International
Virtual
Observatory
Alliance

Registry Interfaces

Version 1.0

IVOA Note 2015-03-25

Working group
Registry WG

This version
http://www.ivoa.net/documents/WebAssets/20150325

Latest version
http://www.ivoa.net/documents/WebAssets

Previous versions
IVOA Registry Interfaces 1.0, IVOA Recommendation 2009-11-04

Author(s)
Theresa Dower, Markus Demleitner, Kevin Benson, Ray Plante, Elizabeth Auden, Matthew Graham, Gretchen Greene, Martin Hill, Tony Linde, Dave Morris, Wil O'Mullane, Guy Rixon, Aurélien Stébé, Kona Andrews

Editor(s)
Theresa Dower, Markus Demleitner

Version Control
Revision 3049, 2015-08-29 10:50:31 -0400 (Sat, 29 Aug 2015)

https://volute.g-vo.org/svn/trunk/projects/registry/RegistryInterface/RegistryInterface.tex

Abstract

The VO Registry provides a mechanism with which VO applications can discover and select resources that are relevant for a particular scientific problem. This specification defines the operation of this system. It is based on a general, distributed model composed of searchable and publishing registries, as introduced at the beginning of this document. The main body of the specification has two components: (a) an interface for harvesting publishing registries, which builds upon the Open Archives Initiative Protocol for Metadata Harvesting. (b) A VOResource extension for registering registry services. Finally, this specification briefly discusses client interfaces to the Registry as provided by searchable registries.

Status of This Document

This is an IVOA Note expressing suggestions from and opinions of the authors. It is intended to share best practices, possible approaches, or other perspectives on interoperability with the Virtual Observatory. It should not be referenced or otherwise interpreted as a standard specification.

A list of current IVOA Recommendations and other technical documents can be found at http://www.ivoa.net/Documents/.

Contents

1 Introduction

In the Virtual Observatory (VO), registries provide a means for discovering useful resources, i.e., data and services. This discovery takes place by searching within structured descriptions of resources, the resource records, authored by the data providers. In order to avoid a single point of failure for the VO, the Registry is distributed. This means that each data provider can run a service injecting resource records into the Registry (a "publishing registry" as defined below), and anyone can run services that allow global discovery (a "searchable registry" as defined below).

To enable this, common mechanisms for registry communication and interaction are required. This document therefore describes the standard interfaces that enable interoperable registries. Through these interfaces, registry builders have a common way of sharing resource descriptions with users, applications, and other registries.

This specification does not cover interfaces for global discovery, which are the subject of other IVOA standards. Also, service operators are free to build interactive, end-user interfaces in any way that best serves their target community.

1.1 Registry Archtecture and Definitions

A *registry* is first a repository of structured descriptions of resources. In the VO, a *resource* is defined by the IVOA Recommendation "Resource Metadata for the Virtual Observatory" (**?**) as being

> a general term referring to a VO element that can be described in terms of who curates or maintains it and which can be given a name and a unique identifier. Just about anything can be a resource: it can be an abstract idea, such as sky coverage or an instrumental

setup, or it can be fairly concrete, like an organization or a data collection.

Organizations, data collections, and services can be considered classes of resources. The most important type of resource to applications is a service that actually does something. A registry (lower case), then, is "a service for which the response is a structured description of resources" (?).

This specification is based on the general IVOA model for registries (?), which builds on ?'s model for resources. In the registry model, the VO environment features different types of registries that serve different functions. The primary distinction is between publishing registries and searchable ones. A secondary distinction is full versus partial.

A *searchable registry* is one that allows users and client applications to search for resource records using selection criteria against the metadata contained in the records. The purpose of this type of registry is to aggregate descriptions of many resources distributed across the network. By providing a single place to locate data and services, applications are spared from having to visit many different sites to just to determine which ones are relevant to the scientific problem at hand. A searchable registry gathers its descriptions from across the network through a process called *harvesting*.

A *publishing registry* is one that simply exposes its resource descriptions to the VO environment in a way that allows those descriptions to be harvested. The contents of these registries tend to be limited to resources maintained by one or a few providers and thus are local in nature; for example, a data center will run its own publishing registry to allow other VO components to gather metadata on the data center's published services. Since the purpose is simply publishing and not to serve users and applications directly, it is not necessary to support full searching capabilities. This simplifies the requirements for a publishing registry: storage, management, and indexing of the records can be simpler, as there is no need to support a search interface facilitating complex discovery queries. While a searchable registry in practice will necessitate the use of a database system, a simple publishing registry may get by storing its records as flat files on disk.

Note that some registries can play both roles; that is, a searchable registry may also publish its own resource descriptions.

A secondary distinction is full versus local. A *full registry* is one that attempts to contain records of all resources known to the VO. Several such registries exist, run by various VO projects. A *local registry*, on the other hand, contains only a subset of known resources. While for publishing registries this subset usually is defined by what services are maintained by the registry's operator, other selection criteria are conceivable. For instance, the IVOA's Education IG is considering running a registry only containing resources manually selected for suitability for primary and secondary education.

As mentioned above, harvesting is the mechanism by which a registry can collect resource records from other registries. It is used by full registries to

aggregate resource records from publishing registries. It can also be used to synchronize two registries to ensure that they have the same contents. Harvesting, in this specification, is modeled as a pull operation between two registries. The term *harvester* refers to the registry that wishes to receive records (usually a searchable registry); it sends its request to the *harvestee* (usually a publishing registry), which responds with the records. Harvesting is a much simpler process than a fully-featured search interface, as only very few constraints need to be supported and only full records are being transmitted in responses. Consequently, different protocols are employed for the two types of registry operations.

In this text, "registry" in lower case refers to concrete services, while "Registry" (or "VO Registry") in upper case refers to the combination of the set of all resource records and the interfaces to query and manage them.

1.2 The Registry Interface within the VO Architecture

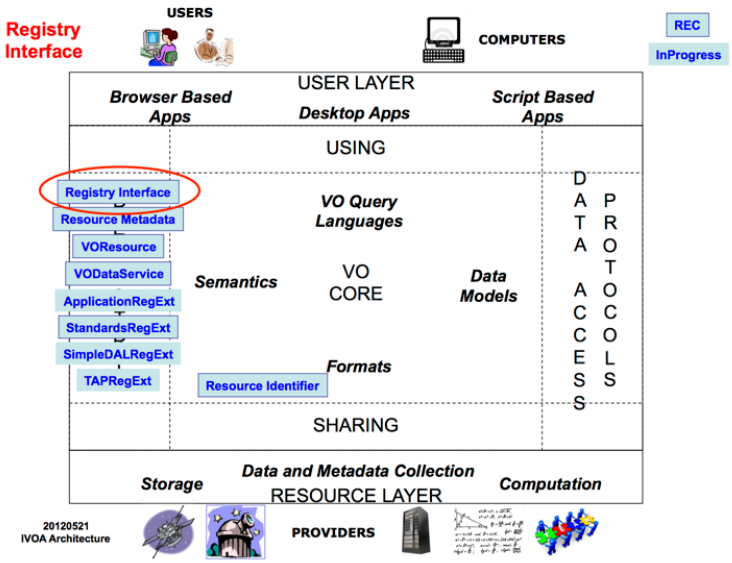

Figure 1: IVOA Architecture diagram with the Registry Interface specification (RI) and the related standards marked up.

This specification directly relates to other VO standards in the following ways:

VOResource (**?**)
: VOResource sets the foundation for a formal definition of the data model for resource records via its schema definition.

IVOA Identifiers
: IVOA identifiers are something like the primary keys to the VO registry. Also, the notion of an authority as laid down in IVOA Identifiers plays an important role as publishing registries can be viewed as a realization of a set of authorities.

2 The IVOA Harvesting Interface

The harvesting interface allows the retrieval of complete VOResource records from registries supporting harvesting. Publishing registries MUST support the IVOA harvesting interface, searchable registries SHOULD do so.

The IVOA harvesting interface is built on the standard Protocol for Metadata Harvesting developed by the Open Archives Initiative, OAI-PMH (**?**). In this section, after giving a brief introduction to OAI-PMH, we define additional constraints and requirements for OAI-PMH services to be interoperable with the VO environment.

Version 1.1 of this document drops support of the SOAP variant of OAI-PMH defined in version 1.0.

2.1 The OAI Protocol for Metadata Harvesting

While for details of OAI-PMH we refer to **?**, in the following we give a brief overview of OAI-PMH that should be sufficient to understand the protocol's role within the Registry interface architecture.

The OAI-PMH v2.0 specification defines:

- the meaning and behavior of the six harvesting operations, referred to as verbs,
- the meaning of the input arguments for each operation, and
- the XML Schema used to encode response messages.

The six standard operations laid down in OAI-PMH are:

Identify
: provides a description of the registry

ListIdentifiers
: returns a list of identifiers for the resource records held by the registry, possibly restricted to records changed within a certain time span or to those belonging to a certain set.

ListRecords
: returns complete resource records in the registry, possibly restricted to records changed within a certain time span or to those belonging to a certain set.

GetRecord
: returns a single resource description matching a given identifier.

ListMetadataFormats
: returns a list of supported formats that the registry can use to encode resource descriptions upon a harvester's request.

ListSets
: returns a list of set names supported by the registry that harvesters can request in order to get back a subset of the descriptions held by the registry.

The ListRecords and GetRecord operations return the actual resource description records held by the registry. These descriptions are encoded in XML and wrapped in a general-purpose envelope defined by the OAI-PMH XML Schema (with the namespace `http://www.openarchives.org/OAI/2.0`).

Through the operations' arguments, OAI-PMH provides a number of useful features:

- Support for multiple return formats. As suggested by the existence of the *ListMetadataFormats* operation, a harvester can request the formats available for encoding returned resource descriptions.
- Harvesting by date. The *ListIdentifiers* and *ListRecords* operations both support `from` and `until` date arguments which restrict the response to records changed withing the given, possibly half-open, interval.
- Harvesting by category. The *ListIdentifiers* and *ListRecords* operations both support a set argument for retrieving resources that are grouped in a particular category. Resource records may belong to multiple sets.
- Marking records as deleted. Registries may mark records as deleted so that harvesters will be notified that a resource has become unavailable even if only performing incremental harvests.
- Support for resumption tokens. If a request results in returning a very large number of records, the registry can choose to split the results over several calls; this is done by passing a resumption token back to the harvester. The harvester uses it to retrieve the next set of matching results.

It is important to note that the OAI-PMH interface is not intended to be a general search interface. The filtering capabilities described above are just enough to support intelligent harvesting between registries. Most end-user applications will use a dedicated search interface on a searchable registry (cf. sect. 5).

In addition to basic OAI-PMH compliance, this specification defines a set of OAI-PMH-compliant requirements and recommendations special to OAI-PMH's use within the VO that are described in the remaining subsections.

2.2 Metadata Formats for Resource Descriptions

All IVOA registries that support the Harvesting Interface must support two standard metadata formats: the OAI Dublin Core format (mandated by the base OAI-PMH standard) and the IVOA VOResource metadata format (**?**).

The VOResource metadata format has the metadata prefix name `ivo_vor`, which can be used wherever **?** allows a metadata prefix name. The format uses the VOResource core XML Schema with the namespace `http://www.ivoa.net/xml/VOResource/v1.0` (recommended namespace prefix *vr:*) along with any legal extension of this schema to encode the resource descriptions within the OAI-PMH metadata tag from the OAI XML Schema (namespace `http://www.openarchives.org/OAI/2.0`, recommended namespace prefix *oai:*).

As VOResource and its extensions do not define global elements, the child element within *oai:metadata* needs to be separately defined. This specification does this by providing the *ri:Resource* element. It is defined in a schema with the target namespace `http://www.ivoa.net/xml/RegistryInterface/v1.0`, which is given in appendix A.

The *ri:Resource* element MUST include an *xsi:type* attribute that assigns the element's type to *vr:Resource* or one of its legal extensions.

It is strongly recommended that all QName values of *xsi:type* attributes within the VOResource record use XML namespace prefixes as recommended in VOResource or the VOResource extensions. Minor version changes are not in general reflected in the recommended prefixes – e.g., both VODataService 1.0 and VODataService 1.1 use *vs:*. Registry operators who must deliver OAI-PMH decuments containing resource records written to different versions of a registry extension are advised to override the prefix bindings on the element level if at all possible.

The OAI Dublin Core format, with the metadata prefix of `oai_dc`, is defined by the OAI-PMH base standard and must be supported by all OAI-PMH compliant registries.

Ray has an XSL that makes oai_dc from ivo_vor – maybe include that in an appendix?

Harvestable registries may support other metadata formats. Responses to the *ListMetadataFormats* operation must list all names for formats supported by the registry; even though they are mandatory, this list must include `ivo_vor` and `oai_dc`.

2.3 Identifiers in OAI Messages

In accordance with the OAI-PMH standard, an OAI-PMH XML envelope that contains a resource description must include a globally unique URI that identifies that resource record. This identifier must be the IVOA identifier used to identify the resource being described as given in its *vr:identifier* child element.

This specification does not follow the recommendation of the OAI-PMH standard with regard to record identifiers. OAI-PMH makes a distinction between the resource record containing resource metadata and the resource itself;

thus, it recommends that the identifier in the OAI envelope be different from the resource identifier. In particular, the former is the choice of the publishing registry. This allows one to distinguish resource descriptions of the same resource from different registries, which in principle could be different.

In the VO, because it is intended that resource descriptions of the same resource from different registries should not differ (apart from possible additions of *vr:validationLevel* elements), there is not a strong need to distinguish between the resource and the resource description.

By making the resource and resource record identifiers the same, it becomes much easier to retrieve the record for a single resource via *GetRecord*, regardless of which registry is being queried. Otherwise – when the registry chooses the record identifier – a client will not a priori know the record identifier for a particular resource, and so it is left to call *ListRecords* and search through the metadata of all the records itself to find the one of interest. In contrast, IVOA identifiers are intended to be a cross-application way of referring to a resource, and thus when a client wants only a single specific resource record, it is very likely that it would know the resource identifier when making a call to the *GetRecord* operation.

2.4 Required Records

This section describes the records that a harvestable VO registry must include among those it emits via the OAI-PMH operations.

The harvestable registry MUST return one record that describes the registry itself as a whole, and the `ivo_vor` format MUST be supported for this record. This record is also included in the *Identify* operation response. When encoded using the `ivo_vor` format, the returned *ri:Resource* element must be of the type *vg:Registry* from the VORegistry schema (see sect. 3.4). The record MUST include a *vg:managedAuthority* for every authority identifier that originates at that registry.

Additions to the list of a registry's managed authorities must follow the protocol outlined in sect. 3.1.

The harvestable registry must be able to return exactly one record in `ivo_vor` for each authority identifier listed as a *vg:managedAuthority* in the *vg:Registry* record that describes that registry. When encoded in the `ivo_vor` format, the type of these elements must be *vg:Authority*.

2.5 The Identify Operation

The *Identify* operation describes the harvestable registry as a whole. The response from this operation must include all information required by the OAI-PMH standard. In particular, it must include an *oai:baseURL* element that must refer to the base URL to the harvesting interface endpoint. The *Identify* response must include an *oai:description* element containing a single *ri:Resource* element with an *xsi:type* attribute that sets the element's type

to *vg:Registry*. The content of *vg:Registry* type must be the registry description of the harvestable registry itself.

In its *Identify* response, an OAI-PMH-compliant registry must declare its support for deleted records. This can be one of

`no` – the registry will never notify harvesters of records that have become unvailable. In an enviroment like the VO, where searchable regiestries frequently harvest publishing registries, this is severely discouraged, as without deleted records, harvesters need to perform full harvests every time or risk delivering stale records.

`transient` – the registry will notify harvesters of records that have become unavailable, but the deleted records will entirely vanish after some time. This specification adds to the OAI-PMH requirements that registries declaring `transient` support MUST keep their deleted records for at least six months (after which they may discard them).

`persistent` – the registry promises to indefinitely keep deleted records.

2.6 IVOA Supported Sets

Sets, as defined in the OAI-PMH standard, are "an optional construct for grouping items for the purpose of selective harvesting" (see **?**, section 2.6). Harvestable IVOA registries are free to define any number of custom sets for categorizing records. The OAI-PMH standard allows a record to be a member of multiple sets.

This specification defines one reserved set name with a special meaning; future versions of this specification may define additional set names. These reserved set names will all start with the characters `ivo_`; implementors should not define their own set names that begin with this string. While support for sets is optional in the OAI-PMH standard, a VO registry MUST support the set with the reserved name `ivo_managed` to be compliant with this specification.

The `ivo_managed` set refers to all records that originate from the queried registry. That is, those records that were harvested from other registries are excluded. The Resource identifiers given in the records MUST have an authority identifier that matches on one of the *vg:managedAuthority* values in the *vg:Registry* record for that registry. Full searchable registries may use this set to avoid getting duplicate records when harvesting from many registries.

2.7 Time Granularity

Datestamps in the OAI-PMH 2.0 standard are encoded using ISO8601 and expressed in UTC, with the UTC designator "z" appended to seconds-based granularity where supplied, i.e. `YYYY-MM-DDThh:mm:ssZ`. In general OAI-PMH registries, granularity at seconds scale is optional. Harvestable IVOA registries MUST report datestamps at the granularity of seconds and accept "from" and

"until" arguments in the same format. This simplifies the incremental harvesting process.

3 Registering Registries

Both harvesting registries and Registry clients have to be able to locate the Registry resources relevant to them. Therefore, the Registry itself needs entries in the Registry.

This specification defines a VOResource extension schema called VORegistry used specifically describe a registry and its support for the registry interfaces described in this document or elsewhere. These descriptions can be stored as resource records in registries. The schema is also used to register a naming authority, which is a publisher's claim of ownership of an authority identifier from which IVOA identifiers may be created. A publishing registry is said to exclusively manage a naming authority on behalf of the owning publisher; this means that only that registry may publish records with IVOA identifiers using that authority identifier.

The XML namespace URI of this schema is `http://www.ivoa.net/xml/VORegistry/v1.0`. It has been chosen to allow it to be resolved as a URL to the XML Schema document, which is also given in appendix B. In particular, it is recommended the namespace URI be given as the location for the VORegistry schema within the *xsi:schemaLocation* attribute. The recommended prefix for this namespace is *vg:*.

I don't really like that recommendation – what's the reason for it? If nobody protests, I'll remove it – MD

The schema has not been changed from the one used in version 1.0, although the standard contents has somewhat changed. The rationale for keeping the schema is that the fact that some schema features are no longer relevant has no detrimental consequences for Registry operations, whereas breaking clients with a change of the schema and XML namespace URI might have.

3.1 The Authority Resource Extension and the Publishing Process

The *vg:Authority* type extends the core *vr:Resource* type to specifically describe the ownership of an authority identifier by a publishing organization.

The IVOA identifier of a *vg:Authority* record provided via the *vr:identifier* element must have an empty resource key component as defined in **?**.

The meaning of a *vg:Authority* record is that the organization referenced in the *vg:managingOrg* element has the sole right to create (in collaboration with a publishing registry) and register resource descriptions using the authority identifier given by the *vr:identifier* element.

Before a publisher can create resource descriptions using a new authority identifier, it must first register its claim to the authority identifier by creating a *vg:Authority* record. Before the publishing registry commits the record for export, it must first search a full registry to determine if a *vg:Authority* with

```
<ri:Resource status="active" xsi:type="vg:Authority"
   updated="2006-07-01T09:00:00" created="2006-07-01T09:00:00">
  <title>IVOA Naming Authority</title>
  <shortName>IVOA</shortName>
  <identifier>ivo://ivoa.net</identifier>
  <curation>
    <publisher ivo-id="ivo://ivoa.net/IVOA">International Virtual
      Observatory Alliance</publisher>
    <creator>
      <name>Raymond Plante</name>
      <logo>http://www.ivoa.net/icons/ivoa_logo_small.jpg</logo>
    </creator>
    <date>2006-07-01</date>
    <contact>
      <name>IVOA Resource Registry Working Group</name>
      <email>registry@ivoa.net</email>
     </contact>
  </curation>
  <content>
    <subject>virtual observatory</subject>
    <description>This registers the IVOA as the owner of the ivoa.net
      authority  identifier .</description>
    <referenceURL>http://rofr.ivoa.net/rofr/</referenceURL>
  </content>
  <managingOrg>International Virtual Observatory Alliance</managingOrg>
</ri:Resource>
```

Figure 2: A sample *vg:Authority*-typed resource record as it would be delivered within *oai:metadata*. XML namespace declarations are for the prefixes *ri:*, *xsi:*, and *vg:* are assumed on enclosing elements.

this identifier already exists; if it does, the publishing of the new *vg:Authority* record must fail.

When a registry creates a *vg:Authority* record, it is said that the registry manages the associated authority identifier (on behalf of the owning publisher) because only that registry may create records with identifiers using that authority identifier. It must also document that fact by adding a corresponding *vg:managedAuthority* element to the registry's own resource record.

The mechanism outlined here is not race-free in the distributed environment of the VO Registry. The IVOA Registry Working group periodically monitors the registry-authority graph to ensure each authority in the Registry is claimed by exactly one registry.

```
<ri:Resource status="active" xsi:type="vg:Registry"
   updated="2015-02-05T20:28:40Z" created="2006-07-01T09:00:00Z">

  <title>IVOA Registry of Registries</title>
  <shortName>RofR</shortName>
  <identifier>ivo://ivoa.net/rofr</identifier>

  <curation>(elided)</curation>

  <content>
    <subject>virtual observatory</subject>
    <description>(elided)</description>
    <referenceURL>http://rofr.ivoa.net/rofr/</referenceURL>
    <type>Registry</type>
  </content>

  <capability xsi:type="vg:Harvest"
      standardID="ivo://ivoa.net/std/Registry">
    <interface xsi:type="vg:OAIHTTP" version="1.0" role="std">
      <accessURL>http://rofr.ivoa.net/cgi-bin/oai.pl</accessURL>
    </interface>
    <maxRecords>0</maxRecords>
  </capability>

  <full>false</full>

  <managedAuthority>ivoa.net</managedAuthority>
</ri:Resource>
```

Figure 3: A sample *vg:Registry*-typed resource record as it would be delivered within *oai:metadata*, including a harvest capability. XML namespace declarations are for the prefixes *ri:*, *xsi:*, and *vg:* are assumed on enclosing elements.

3.2 Describing Registries with the Registry Resource Extension

The *vg:Registry* type extends the core *vr:Service* type to specifically describe registries in order to support discovering them and collecting their metadata; in addition, the extension type also defines the VO-specific metadata in the response to an OAI-PMH *Identify* request.

As a subclass of *vr:Service*, the *vg:Registry* type uses *vr:capability* elements to describe its support for network interfaces to the services. The specific types defined here derive from an intermediate restriction on *vr:Capability* called *vg:RegCapRestriction* to force the value of the *standardID* attribute to be `ivo://ivoa.net/std/Registry`. When the schema will be revised, it

is planned to change this *standardID* to refer to a fragment within the standard's resource record. In particular, OAI-PMH endpoints as specified here will be identified by `ivo://ivoa.net/std/Registry#OAI-2.0`. Client writers are advised to write their discovery routines accordingly.

If the *vg:full* element in an *vg:Registry* instance is set to `true`, it indicates the registry's intent to accept all valid resource records it harvests from other registries in accordance with the OAI-PMH specification. This will typically be searchable registries implementing some Registry search interface, but there are use cases for full registries just implementing OAI-PMH (and thus just providing an *vg:Harvest* capability), too.

The *vg:managedAuthority* is used by publishing registries to claim an authority identifier (see also sect. 2.4). Note that for each managed authority claimed, the registry MUST provide a *vg:Authority*-typed resource record for that authority identifier within its `ivo_managed` set.

As of version 1.1 of this specification, VO registries must provide the three mandatory VOSI capabilities (**?**).

3.3 The Search Capability

Version 1 of this standard defined a search interface, and such interfaces are described by capabilites of the type *vg:Search*. Since in this version, search interfaces are specified by external standards, such external standards may define differing ways of discovering them[1]. The search capability nevertheless is not removed from the schema in order to allow operators to register RI1 registries without having to support different versions of the VORegistry schema. Also, the type may be useful when other registry search interfaces want to define capability types of their own.

3.4 The Harvesting Capability

A registry declares itself to be a harvestable registry by including a *vr:capability* element with an *xsi:type* attribute set to *vg:Harvest*.

A *vr:capability* element of type *vg:Harvest* MUST include at least one *vr:interface* element with an *xsi:type* attribute set to *vg:OAIHTTP* and the *role* attribute set to `std`. If the *vr:capability* element is used to simultaneously describe support for other versions of this Registry Interface standard, then the *vr:interface* element describing support for this version must include the version attribute set to `1.0`. The *vr:accessURL* element must be set to the base URL for the OAI-PMH interface.

The *vg:OAISOAP* extension of *vr:WebService* was used by version 1 of this specification and is no longer part of VO Registry interfaces.

[1] For instance, RegTAP (**?**) uses the *tre:dataModel* element from TAPRegExt as its primary discovery mechanism

4 Discovering Registries

4.1 The Registry of Registries

To facilitate discovery and automated harvesting of registries containing VOResource records, a registry serving as a master list of IVOA registries exists as part of the IVOA web infrastructure, hosted at `http://rofr.ivoa.net/`. It is referred to as the Registry of Registries, or RofR. As the Registry of Registries is itself a registry, an OAI-PMH interface is provided which conforms to this document. The OAI-PMH interface is always available at `http://rofr.ivoa.net/cgi-bin/oai.pl`.

The Registry of Registries includes the VOResource records directly representing each currently active registry of IVOA resources, be they fully searchable or providing only an OAI-PMH harvesting interface. These resources are of type *vg:Registry* as defined in section 3.2. The Registry of Registries also contains the canonical VOResource descriptions of the most recent versions of VOResource standards and extensions themselves, which are of type *vstd:Standard*.

4.2 Inclusion in the Registry of Registries listing

To be considered an IVOA registry, the curator of a publishing registry containing VO resources must validate their registry's contents using the RofR web form at `http://rofr.ivoa.net/regvalidate/regvalidate.html`. The automated validation process includes testing the registry's standard interfaces against this document and the OAI-PMH 2.0 standard, schema-validating each included resource against the VOResource standard, and checking the referential integrity of its Authority and Registry records.

Upon successful validation, the publishing registry will be automatically included in the RofR listing, both in the website and the RofR's standard interfaces. After initial inclusion in the RofR listing, updates to the registry which are not re-run against the validation service are not necessarily automatically validated, and updates to the Registry record itself are not automatically reflected in the RofR contents.

4.3 Harvesting the Registry of Registries

Given the Registry of Registries contains records for all other currently active and validated IVOA registries, a client wishing to harvest the contents of all registries should begin at the RofR. Fully searchable registries wishing to include records from the other IVOA registries count among these potential clients. To harvest the entire contents of IVOA registries, it is recommended to first harvest the Registry of Registries via its OAI-PMH interface.

The next step in harvesting the entire distributed IVOA registry contents is to iterate over the *accessURL* of each *vg:Registry* record's *vr:capability* of type *vg:Harvest*, and use the url for each of those OAI-PMH interfaces to harvest the individual registries. This filtering of RofR contents can be

done by adding the `set` parameter to an OAI query to the RofR: registries in the RofR comprise the supported set `ivo_publishers`. The following query to the RofR OAI-PMH interface will return only the records for the publishing registries in the RofR: `http://rofr.ivoa.net/cgi-bin/oai.pl?verb=ListRecords&metadataPrefix=ivo_vor&set=ivo_publishers`

For the next step of harvesting each registry in turn, to avoid harvesting duplicate records from the fully searchable registries, it is recommended to add the `set` parameter to the OAI query for each individual publishing registry: records specifically published by a registry which also has a search interface comprise that registry's supported set `ivo_managed`.

5 Searching the Registry

Experience with version 1 of this specification suggests that it is preferable to not couple the relatively stable parts on harvesting and general registry maintenance with client interfaces to the registry, which were found to be in much more need of experimentation. For a discussion of the history of client interfaces in the VO, see **?**.

One second-generation standard search interface to the VO Registry that has progressed to become an IVOA recommendation is RegTAP (**?**), an interface based on a relational representation of major parts of VOResource and the VO's TAP protocol (**?**). Services are available from several data providers.

A The RegistryInterface Schema

The following schema defines a global element, allowing the inclusion of VOResource records into *oai:metadata* elements in OAI-PMH responses for the `ivo_vor` metadata prefix. See sect. 2.2 for details.

The schema is unchanged from version 1.0 of this specification and therefore does not change its version.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.ivoa.net/xml/RegistryInterface/v1.0"
          xmlns:ri="http://www.ivoa.net/xml/RegistryInterface/v1.0"
          xmlns="http://www.w3.org/2001/XMLSchema"
          xmlns:xs="http://www.w3.org/2001/XMLSchema"
          xmlns:vr="http://www.ivoa.net/xml/VOResource/v1.0"
          elementFormDefault="qualified"
          version="1.0">

   <xs:import namespace="http://www.ivoa.net/xml/VOResource/v1.0"
              schemaLocation="http://www.ivoa.net/xml/VOResource/v1.0"/>

   <xs:element name="VOResources">
      <xs:annotation>
         <xs:documentation>
```

```
          a container for one or more resource descriptions or
          identifier references to resources.
        </xs:documentation>
        <xs:documentation>
          This is used to transmit multiple resource descriptions
          resulting from a query.
        </xs:documentation>
      </xs:annotation>

      <xs:complexType>
        <xs:sequence>
          <xs:choice>
            <xs:element ref="ri:Resource"
                        minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="identifier" type="vr:IdentifierURI"
                        minOccurs="0" maxOccurs="unbounded"/>
           </xs:choice>
        </xs:sequence>
        <xs:attribute name="from" type="xs:positiveInteger" use="required" />
        <xs:attribute name="numberReturned" type="xs:positiveInteger"
                      use="required" />
        <xs:attribute name="more" type="xs:boolean" use="required" />
      </xs:complexType>
    </xs:element>

    <xs:element name="Resource" type="vr:Resource">
      <xs:annotation>
        <xs:documentation>
          a description of a single resource
        </xs:documentation>
      </xs:annotation>
    </xs:element>

</xs:schema>
```

B The VORegistry Schema

The following schema defines VOResource types for describing registries in the Registry. It is unchanged from version 1.0 of this specification and therefore does not change its version.

Note that standards defining search interfaces may specify alternative or complementary methods of registering the services defined by them.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.ivoa.net/xml/VORegistry/v1.0"
          xmlns:xs="http://www.w3.org/2001/XMLSchema"
          xmlns:vr="http://www.ivoa.net/xml/VOResource/v1.0"
          xmlns:vg="http://www.ivoa.net/xml/VORegistry/v1.0"
```

```
       xmlns:vm="http://www.ivoa.net/xml/VOMetadata/v0.1"
       elementFormDefault="unqualified" attributeFormDefault="unqualified"
       version="1.0wd">

<xs:annotation>
   <xs:appinfo>
     <vm:schemaName>VORegistry</vm:schemaName>
     <vm:schemaPrefix>xs</vm:schemaPrefix>
     <vm:targetPrefix>vg</vm:targetPrefix>
   </xs:appinfo>
   <xs:documentation>
     An extension to the core resource metadata (VOResource) for
     describing registries and authority IDs.
   </xs:documentation>
</xs:annotation>

<xs:import namespace="http://www.ivoa.net/xml/VOResource/v1.0"
          schemaLocation="http://www.ivoa.net/xml/VOResource/v1.0"/>

<xs:complexType name="Registry">
   <xs:annotation>
     <xs:documentation>
       a service that provides access to descriptions of resources.
     </xs:documentation>
     <xs:documentation>
       A registry is considered a publishing registry if it
       contains a capability element with xsi:type="vg:Harvest".
       It is considered a searchable registry if it contains a
       capability element with xsi:type="vg:Search".
     </xs:documentation>
   </xs:annotation>

   <xs:complexContent>
     <xs:extension base="vr:Service">
       <xs:sequence>
         <xs:element name="full" type="xs:boolean">
           <xs:annotation>
             <xs:documentation>
               If true, this registry attempts to collect all resource
               records known to the IVOA.
             </xs:documentation>
             <xs:documentation>
               A registry typically collects everything by harvesting
               from all registries listed in the IVOA Registry of
               Registries.
             </xs:documentation>
           </xs:annotation>
         </xs:element>

         <xs:element name="managedAuthority" type="vr:AuthorityID"
```

```
                    minOccurs="0" maxOccurs="unbounded">
               <xs:annotation>
                  <xs:documentation>
                    an authority identifier managed by the registry.
                  </xs:documentation>
                  <xs:documentation>
                    Typically, this means the AuthorityIDs that originated
                    (i.e. were first published by) this registry. Currently,
                    only one registry can lay claim to an AuthorityID via
                    this element at a time.
                  </xs:documentation>
               </xs:annotation>
            </xs:element>
         </xs:sequence>
      </xs:extension>
   </xs:complexContent>
</xs:complexType>

<xs:complexType name="RegCapRestriction" abstract="true">
   <xs:annotation>
      <xs:documentation>
         an abstract capability that fixes the standardID to the
         IVOA ID for the Registry standard.
      </xs:documentation>
      <xs:documentation>
         See vr:Capability for documentation on inherited children.
      </xs:documentation>
   </xs:annotation>
   <xs:complexContent>
      <xs:restriction base="vr:Capability">
         <xs:sequence>
            <xs:element name="validationLevel" type="vr:Validation"
                        minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="description" type="xs:token"
                        minOccurs="0"/>
            <xs:element name="interface" type="vr:Interface"
                        minOccurs="0" maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="standardID" type="vr:IdentifierURI"
                      use="required" fixed="ivo://ivoa.net/std/Registry"/>
      </xs:restriction>
   </xs:complexContent>
</xs:complexType>

<xs:complexType name="Harvest">
   <xs:annotation>
      <xs:documentation>
         The capabilities of the Registry Harvest implementation.
      </xs:documentation>
   </xs:annotation>
```

```
   <xs:complexContent>
      <xs:extension base="vg:RegCapRestriction">
         <xs:sequence>

            <xs:element name="maxRecords" type="xs:int">
               <xs:annotation>
                  <xs:documentation>
                     The largest number of records that the registry search
                     method will return. A value greater than one implies
                     that an OAI continuation token will be provided when
                     the limit is reached. A value of zero or less
                     indicates that there is no explicit limit and
                     thus, continuation tokens are not supported.
                  </xs:documentation>
               </xs:annotation>
            </xs:element>

         </xs:sequence>
      </xs:extension>
   </xs:complexContent>
</xs:complexType>

<xs:complexType name="Search">
   <xs:annotation>
      <xs:documentation>
         The capabilities of the Registry Search implementation.
      </xs:documentation>
   </xs:annotation>

   <xs:complexContent>
      <xs:extension base="vg:RegCapRestriction">
         <xs:sequence>

            <xs:element name="maxRecords" type="xs:int">
               <xs:annotation>
                  <xs:documentation>
                     The largest number of records that the registry search
                     method will return. A value of zero or less indicates
                     that there is no explicit limit.
                  </xs:documentation>
               </xs:annotation>
            </xs:element>

            <xs:element name="extensionSearchSupport"
                        type="vg:ExtensionSearchSupport">
               <xs:annotation>
                  <xs:documentation>
                    the level of support provided for searching against
                    metadata defined in a legal VOResource extension schema.
```

```
            </xs:documentation>
            <xs:documentation>
              A legal VOResource extension schema is one that imports
              and extends the VOResource core schema in compliance
              with the VOResource standard.
            </xs:documentation>
          </xs:annotation>
        </xs:element>

        <xs:element name="optionalProtocol" type="vg:OptionalProtocol"
                  minOccurs="0" maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>
              the name of an optional advanced search protocol
              supported.
            </xs:documentation>
            <xs:documentation>
              Only one optional protocol is currently allowed
              (XQuery). It is assumed that the required protocols
              (simple keyword search and ADQL) are supported.
            </xs:documentation>
          </xs:annotation>
        </xs:element>

      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:simpleType name="ExtensionSearchSupport">
  <xs:restriction base="xs:NMTOKEN">
    <xs:enumeration value="core">
      <xs:annotation>
        <xs:documentation>
          Only searches against the core VOResource metadata are
          supported.
        </xs:documentation>
      </xs:annotation>
    </xs:enumeration>
    <xs:enumeration value="partial">
      <xs:annotation>
        <xs:documentation>
          Searches against some VOResource extension metadata are
          supported but not necessarily all that exist in the registry.
        </xs:documentation>
      </xs:annotation>
    </xs:enumeration>
    <xs:enumeration value="full">
      <xs:annotation>
        <xs:documentation>
```

```
            Searches against all VOResource extension metadata contained
            in the registry are supported.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="OptionalProtocol">
   <xs:restriction base="xs:NMTOKEN">
      <xs:enumeration value="XQuery">
         <xs:annotation>
           <xs:documentation>
              the XQuery (http://www.w3.org/TR/xquery/) protocol as defined
              in the VO Registry Interface standard.
           </xs:documentation>
         </xs:annotation>
      </xs:enumeration>
   </xs:restriction>
</xs:simpleType>

<xs:complexType name="OAIHTTP">
   <xs:annotation>
      <xs:documentation>
        a description of the standard OAI PMH interface using HTTP
        (GET or POST) queries.
      </xs:documentation>
      <xs:documentation>
        the accessURL child element is the base URL for the OAI
        service as defined in section 3.1.1 of the OAI PMH
        standard.
      </xs:documentation>
   </xs:annotation>

   <xs:complexContent>
      <xs:extension base="vr:Interface">
         <xs:sequence/>
      </xs:extension>
   </xs:complexContent>

</xs:complexType>

<xs:complexType name="OAISOAP">
   <xs:annotation>
      <xs:documentation>
        a description of the standard OAI PMH interface using a SOAP
        Web Service interface.
      </xs:documentation>
      <xs:documentation>
        the accessURL child element is the service port location URL for
```

```
        the OAI SOAP Web Service.
      </xs:documentation>
    </xs:annotation>

    <xs:complexContent>
      <xs:extension base="vr:WebService">
        <xs:sequence/>
      </xs:extension>
    </xs:complexContent>

  </xs:complexType>

  <xs:complexType name="Authority">
    <xs:annotation>
      <xs:documentation>
        a naming authority; an assertion of control over a
        namespace represented by an authority identifier.
      </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="vr:Resource">
        <xs:sequence>

          <xs:element name="managingOrg" type="vr:ResourceName">
            <xs:annotation>
              <xs:documentation>
                the organization that manages or owns this authority.
              </xs:documentation>
              <xs:documentation>
                In most cases, this will be the same as the Publisher.
              </xs:documentation>
            </xs:annotation>
          </xs:element>

        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

</xs:schema>
```

C Changes from Previous Versions

For pre-REC-1.0 changes, see ?.

D Changes from Version 1.0

- Added requirement for OAI-PMH interface of IVOA Registries to support seconds granularity, optional in the OAI-PMH 2.0 standard itself.
- Removed requirement for version number changes to the VOResource standard to force an update of this document.
- Removed the entire section 2, specifically the SOAP-based services based on "ADQL 1.0" and XQuery.
- Dropped the requirement on registries to not deliver any records that are OAI-PMH deleted when no temporal constraint is given.
- Announcing a migration to `ivo://ivoa.net/std/Registry#OAI-2.0` as the harvesting capability's standard id on a schema change.
- Added a requirement to provide VOSI endpoints.
- Clarified that the requirement to keep deleted records for six months only applies to the transient case; also discouraging registries with no support of deleted records.
- Added recommended process for discovery of registries and their resources using the Registry of Registries.
- Many editorial changes across the text, mostly as a consequence of externalizing the search interface.