IVOA Support Interfaces: Mandatory Interfaces
Version 0.26

IVOA Working Draft 2007 May 3

This version:

http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices /VOSupportInterfacesMandatory-0.26.pdf

Previous versions:

http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices /VOSupportInterfacesMandatory-0.25.pdf

http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices /VOSupportInterfaces-0.24.pdf

http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices /VOSupportInterfaces-0.23.pdf

http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices /VOSupportInterfaces-0.22.pdf

http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices /VOSupportInterfaces-0.21.pdf

http://www.ivoa.net/internal/IVOA/VOSupportInterfaces-0.2.pdf

http://www.ivoa.net/internal/IVOA/StandardInterfaces-0.1.pdf

Editor for this version: Guy Rixon

Authors: Web and Grid Services Working Group

Please send comments to: grid@ivoa.net

Abstract

This document describes the minimum required interface to participate in the IVOA as a web service.

Status of this document

This is an Internal Working Draft. No versions of this document have been released outside the working group.

This is an IVOA Working Draft for review by IVOA members and other interested parties. It is a draft document and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use IVOA Working Drafts as reference materials or to cite them as other than "work in progress." A list of current IVOA Recommendations and other technical documents can be found at http://www.ivoa.net/docs/.

Acknowledgments

This work is based on discussions and actions from the 2003 IVOA meeting in Strasbourg and further discussions on registry functionality at JHU late in 2003. Later inputs came from a local meeting at JHU Sept. 2004. William O' Mullane and Ani Thakar were the editors and primary authors for these early versions.

The decision to split the interfaces into a mandatory set and optional logging interfaces was taken by GWS-WG at the IVOA meeting of May 2006.

Contents

1 Introduction

Much of the virtual observatory is made up of web services in SOAP and REST forms. It is felt there are some basic functions that all these services should provide in order to support management of the virtual observatory.

These standard interfaces are mandated by the VO basic profile for SOAP services and hence should appear in all SOAP services, using the standard WSDL extract in this specification. For REST services, the requirement for the support interfaces is stated in the specification for each kind of service. A contract for a REST service may specify extra constraints (e.g. on the form of the URIs) for the support interfaces.

In the normal requirements manner the words “should” and “shall” are used to convey the level of necessity of the interface.

2 Standard VO Interfaces

The standard interfaces all return metadata without changing the state of the service to which they apply. They may be implemented in any of three ways.

1. As extra SOAP operations on an existing SOAP endpoint of the service to which they apply.
2. As SOAP operations on a separate SOAP endpoint.
3. As web resources with distinct URLs, without using the SOAP protocol. This is the 'REST binding' for the standard interfaces.

Where the support interfaces are implemented as SOAP operations, they shall conform to the WSDL given as appendix 1.

In the REST binding, the support interfaces shall have distinct URLs in the HTTP scheme and shall be accessible by the GET operation in the HTTP protocol. The response to an HTTP POST, PUT or DELETE to these resources is not defined by this specification. However, if an implementation has no special action to perform for these requests, the normal response would be a 405 “Method not allowed” error.

The endpoints and interface types for the support interfaces shall be defined in the service's registration using one Capability element for each interface. The *standardId* values for these Capabilities are TBD.

When using the REST binding, any HTTP URI may be used. The client must find the appropriate URI from the registration and, in general, should not try and infer the URI from any other URI for that service. However, standards for specific services may put extra constraints on the form of the URI.

2.1 Registration Metadata

This interface provides the metadata which the author of the service wishes to be entered in the IVO registry. This interface supports the concept that the service is the authoritative source for its own metadata and the registry is a cache for those metadata. Ideally, registry implementations should be made to download the registration metadata from the services rather than requiring it to be re-entered locally.

The metadata shall be represented as an XML document which contains a sequence of zero or more VOResource elements. The case of zero VOResource elements arises when the service author does not wish the service to be registered at all; more normally, there will be at least one VOResource.

In the SOAP bindings, the registration metadata shall be obtained from the *getRegistration* operation. The date and time on which the metadata last changed shall be obtained from the *registrationChangedOn* operation.

In the REST binding, the registration metadata shall be a single web-resource with a registered URI. The data and time at which the metadata last changed shall be obtained the *Last-Modified* HTTP-header sent in the reponse to a GET or HEAD request to the registered URI. There is no separate resource in the REST binding that corresponds to r*egistrationChangedOn* in the SOAP binding.

All VO services should provide this interface.

2.2 Full metadata

This interface supplies a superset of the service registration, possibly including metadata which the service author does not wish to appear in the IVO registry. These metadata may be excluded from registration by policy or because they change more rapidly than the registry's harvesting system can track.

The metadata shall be represented as an XML document which contains a sequence of one or more VOResource elements.

In the SOAP bindings, the full metadata shall be obtained from the *getMetdata* operation. The date and time on which the metadata last changed shall be obtained from the *metadataChangedOn* operation.

In the REST binding, the metadata shall be a single web-resource with a registered URI. The data and time at which the metadata last changed shall be obtained the *Last-Modified* HTTP-header sent in the response to a GET or HEAD request to the registered URI. There is no separate resource in the REST binding that corresponds to *metadataChangedOn* in the SOAP binding.

All VO services should provide this interface.

2.3 Availability

This interface indicates whether the service is operable and the reliability of the service for extended and scheduled requests.

The availability shall be represented as an XML document in which the root element is *{http://www.ivoa.net/xml/Availability/v0.25}availability*. (Note that this is the same XML namespace specified in v0.25 of this standard.) This element contains child elements providing the following information.

- uptime - the up time of the service
- validTo - how long it believes this will be valid i.e. next scheduled downtime.
- contactDetails – name, email and phoneNumber of a person to contact if there is a problem.

The XML document shall conform to the schema given in appendix 2 of this specification.

When reporting availability, the service should do a good check on its underlying parts to see if it is still operational and not just make a simple return from a web server, e.g., if it relies on a database it should check that the database is still up.

If a service wishes to be on-line but unavailable for work (e.g. when a service with a work queue intends to shut down after draining the queue) then the service should return the current date/time for the *validTo* element of the availability document.

Ultimately some portals may track these heartbeats and compile uptime statistics. With the location we could have 3D global maps of services and availability.

In the SOAP bindings, the availability shall be obtained from the *getAvailability* operation.

In the REST binding, the availability shall be a single web-resource with a registered URI.

All VO services shall provide this interface.

3 Changes from previous versions

- Added the *getMetadata* and *metadataChangedOn* interfaces in addition to the existing *getRegistration* and *registrationChangedOn*. This is intended to make VOSI a better fit with emerging standards in DAL-WG and also to better support services with volatile state, such as ADQL interfaces for changeable data collections.
- Noted that the REST binding for *registrationChangedOn* is really the HTTP header Last-Modified in the response for the registration itself.
- Tightened the language specifying the REST binding.
- Changed the emphasis in the descriptive text from the SOAP binding to the REST binding.
- Changed the WSDL namespace to *http://www.ivoa.net/xml/VOSupportInterfacesMandatory/v0.26*. Note that the availability namespace is unchanged since v0.25.
- In the WSDL, added *minOccurs="0"* for the VOResource elements in *getRegistration*, matching the tighter definition in the text.

4 References

[RM] Resource Metadata for the Virtual Observatory; Bob Hanisch et al.; http://www.ivoa.net/Documents/WD/ResMetadata/WD-ResMetadata.html

[LOG] W3C Logging ; http://www.w3.org/Daemon/User/Config/Logging.html and http://www.w3.org/TR/WD-logfile.html

[VOS] IVOA VOStore; Dave Morris et al.; http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices/vostore-0.18.pdf

Appendix 1: WSDL for support interfaces

This WSDL governs the presentation of the support interfaces when implemented as a SOAP service.

```
<wsdl:definitions
 targetNamespace="http://www.ivoa.net/xml/VOSupportInterfacesMandatory/
v0.26"
 xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:tns="http://www.ivoa.net/xml/VOSupportInterfacesMandatory/v0.26"
 xmlns:avail="http://www.ivoa.net/xml/Availability/v0.25"
 xmlns:vor="http://www.ivoa.net/xml/VOResource/v1.0">

  <wsdl:types>
    <xs:schema
 targetNamespace="http://www.ivoa.net/xml/VOSupportInterfacesMandatory/
v0.26">
      <xs:import namespace="http://www.ivoa.net/xml/Availability/v0.25"
        schemaLocation="Availability-v0.25.xsd"/>
      <xs:import namespace="http://www.ivoa.net/xml/VOResource/v1.0"
         schemaLocation="http://www.ivoa.net/xml/VOResource/v1.0"/>

      <xs:element name="getAvailability">
        <xs:complexType/>
      </xs:element>

      <xs:element name="getAvailabilityResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="availability" type="avail:Availability"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="getRegistration">
        <xs:complexType/>
      </xs:element>

      <xs:element name="getRegistrationResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="resource" type="vor:Resource"
                minOccurs=”0” maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>

      <xs:element name="registrationChangedOn">
        <xs:complexType/>
      </xs:element>

      <xs:element name="registrationChangedOnResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="changedOn" type="xs:dateTime"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>

     <xs:element name="getMetadata">
        <xs:complexType/>
      </xs:element>

      <xs:element name="getMetadataResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="resource" type="vor:Resource"
                maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
```

```
      <xs:element name="metadataChangedOn">
        <xs:complexType/>
      </xs:element>

      <xs:element name="metadataChangedOnResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="changedOn" type="xs:dateTime"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:schema>

  </wsdl:types>

  <wsdl:message name="getAvailability">
    <wsdl:part name="getAvailability" element="tns:getAvailability"/>
  </wsdl:message>

  <wsdl:message name="getAvailabilityResponse">
    <wsdl:part name="getAvailabilityResponse"
element="tns:getAvailabilityResponse"/>
  </wsdl:message>

  <wsdl:message name="getRegistration">
    <wsdl:part name="getRegistration" element="tns:getRegistration"/>
  </wsdl:message>

  <wsdl:message name="getRegistrationResponse">
    <wsdl:part name="getRegistrationResponse"
       element="tns:getRegistrationResponse"/>
  </wsdl:message>

  <wsdl:message name="registrationChangedOn">
    <wsdl:part name="registrationChangedOn"
       element="tns:registrationChangedOn"/>
  </wsdl:message>
  <wsdl:message name="registrationChangedOnResponse">
    <wsdl:part name="registrationChangedOnResponse"
element="tns:registrationChangedOnResponse"/>
  </wsdl:message>

  <wsdl:portType name="registration">
    <wsdl:operation name="getRegistration">
      <wsdl:input message="tns:getRegistration"/>
      <wsdl:output message="tns:getRegistrationResponse"/>
    </wsdl:operation>
    <wsdl:operation name="registrationChangedOn">
      <wsdl:input message="tns:registrationChangedOn"/>
      <wsdl:output message="tns:registrationChangedOnResponse"/>
    </wsdl:operation>
  </wsdl:portType>

  <wsdl:portType name="availability">
    <wsdl:operation name="getAvailability">
      <wsdl:input message="tns:getAvailability"/>
      <wsdl:output message="tns:getAvailabilityResponse"/>
    </wsdl:operation>
  </wsdl:portType>

  <wsdl:binding name="registrationSoap" type="tns:registration">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
      style="document"/>
    <wsdl:operation name="getRegistration">
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="registrationChangedOn">
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>

  <wsdl:binding name="availabilitySoap" type="tns:availability">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
      style="document"/>
    <wsdl:operation name="getAvailability">
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>

</wsdl:definitions>
```

Appendix 2: XML schema for getAvailability

The following normative schema defines the XML document returned by the getAvailability interface.

```
<xsd:schema
  targetNamespace="http://www.ivoa.net/xml/Availability/v0.25"
  xmlns:tns="http://www.ivoa.net/xml/Availability/v0.25"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:vr="http://www.ivoa.net/xml/VOResource/v0.10"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">

  <xsd:import namespace="http://www.ivoa.net/xml/VOResource/v0.10"
    schemaLocation="http://www.ivoa.net/xml/VOResource/v0.10"/>

  <xsd:element name="availability" type="tns:Availability"/>

  <xsd:complexType name="Availability">
    <xsd:all>

      <!-- Indicates whether the service is currently available. -->
      <xsd:element name="available" type="xsd:boolean"/>

      <!-- Time since last restart of service. -->
      <xsd:element name="uptime" type="xsd:duration"/>

      <!-- Next scheduled down-time, if known.
           Set nil=true if the value is unknown. -->
      <xsd:element name="validTo" type="xsd:dateTime" nillable="true"/>

      <!-- Contact details for the service owner or operator. -->
      <xsd:element name="contact" type="vr:Contact"/>
    </xsd:all>
  </xsd:complexType>

</xsd:schema>
```