IVOA Support Interfaces: Mandatory Interfaces
Version 0.25

IVOA Working Draft 2006 Sep 18

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

Previous versions:

0.24	**http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices/VOSupportInterfaces-0.24.pdf**
0.23	**http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices/VOSupportInterfaces-0.23.pdf**
0.22	**http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices/VOSupportInterfaces-0.22.pdf**
0.21	**http://www.ivoa.net/internal/IVOA/IvoaGridAndWebServices/VOSupportInterfaces-0.21.pdf**
0.2	**http://www.ivoa.net/internal/IVOA/VOSupportInterfaces-0.2.pdf**
0.1	**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

Web Services form an increasing part of the Virtual Observatory. It is felt there are some basic Interfaces that these services should provide. The word “interface” is used here in the SOAP sense - this has been referred to as a “port” in older SOAP documentation and perhaps still in the Grid community.

Hence this document attempts to define the standard interfaces that a service should provide.The VO also has many non-SOAP services that would benefit from standard interfaces. A method is also described for invoking these services in a CGI manner.

A new document is also expected which will give a VO-Profile for our services. This will tie in with WS-I specifications.

1.1 About the document

In the normal requirements manner the words "should" and "shall" are used to convey the level of necessity of the interface. Each interface is clearly given a short description and a requirement number of the form SI-N where N is a running number in the document..

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.

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

SI-2 Where the support interfaces are implemented as web resources instead of SOAP operations, they shall have distinct URLs in the HTTP scheme and shall be accessible by the 'get' operation in the HTP protocol.

SI-3 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.

2.1 Registration MetaData

The service should be the single authoritative source for metadata. WEB services provide WSDL in a standard manner. WSDL alone is insufficient for the purposes of the VO. We need something more like the VOResource implementation of the RM (Resource Metadata) document[RM]. In a web service we would return the object type produced from the XSD for the RSM. The definition of VOResource must be included in the WSDL for the service. This does not preclude returning an extension of the VOResource.

The registry remains the place to go to find metadata but a standard service like this would allow us to build a registry where we just input our service URL and the registry then asks for and validates the metadata. It could even periodically check whether it is up to date.

SI-4 All VO services should implement the "getRegistration" interface. This shall return a valid VOResource document describing the metadata of this service. The owner will have to already get an authorityID and resourceKey from a provider for this.

SI-5 All VO services should implement the "registrationChangedOn" interface. This shall return the date the metadata last changed.

2.2 Availability

The heartbeat interface is to tell us if the service is in operation. It should do a good check on the underlying service to see if it is still operational and not just be a simple

return from a web server, e.g., if it relies on a database it should check that the database is still up.
Ultimately some portals may track these heartbeats and compile uptime statistics. With the location we could have 3D global maps of services and availability.

SI-6 All VO services shall implement the “getAvailability” interface. This shall return an XML document containing:

- 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.

SI-7 The returned document shall contain an availability element as defined by the XML schema given in appendix 2.

3 Changes from previous versions

- Deleted the logging interfaces. These will be reborn in a separate standard.
- Noted that the support interfaces may be implemented in three different ways, possibly on separate endpoints from their parent service.
- The requirements have been renumbered; new ones have been added.
- Added the XSD for Availability.
- Added the WSDL

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/
v0.25"
 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/v0.25"
 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/
v0.25">
      <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"
                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: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>
```