding="UTF-8"?>
```
<xrd:title xml:lang="et">Isiku andmed</xrd:title>
</appinfo>
</annotation>
</element>
</complexType>
</element>
</sequence>
</complexType>
</element>
</element>
<element name="isikOtsing">
<complexType>
<sequence>
<element name="request">
<complexType>
<sequence>
<element name="isikukood" type="string">
<annotation>
<appinfo>
<xrd:title xml:lang="en">Personal ID code</xrd:title>
<xrd:title xml:lang="et">Isikukood</xrd:title>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</element>
</sequence>
</complexType>
</element>
</sequence>
</complexType>
</element>
</sequence>
</complexType>
</element>
</element>
<element name="isikOtsingResponse">
<complexType>
<sequence>
<element name="request">
<complexType>
<sequence>
<element name="isikukood" nillable="true" type="string"/>
</sequence>
</complexType>
</element>
<element name="response">
<complexType>
<sequence>
<element name="faultCode" nillable="true" type="xrd:faultCode"/>
<element name="faultString" nillable="true" type="xrd:faultString"/>
<element name="isiku_andmed" nillable="true" type="tns:isik"/>
</sequence>
</complexType>
</element>
</sequence>
</complexType>
</element>
</element>
</element>
<element name="changeAddress">
<complexType>
<sequence>
<element name="request">
<complexType>
<sequence>
<element name="isikukood" type="string">
<annotation>
<appinfo>
<xrd:title xml:lang="et">Isikukood</xrd:title>
<xrd:title xml:lang="en">Personal ID code</xrd:title>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</element>
</sequence>
</complexType>
</element>
</sequence>
</complexType>
</element>
</sequence>
</complexType>
</element>
<element name="aadress" type="tns:aadress"/>
</sequence>
</complexType>
</element>
<element name="changeAddressResponse">
<complexType>
<sequence>
<element name="request">
<complexType>
<sequence>
<element name="isikukood" nillable="true" type="string"/>
</sequence>
</complexType>
</element>
<element name="response">
<complexType>
<sequence>
<element name="faultCode" nillable="true" type="xrd:faultCode"/>
<element name="faultString" nillable="true" type="xrd:faultString"/>
</sequence>
</complexType>
</element>
</sequence>
</complexType>
</element>
<element name="fileDownload">
<complexType>
<sequence>
<element name="request">
<complexType>
<sequence>
<element name="fileName" nillable="true" type="string">
<annotation>
<appinfo>
<xrd:title xml:lang="en">FileName</xrd:title>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</element>
</sequence>
</complexType>
</element>
<element name="fileDownloadResponse">
<complexType>
<sequence>
<element name="request">
<complexType>
<sequence>
<element name="fileName" nillable="true" type="string"/>
</sequence>
</complexType>
</element>
<element name="response">
<complexType>
<sequence>
<element name="faultCode" nillable="true" type="xrd:faultCode"/>
<element name="faultString" nillable="true" type="xrd:faultString"/>
<element name="file" nillable="true"/>
</sequence>
</complexType>
</element>
</sequence>
</complexType>
</element>
<element name="fileDownloadMTOMResponse">
<complexType>
<sequence>
<element name="request">
<complexType>
<sequence>
<element name="fileName" nillable="true" type="string"/>
</sequence>
</complexType>
</element>
<element name="response">
<complexType>
<sequence>
<element name="faultCode" nillable="true" type="xrd:faultCode"/>
<element name="faultString" nillable="true" type="xrd:faultString"/>
</sequence>
</complexType>
</element>
</sequence>
</complexType>
</element>
<element name="fileUploadMTOM">
<complexType>
<sequence>
<element name="request">
<complexType>
<sequence>
<element name="filemtom" type="base64Binary" xmime:expectedContentTypes="image/jpeg"/>
<element name="fileName" nillable="true" type="string">FileName</element>
</sequence>
</complexType>
</element>
<element name="response">
<complexType>
</sequence>
</complexType>
</element>
<element name="fileUploadMTOMResponse">
<complexType>
<sequence>
<element name="request">
<complexType>
<sequence>
<element name="fileName" nillable="true" type="string"/>
</sequence>
</complexType>
</element>
<element name="response">
<complexType>
</sequence>
</complexType>
</element>
<sequence>
  <element name="faultCode" nillable="true" type="xrd:faultCode"/>
  <element name="faultString" nillable="true" type="xrd:faultString"/>
</sequence>
</complexType>
</element>
<complexType name="isik">
  <all>
    <element name="eesnimi" type="string">
      <annotation>
        <appinfo>
          <xrd:title xml:lang="en">Given name</xrd:title>
          <xrd:title xml:lang="et">Eesnimi</xrd:title>
        </appinfo>
      </annotation>
    </element>
    <element name="perenimi" type="string">
      <annotation>
        <appinfo>
          <xrd:title xml:lang="en">Surname</xrd:title>
          <xrd:title xml:lang="et">Perenimi</xrd:title>
        </appinfo>
      </annotation>
    </element>
    <element name="isikukood" type="string">
      <annotation>
        <appinfo>
          <xrd:title xml:lang="en">Personal ID code</xrd:title>
          <xrd:title xml:lang="et">Isikukood</xrd:title>
        </appinfo>
      </annotation>
    </element>
    <element name="tookoht" type="tns:tookoht"/>
    <element name="aadress" type="tns:aadress">
      <annotation>
        <appinfo>
          <xrd:title xml:lang="en">Address</xrd:title>
          <xrd:title xml:lang="et">Aadress</xrd:title>
        </appinfo>
      </annotation>
    </element>
    <element name="telefon" type="string">
      <annotation>
        <appinfo>
          <xrd:title xml:lang="en">Phone number</xrd:title>
          <xrd:title xml:lang="et">Kontaktefon</xrd:title>
        </appinfo>
      </annotation>
    </element>
    <element name="epost" type="string">
      <annotation>
        <appinfo>
          <xrd:title xml:lang="en">E-mail</xrd:title>
          <xrd:title xml:lang="en">E-post</xrd:title>
        </appinfo>
      </annotation>
    </element>
  </all>
</complexType>
<complexType name="isik_short">
  <all>
    </all>
</complexType>
<element name="eesnimi" type="string">
  <annotation>
    <appinfo>
      <xrd:title xml:lang="en">Given name</xrd:title>
      <xrd:title xml:lang="et">Eesnimi</xrd:title>
    </appinfo>
  </annotation>
</element>

<element name="perenimi" type="string">
  <annotation>
    <appinfo>
      <xrd:title xml:lang="en">Surname</xrd:title>
      <xrd:title xml:lang="et">Perenimi</xrd:title>
    </appinfo>
  </annotation>
</element>

<element name="isikukood" type="string">
  <annotation>
    <appinfo>
      <xrd:title xml:lang="en">Personal ID code</xrd:title>
      <xrd:title xml:lang="et">Isikukood</xrd:title>
    </appinfo>
  </annotation>
</element>

<complexType name="aadress">
  <all>
    <element name="maakond" type="xrd:maakond">
      <annotation>
        <appinfo>
          <xrd:title xml:lang="en">State</xrd:title>
          <xrd:title xml:lang="et">Maakond</xrd:title>
        </appinfo>
      </annotation>
    </element>
    <element name="linnvald" type="string">
      <annotation>
        <appinfo>
          <xrd:title xml:lang="en">City</xrd:title>
          <xrd:title xml:lang="et">Linn/Vald</xrd:title>
        </appinfo>
      </annotation>
    </element>
    <element name="tanav" type="string">
      <annotation>
        <appinfo>
          <xrd:title xml:lang="en">Street</xrd:title>
          <xrd:title xml:lang="et">Tänav</xrd:title>
        </appinfo>
      </annotation>
    </element>
    <element name="majaNr" type="string">
      <annotation>
        <appinfo>
          <xrd:title xml:lang="en">House nr.</xrd:title>
          <xrd:title xml:lang="et">Maja nr.</xrd:title>
        </appinfo>
      </annotation>
    </element>
    <element name="korteriNr" type="integer">
      <annotation>
        <appinfo>
          <xrd:title xml:lang="en">Flat nr.</xrd:title>
          <xrd:title xml:lang="et">Korteri nr.</xrd:title>
        </appinfo>
      </annotation>
    </element>
  </all>
</complexType>
<part name="body" element="tns:isikuteList"/>
</message>
<message name="listMethods">
  <part name="body" element="xrd:listMethods"/>
</message>
<message name="loadClassification">
  <part name="body" element="xrd:loadClassification"/>
</message>
<message name="unitRepresent">
  <part name="body" element="xrd:unitRepresent"/>
</message>
<message name="fileDownload">
  <part name="body" element="tns:fileDownload"/>
</message>
<message name="fileDownloadMTOM">
  <part name="body" element="tns:fileDownloadMTOM"/>
</message>
<message name="testSystem">
  <part name="body" element="xrd:testSystem"/>
</message>
<message name="fileUpload">
  <part name="body" element="tns:fileUpload"/>
  <part name="file" type="xsd:base64Binary"/>
</message>
<message name="fileUploadMTOM">
  <part name="body" element="tns:fileUploadMTOM"/>
</message>
<message name="unitValidResponse">
  <part name="body" element="xrd:unitValidResponse"/>
</message>
<message name="fileDownloadResponse">
  <part name="body" element="tns:fileDownloadResponse"/>
  <part name="file" type="xsd:base64Binary"/>
</message>
<message name="fileDownloadMTOMResponse">
  <part name="body" element="tns:fileDownloadMTOMResponse"/>
</message>
<message name="isikOtsingResponse">
  <part name="body" element="tns:isikOtsingResponse"/>
</message>
<message name="testSystemResponse">
  <part name="body" element="xrd:testSystemResponse"/>
</message>
<message name="changeAddress">
  <part name="body" element="tns:changeAddress"/>
</message>
<message name="changeAddressResponse">
  <part name="body" element="tns:changeAddressResponse"/>
</message>
<portType name="TestPortType">
  <operation name="isikuteList">
    <documentation>
      <xrd:title xml:lang="et">Isikute nimekirja teenus</xrd:title>
      <xrd:title xml:lang="en">List of persons</xrd:title>
    </documentation>
    <input message="tns:isikuteList"/>
    <output message="tns:isikuteListResponse"/>
  </operation>
  <operation name="isikOtsing">
    <documentation>
      <xrd:title xml:lang="et">Isiku andmete otsimine isikukoodi järgi</xrd:title>
      <xrd:title xml:lang="en">Search person by Id-code</xrd:title>
    </documentation>
    <input message="tns:isikOtsing"/>
    <output message="tns:isikOtsingResponse"/>
  </operation>
</portType>

Protocol for Data Exchange Between Databases and Information Systems 9.5

28.05.2013 55/60
<operation name="listMethods">
  <documentation>
    <xrd:title xml:lang="en">listMethods</xrd:title>
  </documentation>
  <input message="tns:listMethods"/>
  <output message="tns:listMethodsResponse"/>
</operation>

<operation name="testSystem">
  <documentation>
    <xrd:title xml:lang="en">testSystem</xrd:title>
  </documentation>
  <input message="tns:testSystem"/>
  <output message="tns:testSystemResponse"/>
</operation>

<operation name="unitValid">
  <documentation>
    <xrd:title xml:lang="en">unitValid</xrd:title>
  </documentation>
  <input message="tns:unitValid"/>
  <output message="tns:unitValidResponse"/>
</operation>

<operation name="unitRepresent">
  <documentation>
    <xrd:title xml:lang="en">unitRepresent</xrd:title>
  </documentation>
  <input message="tns:unitRepresent"/>
  <output message="tns:unitRepresentResponse"/>
</operation>

<operation name="loadClassification">
  <documentation>
    <xrd:title xml:lang="en">loadClassification</xrd:title>
  </documentation>
  <input message="tns:loadClassification"/>
  <output message="tns:loadClassificationResponse"/>
</operation>

<operation name="fileDownload">
  <documentation>
    <xrd:title xml:lang="et">Faili allalaadimine</xrd:title>
    <xrd:title xml:lang="en">File download</xrd:title>
  </documentation>
  <input message="tns:fileDownload"/>
  <output message="tns:fileDownloadResponse"/>
</operation>

<operation name="fileDownloadMTOM">
  <documentation>
    <xrd:title xml:lang="et">Faili allalaadimine (MTOM)</xrd:title>
    <xrd:title xml:lang="en">File Download (MTOM)</xrd:title>
  </documentation>
  <input message="tns:fileDownloadMTOM"/>
  <output message="tns:fileDownloadMTOMResponse"/>
</operation>

<operation name="fileUpload">
  <documentation>
    <xrd:title xml:lang="et">Faili üleslaadimine</xrd:title>
    <xrd:title xml:lang="en">File Upload</xrd:title>
  </documentation>
  <input message="tns:fileUpload"/>
  <output message="tns:fileUploadResponse"/>
</operation>

<operation name="fileUploadMTOM">
  <documentation>
    <xrd:title xml:lang="et">Faili üleslaadimine (MTOM)</xrd:title>
    <xrd:title xml:lang="en">File Upload (MTOM)</xrd:title>
  </documentation>
  <input message="tns:fileUploadMTOM"/>
<operation name="changeAddress">
  <documentation>
    <xrd:title xml:lang="et">Aadressi muutmine</xrd:title>
    <xrd:title xml:lang="en">Change Address</xrd:title>
  </documentation>
  <input message="tns:changeAddress"/>
  <output message="tns:changeAddressResponse"/>
</operation>

<binding name="TestSoapBinding" type="tns:TestPortType">
  <soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="isikuteList">
    <soap:operation soapAction="" style="document"/>
    <xrd:version>v1</xrd:version>
    <input>
      <soap:body parts="body" use="literal"/>
      <soap:header message="tns:standardheader" part="consumer" use="literal"/>
      <soap:header message="tns:standardheader" part="producer" use="literal"/>
      <soap:header message="tns:standardheader" part="userId" use="literal"/>
      <soap:header message="tns:standardheader" part="id" use="literal"/>
      <soap:header message="tns:standardheader" part="service" use="literal"/>
    </input>
    <output>
      <soap:body parts="body" use="literal"/>
      <soap:header message="tns:standardheader" part="consumer" use="literal"/>
      <soap:header message="tns:standardheader" part="producer" use="literal"/>
      <soap:header message="tns:standardheader" part="userId" use="literal"/>
      <soap:header message="tns:standardheader" part="id" use="literal"/>
      <soap:header message="tns:standardheader" part="service" use="literal"/>
    </output>
  </operation>
  <operation name="isikOtsing">
    <soap:operation soapAction="" style="document"/>
    <xrd:version>v1</xrd:version>
    <input>
      <soap:body parts="body" use="literal"/>
      <soap:header message="tns:standardheader" part="consumer" use="literal"/>
      <soap:header message="tns:standardheader" part="producer" use="literal"/>
      <soap:header message="tns:standardheader" part="userId" use="literal"/>
      <soap:header message="tns:standardheader" part="id" use="literal"/>
      <soap:header message="tns:standardheader" part="service" use="literal"/>
    </input>
    <output>
      <soap:body parts="body" use="literal"/>
      <soap:header message="tns:standardheader" part="consumer" use="literal"/>
      <soap:header message="tns:standardheader" part="producer" use="literal"/>
      <soap:header message="tns:standardheader" part="userId" use="literal"/>
      <soap:header message="tns:standardheader" part="id" use="literal"/>
      <soap:header message="tns:standardheader" part="service" use="literal"/>
    </output>
  </operation>
  <operation name="listMethods">
    <soap:operation soapAction="" style="document"/>
    <input>
      <soap:body parts="body" use="literal" namespace="http://x-
        tee.riik.ee/xsd/xtee.xsd"/>
    </input>
    <output>
      <soap:body parts="body" use="literal" namespace="http://x-
        tee.riik.ee/xsd/xtee.xsd"/>
    </output>
  </operation>
  <operation name="testSystem">
    <soap:operation soapAction="" style="document"/>
  </operation>
</binding>
<soap:operation soapAction="" style="document"/>
<input>
<soap:body parts="body" use="literal"/>
</input>
<output>
<soap:body parts="body" use="literal"/>
</output>
</operation>

<operation name="unitValid">
<soap:operation soapAction="" style="document"/>
<input>
<soap:body parts="body" use="literal"/>
<soap:header message="tns:standardheader" part="consumer" use="literal"/>
<soap:header message="tns:standardheader" part="producer" use="literal"/>
<soap:header message="tns:standardheader" part="userId" use="literal"/>
<soap:header message="tns:standardheader" part="id" use="literal"/>
<soap:header message="tns:standardheader" part="service" use="literal"/>
</input>
<output>
<soap:body parts="body" use="literal"/>
<soap:header message="tns:standardheader" part="consumer" use="literal"/>
<soap:header message="tns:standardheader" part="producer" use="literal"/>
<soap:header message="tns:standardheader" part="userId" use="literal"/>
<soap:header message="tns:standardheader" part="id" use="literal"/>
<soap:header message="tns:standardheader" part="service" use="literal"/>
</output>
</operation>

<operation name="unitRepresent">
<soap:operation soapAction="" style="document"/>
<input>
<soap:body parts="body" use="literal"/>
<soap:header message="tns:standardheader" part="consumer" use="literal"/>
<soap:header message="tns:standardheader" part="producer" use="literal"/>
<soap:header message="tns:standardheader" part="userId" use="literal"/>
<soap:header message="tns:standardheader" part="id" use="literal"/>
<soap:header message="tns:standardheader" part="service" use="literal"/>
</input>
<output>
<soap:body parts="body" use="literal"/>
<soap:header message="tns:standardheader" part="consumer" use="literal"/>
<soap:header message="tns:standardheader" part="producer" use="literal"/>
<soap:header message="tns:standardheader" part="userId" use="literal"/>
<soap:header message="tns:standardheader" part="id" use="literal"/>
<soap:header message="tns:standardheader" part="service" use="literal"/>
</output>
</operation>

<operation name="loadClassification">
<soap:operation soapAction="" style="document"/>
<input>
<soap:body parts="body" use="literal"/>
<soap:header message="tns:standardheader" part="consumer" use="literal"/>
<soap:header message="tns:standardheader" part="producer" use="literal"/>
<soap:header message="tns:standardheader" part="userId" use="literal"/>
<soap:header message="tns:standardheader" part="id" use="literal"/>
<soap:header message="tns:standardheader" part="service" use="literal"/>
</input>
<output>
<soap:body parts="body" use="literal"/>
<soap:header message="tns:standardheader" part="consumer" use="literal"/>
<soap:header message="tns:standardheader" part="producer" use="literal"/>
<soap:header message="tns:standardheader" part="userId" use="literal"/>
<soap:header message="tns:standardheader" part="id" use="literal"/>
<soap:header message="tns:standardheader" part="service" use="literal"/>
</output>
</operation>

<operation name="fileDownload"/>