

FutureFarm WP4 – Rule server interface

Authors: Raimo Nikkilä, Edward Nash

Last edit: 17.8.2009

Introduction

This document

This document specifies the service-oriented interface for a rule server for the automated distribution of rules representing encoded agricultural standards. The scope of this document is restricted to the interface of the rule server with no consideration on the encoding of the rules, apart from the assumption that the rules can be represented as parts of an XML-document. The rules are assumed to be grouped as standards and the rule server interface is provided on various levels of abstraction as some operations are illegal for individual rules but legal for standards (a group of rules). Most of the meta-information associated with the rules is also covered in this document.

The specification in this document is a partial fulfillment of the requirements identified for the complete system by the FutureFarm project.

Purpose of the rule server

A rule server maintains a set of standards containing the rules that represent encoded agricultural standards. A rule server provides an interface with operations for searching, getting, adding, removing and updating rules. The rule servers are discovered through the use of catalogue servers which are described in another document produced by this project. Several rule servers are expected to exist as one rule is likely to cover only the standards provided by some organisation or group of organisations. However, unlike the catalogue servers, rule servers are autonomous in the sense that they need not know of other rule servers or catalogue servers.

Technical description

Overview

This specification is based on REST and the rationale for choosing REST over the more formal SOA protocols is the simplicity of the rule server. The functionality of the rule servers does not warrant the use of the considerably more complicated SOA family of protocols as full functionality can be achieved with much greater simplicity and without any loss of features.

This specification covers the basic operations of the catalogue server as operations of REST and the document format and schema used in the communication in XSD format. Unlike in the catalogue server specification, XSD is used instead of DTD due to the amount of several imported XSD schemata. An introduction to REST can be found in the specification of the catalogue servers.

Detailed technical description

General communication

All communication occurs following REST. The Resource:s provided are:

/standards

/serverInformation

Which contain the standards and information on the rule server, of which the latter is a static Resource: that can only be read but offers no functionality. The Resource: for standards can be used for entire standards or individual rules. For example the rule with unique identifier “120” of the standard with unique identifier “145” would be obtained in the following way:

/standards/145/120

An additional Resource: is defined for the purpose of searching and filtering standards:

/standards/searchStandards

Note that there is no defined way to alter an existing standard, such as changing or updating an individual rule of a standard. Only changes allowed for standards are adding, replacing and deleting the entire standard.

Another additional Resource: exists to fetch all rules of a standard which affect some particular field operation.

/standards/145/operation

This Resource: is the only operation apart from fetching a single rule that operates on individual rules instead of complete standards. This also implies that the string “operation” is an invalid identifier for a rule.

For all queries and replies an additional attribute “version” must exist for the root element indicating the version of the used syntax.

Values and identifiers

All values and identifiers are currently unbound and can be any suitable strings within the XML replies and queries. However, as the development of the system progresses these values can, and should be, further restricted. Having these values bound at this point of development would only result in further changes to the specification in the future.

Fetching standards

This operation returns one standard with all the information associated to the standard.

Corresponding REST operation: GET

Resource: /standards/{standardID}

Query: None

Successful reply: <standard>-element

Possible error codes: 200, 201

Listing standards

This operation lists all known standards in a shortened format and without any rules.

Corresponding REST operation: GET

Resource: /standards

Query: None

Successful reply: <standards>-element

Possible error codes:

Searching standards

This operation lists all standards matching a given query. Empty elements match any entries, thus an empty <search_standards>-element lists all known standards. Standards are listed as in the listing operation meaning that rules are not included in the results.

Corresponding REST operation: GET

Resource: /standards/searchStandards

Query: <search_standards>-element

Successful reply: <standards>-element with all standards matching the given query

Possible error codes: 202

Adding a standard

This operation adds one standard to the rule server or replaces an existing standard entry.

Corresponding REST operation: PUT

Resource: /standards/{standardID}

This is a privileged operation requiring user authentication

Query: <add_standard>-element

Successful reply: <confirmation>-element

Possible error codes: 201, 204, 205, 206, 207, 207, 301, 302, 303

Deleting a standard

This operation deletes one standard.

Corresponding REST operation: DELETE

Resource: /standards/{standardID}

This is a privileged operation requiring user authentication

Query: None

Successful reply: <confirmation>-element

Possible error codes: 200, 201

Fetching individual rules

This operation returns one rule of a standard. The information returned for the rule is identical to the representation of the rule when listed as a part of a standard.

Corresponding REST operation: GET

Resource: /standards/{standardID}/{ruleID}

Query: None

Successful reply: <rule>-element

Possible error codes: 200, 201, 300, 301

Fetching all rules for an operation

This operation returns all rules of a standard which are tagged to affect some specified field operation.

Corresponding REST operation: GET

Resource: /standards/145/operation/<operation>

Query: None

Successful reply: <standards>-element

Possible error codes: 200, 201, 209

Obtaining server information

This operation returns information on the rule server.

Corresponding REST operation: GET

Resource: /serverInformation

Query: None

Successful reply: <server_information>-element

Possible error codes:

Security considerations

Sufficient degree of security can be achieved through the use of the HTTPS protocol and client authentication. The service can be provided in both HTTP and HTTPS for most operations of searching and listing but for the rare operations of adding and deleting, HTTPS should be preferred. Authentication should occur through normal HTTP authentication using the basic access authentication protocol as detailed in RFC 2617.

Discussion

Like the catalogue server, the rule server can easily be implemented with a REST based-architecture. Most of the error codes and element names in the rule server specification match those of the catalogue server for ease of implementation and similarity of the two services. Most of the values, especially the rule body, are currently unspecified and will have to be replaced with proper value representations in the future.

References

Fielding, Roy Thomas (2000) ([HTML](#)), *Architectural Styles and the Design of Network-based Software Architectures*, Doctoral dissertation, University of California, Irvine, <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>

Pautasso, Cesare; Zimmermann, Olaf; Leymann, Frank (2008-04), "[RESTful Web Services vs. Big Web Services: Making the Right Architectural Decision](#)" ([HTML](#)), *17th International World Wide Web Conference (WWW2008)* (Beijing, China)

Appendix (XSD schema definitions)

Replies

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ff="http://www.futurefarm.eu/standards"
xmlns:rs="http://www.futurefarm.eu/ruleserver"
targetNamespace="http://www.futurefarm.eu/ruleserver"
elementFormDefault="qualified">
  <xs:import namespace="http://www.futurefarm.eu/standards"
schemaLocation="http://goblin.tkk.fi/~rnikkila/schema.xsd">
    <xs:annotation>
      <xs:documentation xml:lang="en">FutureFarm encoding of
standards</xs:documentation>
    </xs:annotation>
  </xs:import>
  <xs:element name="ff_rules_reply">
    <xs:complexType>
      <xs:choice>
        <xs:element ref="ff:AgriStandard"/>
        <xs:element ref="rs:standards"/>
        <xs:element ref="rs:error"/>
        <xs:element ref="rs:server_information"/>
        <xs:element ref="rs:confirmation"/>
      </xs:choice>
      <xs:attribute name="version" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="standards">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="ff:AgriStandard" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="error">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="rs:query"/>
        <xs:element ref="rs:code"/>
        <xs:element ref="rs:message"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

```

</xs:element>
<xs:element name="query" type="xs:string"/>
<xs:element name="code" type="xs:string"/>
<xs:element name="message" type="xs:string"/>
<xs:element name="server_information" type="xs:string"/>
<xs:element name="confirmation">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="rs:query"/>
      <xs:element ref="rs:message"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>

```

Queries

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:rs="http://www.futurefarm.eu/ruleserver"
  xmlns:ff="http://www.futurefarm.eu/standards"
  targetNamespace="http://www.futurefarm.eu/ruleserver"
  elementFormDefault="qualified">
  <xs:import namespace="http://www.futurefarm.eu/standards"
    schemaLocation="http://goblin.tkk.fi/~rnikkila/schema.xsd">
    <xs:annotation>
      <xs:documentation xml:lang="en">FutureFarm encoding of
standards</xs:documentation>
    </xs:annotation>
  </xs:import>
  <xs:element name="ff_rules_request">
    <xs:complexType>
      <xs:choice>
        <xs:element ref="ff:AgriStandard"/>
        <xs:group ref="rs:search_standards"/>
      </xs:choice>
      <xs:attribute name="version" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:group name="search_standards">
    <xs:sequence>
      <xs:element ref="rs:id" minOccurs="0"/>
      <xs:element ref="rs:name" minOccurs="0"/>
      <xs:element ref="rs:publisher" minOccurs="0"/>
      <xs:element ref="rs:region" minOccurs="0"/>
    </xs:sequence>
  </xs:group>

```

```

        <xs:element ref="rs:type" minOccurs="0"/>
        <xs:element ref="rs:date" minOccurs="0"/>
    </xs:sequence>
</xs:group>
<xs:element name="id" type="xs:string"/>
<xs:element name="name" type="xs:string"/>
<xs:element name="publisher" type="xs:string"/>
<xs:element name="region" type="xs:string"/>
<xs:element name="type" type="xs:string"/>
<xs:element name="date" type="xs:string"/>
</xs:schema>

```

Valid error codes and their HTTP codes

Specific Error code	HTTP Response code	Description
100	401	Authentication required
101	403	Authentication error or insufficient privileges
102	400	Invalid syntax in query
103	400	Invalid XML structure in query
104	400	Invalid query
110	500	General system error
200	400	Unknown identifier for standard
201	400	Invalid identifier for standard
202	400	Invalid search
204	400	Invalid name when adding a standard
205	400	Invalid publisher when adding a standard
206	400	Invalid region when adding a standard
207	400	Invalid type when adding a standard
208	400	Invalid date when adding a standard
209	400	Invalid operation
300	400	Unknown identifier for a rule
301	400	Invalid identifier for a rule
302	400	Invalid rule information
303	400	Invalid rule body