



International
Virtual
Observatory
Alliance

IVOA Support Interfaces

Version 1.00-20090925

IVOA Working Draft 2009 August 25

This version:

<http://www.ivoa.net/Documents/WD/GWS/VOSI-20090825.doc>

Latest version:

<http://www.ivoa.net/Documents/latest/VOSI.html>

Previous version:

<http://www.ivoa.net/Documents/WD/GWS/VOSI-20081030.doc>

Working Group:

[Grid & Web Services](#)

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](http://www.ivoa.net/docs/) 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

Abstract.....	2
Status of this document.....	2
Acknowledgments.....	2
Contents.....	2
1 Introduction.....	3
2 Standard VO interfaces.....	3
2.1 Capability metadata.....	3
2.2 Non-service metadata (non-normative description).....	4
2.3 Availability metadata.....	5
2.4 Table metadata.....	5
2.5 Registration of VOSI endpoints.....	6
3 Changes from previous versions	6
Appendix 1: WSDL for support interfaces.....	7
Appendix 2: XML schema for getAvailability.....	11

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 could, in principle be implemented in any of three ways.

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

This standard requires the REST binding of VOSI even when applied to services that otherwise use SOAP. No details of the SOAP binding are given in this version of the standard.

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 given in section 2.4.

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

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 REST binding, the service metadata shall be a single web-resource with a registered URL. 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.

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.

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 unavailability.

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 1 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 REST binding, the availability shall be a single web-resource with a registered URIL

All VO services shall provide this interface.

2.4 Table metadata

Some services deal with tabular data. These data may be the target of ADQL queries, as in the Table Access Protocol, or they may be the output of other operations, as in SIAP queries. In each case, it is useful if the service describes the details of the tables concerned. It is more useful if this description can be captured in the resource registry.

The *VODataService* standard defines XML elements for describing a set of tables. These elements can be included in a resource document for a service.

A service which uses tables in its interface should define a VOSI endpoint from which table metadata can be read. The table metadata shall be represented as an XML document of which the root element is of type

{http://www.ivoa.net/xml/VODataService/v1.1}TableSet. This element may contain any mix of elements allowed by the VODataService XML-schema.

In the REST binding, the table metadata shall be a single web-resource with a registered URL.

2.5 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*.

A tables 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#tables*.

3 Changes from previous versions

The REST binding is made mandatory for all kinds of service. Details of the SOAP binding, including its WSDL contract, are removed.

The definition of the root element for the table-metadata document is corrected. Instead of requiring the *tableset* element from *VODataService* 1.1 (which element does not exist in that schema), the text now requires an element of *type TableSet*.

Appendix 1: XML schema availability

The following normative schema defines the XML document returned by the avai resource.

```
<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:sequence>
  </xsd:complexType>
</xsd:schema>
```

```
<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>
```