IVOA Support Interfaces
Version 1.00

IVOA Working Draft 2008 October 23

This version:

http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices/VOSI-0.5.pdf

Previous versions:

http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices/VOSI-0.5.pdf
http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices/VOSupportInterfacesMandatory-0.4.pdf
http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices/VOSupportInterfacesMandatory-0.3.pdf
http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices/VOSupportInterfacesMandatory-0.26.pdf
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 IVOA Working Draft. It is the first release of this document outside the GWS 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 Capability metadata

This interface provides the service metadata in the form of a list of Capability descriptions. Each of these descriptions is an XML element that

- states that the service provides a particular, IVOA-standard function;
- lists the interfaces for invoking that function;

- records any details of the implementation of the function that are not fixed in the standard for that function.

E.g., one capability might describe a cone-search function and another the table-access-protocol implementation; these two might well be apply to the same service.

An entry for a service in the resource registry – i.e. its VOResource – contains the Dublin-core resource metadata (identifier, curation information, content description etc.) followed by the service's capability descriptions. Effectively, the resource metadata describe the service to human users and the capability list describes it to software. Therefore, the capability list alone has two uses.

- It may be read by a client application to find out how to drive the service. This presumes that the service has been already been selected and the VOSI endpoint located.
- It may be read by the registry itself to compile the registry entry for the service. In this case, the resource metadata are entered into the registry directly and the service metadata are then read from the service. Since the service implementation usually knows its capabilities, this removes the need for a human to type them into the registry.

The service metadata shall be represented as an XML document which contains a sequence of one or more elements of type *{http://www.ivoa.net/xml/VOResource/v1.0}Capability* or sub-types thereof.

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

In the REST binding, the service 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 *capabilitiesChangedOn* in the SOAP binding.

All VO services should provide this interface.

2.2 Non-service metadata (non-normative description)

There may be other metadata associated with a service than the capability metadata described above.

- Every service has the Dublin-core “resource” metadata.
- Some services are associated with registered applications.
- Some services are associated with registered data collections.
- Some services are associated with custom views of data which must the client must understand in order to use the service.

None of these are explicitly provided for in this version of VOSI. Some might be covered in later versions of VOSI.

2.3 Availability metadata

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/v0.4}availability*. This element shall contain child elements providing the following information.

- *available* – whether the service is currently accepting requests;
- *upSince* – duration for which the service has been continuously available
- *downAt* – the instant at which the service is next scheduled to be unavailable
- *backAt* – the instant at which the service is scheduled to become available again after down time;
- *note* – textual note, e.g. explaining the reason for unavailaility.

The elements *upSince*, *downAt*, *backAt* and *notes* are optional. The *available* element is mandatory. There may be more than one *note* element.

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 any of these check fail, the service should set *available* to false in the availability output.

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 set *available* to false.

There are no special elements in the availability document for the contact details of the service operator. These details may be given as a *note* element if they are known to the service.

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.

2.4 Registration of VOSI endpoints

The endpoints for the service and availability metadata shall be included in the registration of each service that provides them.

An availability endpoint shall be represented by an element named *capability,* of type *{http://www.ivoa.net/xml/VOResource/v1.0}Capability.* The value of the *standardID*

attribute of the *capability* shall be *ivo://ivoa.net/std/VOSI#availability* (sic; note the use of the document fragment to identify one part of the VOSI standard).

A capabilities endpoint should be represented an element named *capability,* of type *{http://www.ivoa.net/xml/VOResource/v1.0}Capability.* If such an *capability* is provided then the value its *standardID* attribute must be *ivo://ivoa.net/std/VOSI#capabilities.*

3 Changes from previous versions

This version is identical to version 0.5 previously circulated within GWS-WG.

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/
v0.4"
 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/v0.4"
 xmlns:avail="http://www.ivoa.net/xml/Availability/v0.4"
 xmlns:vor="http://www.ivoa.net/xml/VOResource/v1.0">


  <wsdl:types>
    <xs:schema
 targetNamespace="http://www.ivoa.net/xml/VOSupportInterfacesMandatory/
v0.3">
      <xs:import namespace="http://www.ivoa.net/xml/Availability/v0.4"
        schemaLocation="Availability-v0.4.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="getCapabilities">
        <xs:complexType/>
      </xs:element>

      <xs:element name="getCapabilitiesResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="resource" type="vor:Capability"
                maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>

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

      <xs:element name="capabilitiesChangedOnResponse">
        <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="getCapabilities">
```

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

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

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

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

  <wsdl:portType name="capabilities">
    <wsdl:operation name="getCapbilities">
      <wsdl:input message="tns:getCapabilities"/>
      <wsdl:output message="tns:getCapabilitiesResponse"/>
    </wsdl:operation>
    <wsdl:operation name="capabilitiesChangedOn">
      <wsdl:input message="tns:capabilitiesChangedOn"/>
      <wsdl:output message="tns:capabilitiesChangedOnResponse"/>
    </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="capabilitiesSoap" type="tns:capabilities">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
```

```
      style="document"/>
    <wsdl:operation name="getCapabilities">
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="capabilitiesChangedOn">
      <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/v0.4"
 xmlns:tns="http://www.ivoa.net/xml/Availability/v0.4"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 elementFormDefault="qualified"
 attributeFormDefault="unqualified">

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

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

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

   <xsd:element name="upSince" type="xsd:dateTime" minOccurs="0">
    <xsd:annotation>
     <xsd:documentation>The instant at which the service last became
     available.</xsd:documentation>
    </xsd:annotation>
   </xsd:element>

   <xsd:element name="downAt" type="xsd:dateTime" minOccurs="0">
    <xsd:annotation>
     <xsd:documentation>The instant at which the service is next scheduled to become
      unavailable.</xsd:documentation>
    </xsd:annotation>
   </xsd:element>
```

```
    <xsd:element name="backAt" type="xsd:dateTime" minOccurs="0">
      <xsd:annotation>
        <xsd:documentation>The instant at which the service is scheduled to become available again
          after a period of unavailability.</xsd:documentation>
      </xsd:annotation>
    </xsd:element>

    <xsd:element name="note" type="xsd:string" minOccurs="0" maxOccurs="unbounded">
      <xsd:annotation>
        <xsd:documentation>A textual note concerning availability.</xsd:documentation>
      </xsd:annotation>
    </xsd:element>

  </xsd:sequence>
 </xsd:complexType>

</xsd:schema>
```