VOResource: an XML Encoding Schema for Resource Metadata
Version 1.1

IVOA Recommendation 2018-06-25

Working group
Registry
This version
http://www.ivoa.net/documents/VOResource/20180625
Latest version
http://www.ivoa.net/documents/VOResource
Previous versions
REC -1.03
WD 2006-11-07
WD 2006-06-20
WD-2006-05-30
Author(s)
Raymond Plante, Kevin Benson, Markus Demleitner, Matthew Graham, Gretchen Greene, Paul Harrison, Gerard Lemson, Tony Linde, Guy Rixon
Editor(s)
Ray Plante, Markus Demleitner

Abstract

This document describes an XML encoding standard for IVOA Resource Metadata, referred to as VOResource. This schema is primarily intended to support metadata exchange between in interoperable registries and resource discovery within them. However, any application that needs to describe resources may use this schema. In this document, we define the types and elements that make up the schema in close alignment to the metadata terms defined in Resource Metadata for the Virtual Observatory (Hanisch and IVOA Resource Registry Working Group et al., 2007), but also taking into account other metadata standards as well as experiences from the operation of the VO Registry. We also describe the general model for the schema and explain how it may be extended to add new metadata terms and describe more specific types of resources.

Status of this document

Contents

Acknowledgments

This document has been developed with support from the National Science Foundation's Information Technology Research Program under Cooperative Agreement AST0122449 with The Johns Hopkins University, from the UK Particle Physics and Astronomy Research Council (PPARC), and from the European Commission's Sixth Framework Program via the Optical Infrared Coordination Network (OPTICON). Funding for the 2016 update was provided by BMBF grant GAVO-2014, grant number 05A14VHA.

The vocabulary management closely follows the one used in the managament of the Datalink (**?**) core vocabulary, designed by Norman Gray.

Conformance-related definitions

The words "MUST", "SHALL", "SHOULD", "MAY", "RECOMMENDED", and "OPTIONAL" (in upper or lower case) used in this document are to be interpreted as described in IETF standard RFC2119 (Bradner, 1997).

The *Virtual Observatory (VO)* is a general term for a collection of federated resources that can be used to conduct astronomical research, education, and outreach. The International Virtual Observatory Alliance (IVOA) is a global collaboration of separately funded projects to develop standards and infrastructure that enable VO applications.

Syntax Notation Using XML Schema

The eXtensible Markup Language, or XML, is document syntax for marking textual information with named tags and is defined by Bray and Paoli et al. (2008). The set of XML tag names and the syntax rules for their use is referred to as the document schema. One way to formally define a schema for XML documents is using the W3C standard known as XML Schema (Thompson and Beech et al., 2004).

The XML Schema of VOResource is available from the IVOA document repository[1] at any time. Parts of the schema appear within the main sections of this document; however, documentation nodes have been left out for the sake of brevity.

[1] `http://www.ivoa.net/xml`

Where the content of the pieces of schema embedded in this text diverges from the schema document in the IVOA document repository, the version in the schema repository is authoritative.

References to specific elements and types defined in the VOResource schema include the namespaces prefix `vr` as in `vr:Resource` (a type defined in the VOResource schema). For more details on the intended interpretation, refer to sect. 2.1.

1 Introduction

The IVOA recommendation "Resource Metadata for the Virtual Observatory" (Hanisch and IVOA Resource Registry Working Group et al., 2007), hereafter referred to as the RM, defines metadata terms for describing resources. The RM defines a resource as:

> . . . 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 organisation or a data collection. This definition is consistent with its use in the general Web community as "anything that has an identity" (Berners-Lee and Fielding et al., 2005). We expand on this definition by saying that it is also describable.

The resource metadata are, then, the terms and concepts that describe a resource in general. The RM defines the terms as well as describes reasonable or allowed values; it does not, however, describe how the terms and values should be encoded. This is because resource metadata may be encoded in several different formats, depending on the context. The present document provides a concrete encoding of the resulting metadata model on XML suitable for embedding into OAI-PMH responses.

In addition to concepts and structures laid down in the RM, VOResource also acknowledges other (though mostly related) metadata schemes. Of particular importance here is the metadata scheme employed by DataCite, which at the time of writing is at version 4.0 (DataCite Metadata Working Group, 2016). This, in particular, concerns the alignment of the vocabularies for date roles and relationships.

The primary intended use of VOResource is to provide an XML interchange format for use with resource registries. A registry is a repository of resource descriptions and is employed by users and applications to discover resources. VOResource can be used to pass descriptions from publishers to registries and then from registries to applications. Another intended use is as a language for services to describe themselves directly. VOResource may be used in other ways, in whole or in part, using the standard XML mechanisms (e.g., import, include).

The VOResource schema provides XML encoding for core metadata from the RM that (with a few exceptions) can apply to all resources; however, it is recognized that a full and useful description of a *specific* resource will require additional metadata that is relevant only to a resource of its type. Thus, the VOResource schema has been especially designed to be extended. The model for doing this is described in sect. 2.3.

Note

The term "VOResource" is used to denote two different concepts within the IVOA community. The first refers specifically to the core XML schema defined by this document. The second refers more broadly to the core schema plus the set of legal extensions. In this document, use of the name "VOResource" corresponds to the first meaning. Reference to the broader set of schemas will be indicated explicitly with the annotating phrase, "and its legal extensions."

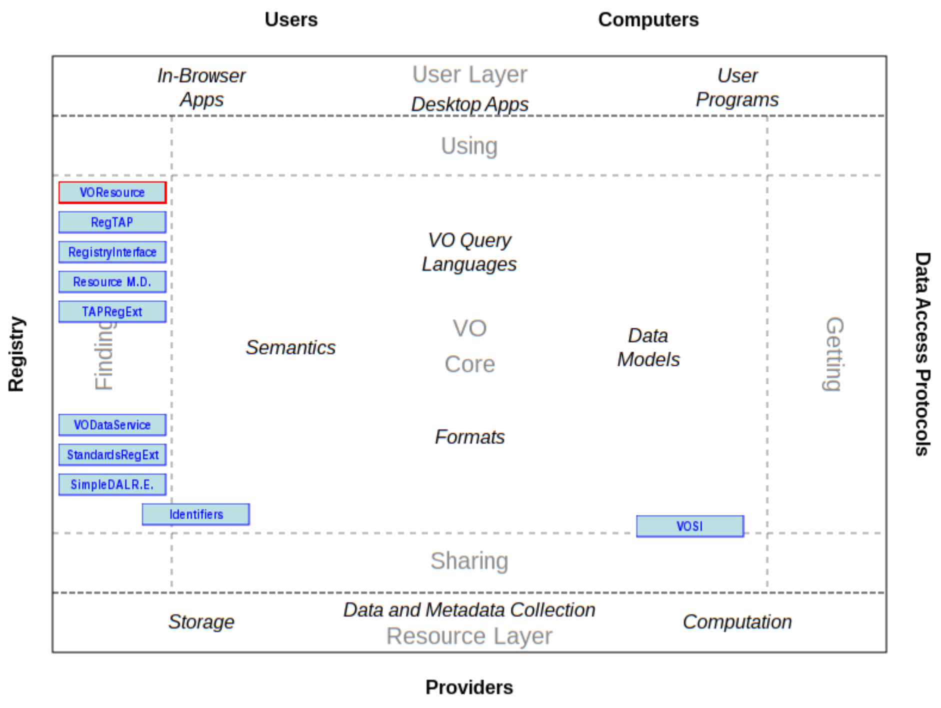

Figure 1: Architecture diagram for this document

1.1 Role within the VO Architecture

Fig. 1 shows the role VOResource plays within the IVOA architecture (Arviset and Gaudet et al., 2010).

VOResource depends on the following other IVOA standards:

Resource Metadata, v1.12 (Hanisch and IVOA Resource Registry Working Group et al., 2007)
The RM provides the central concepts and structures mapped here into an XML schema.

There are relationships to the following other IVOA standards:

Registry Interfaces (Benson and Plante et al., 2009) Registry Interfaces describes how registries exchange VOResource records, and it provides a `vr:Resource`-typed element to hold them within XML data (VOResource itself has no global elements).

VOResource extensions VOResource lays the foundations upon which description schemes for concrete resources are built in various VOResource extensions. At the time of writing, recommendations have been passed for the description of VO Standards (Harrison and Burke et al., 2012), various "simple" data discovery protocols (Plante and Delago et al., 2013), generic data services (Plante and Stébé et al., 2010), and TAP services (Demleitner and Dowler et al., 2012). The Registry Interfaces standard also contains a VOResource extension.

RegTAP (Demleitner and Harrison et al., 2014) The Registry Relational Schema gives a relational mapping of the models discussed here.

VOSI (Graham and Rixon et al., 2011) The VO support interfaces re-use the service model developed here to facilitate service self-description.

2 The VOResource Data Model

The primary use for VOResource, of course, is to describe a resource using the metadata concepts defined in the RM. Here is an example of a VOResource document describing an organisation, the Radio Astronomy Imaging Group at the National Center for Supercomputing Applications.

```
<?xml version="1.0" encoding="UTF-8"?>
<ri:Resource xsi:type="vr:Organisation"
        xmlns:vr="http://www.ivoa.net/xml/VOResource/v1.0"
        xmlns:ri="http://www.ivoa.net/xml/RegistryInterface/v1.0"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.ivoa.net/xml/VOResource/v1.0
                        http://www.ivoa.net/xml/VOResource/v1.0
                        http://www.ivoa.net/xml/RegistryInterface/v1.0
                        http://www.ivoa.net/xml/RegistryInterface/v1.0"
        created="2009-02-15T12:00:00"
        updated="2009-02-15T12:00:00"
        status="active">
   <validationLevel validatedBy="ivo://archive.stsci.edu/nvoregistry">
     2
   </validationLevel>

   <title>NCSA Radio Astronomy Imaging</title>
   <shortName>NCSA-RAI</shortName>
   <identifier>ivo://rai.ncsa/RAI</identifier>

   <curation>
      <publisher ivo-id="ivo://ncsa.uiuc/NCSA">
         National Center for Supercomputing Applications
      </publisher>
      <creator>
         <name> Crutcher, Richard </name>
         <logo>
            http://rai.ncsa.uiuc.edu/rai.jpg
         </logo>
      </creator>
```

```
31          <date>1993-01-01</date>
32          <contact>
33              <name>Plante, R.</name>
34              <email>rplante@ncsa.uiuc.edu</email>
35          </contact>
36      </curation>
37
38      <content>
39          <subject>radio astronomy</subject>
40          <subject>data repositories</subject>
41          <subject>digital libraries </subject>
42          <subject>grid-based processing</subject>
43          <description>
44              The Radio Astronomy Imaging Group at the National Center for
45              Supercomputing Applications is focused on applying
46              high-performance computing to astronomical research. Our
47              projects include the NCSA Astronomy Digital Image Library,
48              the BIMA Data Archive, the BIMA Image Pipeline, and the
49              National Virtual Observatory.
50          </description>
51          <referenceURL>http://rai.ncsa.uiuc.edu/</referenceURL>
52          <type>Organisation</type>
53          <contentLevel>Research</contentLevel>
54      </content>
55
56      <facility>Berkeley-Illinois-Maryland Array (BIMA)</facility>
57      <facility>
58          Combined Array for Research in Millimeter Astronomy (CARMA)
59      </facility>
60
61  </ri:Resource>
```

This example illustrates some important components of a VOResource record:

1. the VOResource namespace (line 3),
2. the specific type of resource indicated by the value of the ***xsi:type*** attribute as discussed in sect. 2.2.6 (line 2),
3. the location of the schema documents used by this description (lines 6–9) – this is to facilitate document validation –,
4. values for the three main types of core metadata: identity (lines 17–19), curation (lines 21–36), and content (lines 38–54),
5. a reference to another resource is made by providing that resource's IVOA identifier (line 22),
6. string values can be padded with spaces for easier readability (e.g., lines 22–24),
7. extension metadata specific to the type of resource (lines 56–59).

Actual services obviously need additional metadata defining the modalities of service access. In VOResource, such information is typcially encapsulated in ***vr:Capability***-typed elements. This document only provides a generic base class for them, deferring the definition of more useful subclasses to VOResource extensions like VODataService (Plante and Stébé et al., 2010) or SimpleDALRegExt (Plante and Delago et al., 2013).

2.1 The Schema Namespace and Location

The VOResource schema namespace is

`http://www.ivoa.net/xml/VOResource/v1.0`.

The namespace URI has been chosen to allow it to be resolved as a URL to the XML Schema document that defines the VOResource schema. Applications may assume that the namespace URI is so resolvable.

Although this schema is in version 1.1 now, the URL still ends in `v1.0`. This is to avoid unnecessarily breaking existing clients relying on the namespace as defined by version 1.0 of this specification. As laid out in the IVOA schema versioning policies (**?**), although minor versions should never have been part of namespace URIs, for namespaces defined before this note they cannot be dropped despite their potential for confusion.

Clients should in general not care about minor versions of schemas. To cover the rare cases in which such information is necessary, instance documents for version 1.1 of VOResource must have a *version* attribute with value `1.1` on their root element (i.e., *Resource*).

Document authors are strongly encouraged to bind this namespace to the *vr:* prefix. While in generic XML processing, the concrete prefix used is irrelevant as long as the namespace URI mapped is the one given by this specification, in the Virtual Observatory context uniform, per-major-version prefixes are viewed as helping interoperability. This is particularly true when prefixes appear in attribute values (e.g., of *xsi:type*), as non-schema aware XML processors cannot URI-normalize such occurrences. Also, non-XML-aware software (e.g., ADQL engines) will use these prefixes rather than the full namespace URIs.

Authors of instance documents that use the VOResource schema may choose to provide a location for VOResource XML Schema document using the *xsi:schemaLocation* attribute; the choice of the location value is the choice of the author. In general, the use of *xsi:schemaLocation* is recommended by this specification with a namespace URI given as the location as illustrated in the example above, unless the application prefers otherwise.

```
xsi:schemaLocation="http://www.ivoa.net/xml/VOResource/v1.0
                    http://www.ivoa.net/xml/VOResource/v1.0"
```

The VOResource XML schema sets *elementFormDefault* to unqualified. This means that instance documents must not bind the empty namespace (using `xmlns="..."`) in any element containing VOResource elements, as XML element names in VOResource must be interpreted outside of any namespace. Conversely, the VOResource namespace prefix must not be used to qualify tag names of VOResource elements.

In general, namespace prefixes in VOResource are only used to qualify type names given as values to the *xsi:type* attribute in VOResource (see next section). VOResource extensions should also set `elementFormDefault="unqualified"` for consistency.

Note

In the example instance document at the beginning of section 2, the root element *resource* is not defined in any schema. Nevertheless, this document is still legal and verifiable XML. This is because the element's type is explicitly specified with the *xsi:type* attribute.

2.2 The Core Structural and Semantic Model

The VOResource schema only defines global types; it does not define any global elements (often referred to as root elements). It is the responsibility of the application to define the root element of the VOResource-employing documents it operates on. Typically, the root element is defined in a separate application-specific schema. The type of an application document's root element is not assumed to be any particular type defined in the VOResource schema (nor any of its legal extensions). In fact, it need not be of a type from the VOResource at all; rather, VOResource types may appear anywhere in the document.

The IVOA Registry Interface standard (Benson and Plante et al., 2009), for example, at the time of writing includes a small schema that defines a global element *ri:Resource* of base type *vr:Resource*. It is primarily intended as the child of *oai:metadata* in OAI-PMH responses (Lagoze and de Sompel et al., 2002), the protocol used for VOResource record exchange by Registry Interface-compliant Registries. Note that even *ri:Resource*-typed elements will still use *xsi:type* to indicate the actual resource type.

Applications describe a single resource using an element of the type *vr:Resource* or a legal derivation of it. The content of this element is referred to as the **core VOResource metadata** and can be grouped into four categories (corresponding to the sections 3.1, 3.2, 3.3, and 4 of the RM):

- identity metadata: the *title*, *shortName*, and *identifier* elements;
- curation metadata: the contents of the *curation* element;
- general content metadata: the contents of the *content* element;
- metadata quality flags: the *validationLevel* element.

These elements are defined in more detail in sect. 3.

2.2.1 Naming Convention

VOResource uses the following conventions for names of elements and types:

- all global types it defines have names that are capitalized. (This practice would extend to global elements, if they existed in the VOResource.)
- Locally defined elements begin with a lower-case character.

- For all types and element names that are made up of multiple words, such as *shortName*, upper-case letters are used demarcate the start of appended word (the “camel” format).

- Names that include abbreviations, such as *IdentifierURI*, all letters in the abbreviation are capitalized.

It is recommended that this convention be followed in other schemas that either use the VOResource schema (via an *xsd:import* or *xsd:include*) or extend it.

2.2.2 String Values

Many of the elements in VOResource that are meant to have string or URI values are defined as being of the type *xs:token*. This allows authors of VOResource instance documents to pad string and URI values with spaces and include carriage returns to improve readability. The definition of these types will cause an XML Schema-compliant parser to replace tab, line feed, and carriage return characters with simple spaces, then replace multiple sequential occurrences of spaces with a single space, and then remove all leading and trailing spaces.

The *description* children of resources and capabilities, on the other hand, are of type *xs:string*, which means that even schema-aware processors will preserve whitespace. This is inteded to allow simple markup like paragraphs (empty lines) in the descriptions, which may be little texts. Since in VO practice, constructs like simple ASCII tables sometimes appear in such descriptions, clients are encouraged to not reflow descriptions and display them in a fixed-with font.

2.2.3 Language and Transliteration

Several VOResource elements contain natural language (e.g., *description*, *title*, *subject*). In order to ensure reliable discovery, in core VOResource, these elements must contain English text, with US spelling strongly preferred; technically, an *xml:lang* value of `en-US` is implied.

Registry extensions may allow *xml:lang* attributes on elements. If they do, such elements must be repeatable, and an element without *xml:lang* (and hence, `en-US` implied) should be required for global discoverability. The requirements on *xml:lang* from the XML specification (Bray and Paoli et al., 2008) apply. Additionally, in VOResource documents BCP 47 language identifiers must be written in lowercase.

Several VOResource elements contain names. Again, for reliable global discoverability, such names must be given in (common) English transliteration where their original form uses non-Latin scripts. Latin letters with diacritics are allowed, but Registry components are generally expected to treat them equivalent to their base letters.

2.2.4 Dates and Times

All VOResource elements and attributes that contain dates must be interpreted as UTC dates and must be encoded in compliance with ISO8601 (International Organization for Standardization), 2004) standard Date and Time Format. The *vr:UTCTimestamp* type provides a special restriction of the format that requires inclusion of date and time, but enforces the use of the UTC timezone. The time format to be used by VOResource (designed to coincide with what the OAI-PMH transport protocol mentioned above defines) therefore requires timestamps like

`2008-02-22T12:22:34Z,`

where optional fractional seconds are allowed. See the schema documentation for the actual regular expression pattern.

For historical reasons, VOResource also allows a form without any timezone marker. Such timestamps are to be interpreted as if they had a trailing Z.

vr:UTCTimestamp Type Schema Documentation

A timestamp that is compliant with ISO8601 and fixes the timezone indicator, if present, to "Z" (UTC). VOResource writers should always include the timezone marker. VOResource readers must interpret timestamps without a timezone marker as UTC.

vr:UTCTimestamp Type Schema Definition

```
<xs:simpleType name="UTCTimestamp" >
  <xs:restriction base="xs:dateTime" >
    <xs:pattern
             value="\d{4}-\d\d-\d\dT\d\d:\d\d:\d\d(\.\d+)?Z?" />
  </xs:restriction>
</xs:simpleType>
```

Where day granularity might suffice for some instance documents, VOResource and extensions can employ the *vr:UTCDateTime* type.

vr:UTCDateTime Type Schema Documentation

A date stamp that can be given to a precision of either a day (type xs:date) or seconds (type xs:dateTime). Where only a date is given, it is to be interpreted as the span of the day on the UTC timezone if such distinctions are relevant.

vr:UTCDateTime Type Schema Definition

```
<xs:simpleType name="UTCDateTime" >
  <xs:union
         memberTypes="xs:date vr:UTCTimestamp" />
</xs:simpleType>
```

2.2.5 VOResource Semantics

All VOResource types and elements have an associated semantic meaning which is given in the first *xsd:documentation* node within the type or element's definition in the schema. The meaning associated with a type is generic, indicating the kind of information the type provides. The semantics that are delivered by a VOResource instance document, however, are those associated with VOResource elements. The meaning of a VOResource element can be thought of as having two parts: the generic meaning of the set of information it contains as defined by its type, and a specific meaning describing the context in which that information applies. Because all VOResource elements are locally defined (in the XML Schema sense), they do not have an absolute meaning, but rather have a meaning tied to the thing being described by that element as represented by the enclosing type.

Here are three examples that illustrate the semantics communicated by VOResource entities:

1. The *vr:Curation* type describes the curation of a resource. The *curation* element describes curation of the specific resource described by the enclosing *vr:Resource* type and identified by its *identifier* element.

2. The *vr:ResourceName* type is a generic reference to another resource. The *publisher* element gives a reference to the publisher of the specific resource being described which may itself be a registered resource described elsewhere.

3. The *title* element gives the title of the resource being described by the enclosing *vr:Resource* type and identified by its *identifier* element. The *title* element's type, *xs:token* (a restriction on *xsd:string*), has no inherent meaning associated with it.

Additional semantics are transmitted through the use of derived types using the *xsi:type* attribute. In the sample instance document above, the use of `xsi:type="vr:Organisation"` means that the resource being described is specifically an organisation as defined by the *vr:Organisation* type. This type provides additional metadata that are not part of the core resource metadata. The semantics associated with the use of *xsi:type* is described further in the next subsection.

2.2.6 Refined Semantics with Derived Types

When a resource is described with an element explicitly of the type *vr:Resource*, it is being described in the most generic sense. The metadata presented in this type, including both free text values and controlled vocabulary, can give some sense of what type of resource is being described and what it might be used for. However, the most useful descriptions of resources will not explicitly use the *vr:Resource* type; rather, they will use types that are derived from *vr:Resource*.

Defining derived *vr:Resource* types accomplishes two things:

1. it sharpens the semantic meaning of the resource description by indicating what specific type of resource it is, and

2. it *may* allow additional metadata not part of the core but specific to that type of resource.

The VOResource schema defines two types derived from *vr:Resource*: *vr:Organisation* and *vr:Service*. The *vr:Organisation* adds metadata describing the astronomical facilities such as telescopes that are associated with the organisation it describes. The *vr:Service* type adds an element called *capability* which describes the service's interface as well as information regarding its behavior.

Extension of the *vr:Resource* type is a key way in which derivation is used in VOResource to provide refined resource descriptions. Two other important parent types in the schema are *vr:Capability* and *vr:Interface*; these are extended to provide more refined descriptions of services (see sect. 2.2.7). The motivation for extending these types is the same as for *vr:Resource*: to provide more specific semantic meaning through the derived type's name, and to provide additional, specialized metadata that is not part of the parent type. Note, however, that in general, a derived type need not alter the content model of its base type. This allows derived types to add more specific meaning without adding any additional metadata.

As described in sect. 2.2, it is intended that derived *vr:Resource*, *vr:Capability* and *vr:Interface* types be invoked in instance documents using the *xsi:type* attribute (as illustrated in the sample document above). This method illustrates a polymorphism for resource metadata in that any place where an element of parent type is expected, the derived type may be inserted. The use of *xsi:type* illustrates both what specific type is being inserted in as well as what it is being inserted in for. That is, as in our example, the *resource* being described is an *organisation*.

The other mechanism for polymorphism provided by XML Schema is substitution groups. Invoking derived *vr:Resource* types via elements in a substitution group is discouraged because it is less obvious from looking at the instance document that a substitution is being made.

2.2.7 The Service Data Model

The *vr:Service* type extends the core *vr:Resource* metadata data by adding the *capability* element (see sect. 2.3.2). This element is used to describe a major functionality of the service, usually accessible through a single service endpoint URL. In particular, it is used to describe support for an IVOA service standard (e.g. Simple Image Access Protocol). A service resource record may have multiple child *capability* elements, each describing a different major functionality; however, these capabilities should be related in an obvious, logical way (they would hardly share their core VOResource metadata otherwise).

The *capability* element, through its type *vr:Capability*, describes the behavior of service capability and how to access it. The latter is described by a child *interface* element. As for the behavior, the base *vr:Capability* type only provides a *description* element that can contain human-readable text on what this capability provides. More structured behavioral information must be provided through specialized *vr:Capability* extensions. In particular, a service standard (e.g. Simple Image Access Protocol) or an ancillary standard built upon the service standard will define an extension of *vr:Capability* that adds additional metadata that can describe the service's behavior in relation to the standard.

The added metadata can describe limitations of the service implementation or indicate support for optional features. As a rough guideline, whenever the standard has a SHOULD or MAY in the sense of RFC2119 and clients have no other way to discover the choices of a concrete service implementation, a respective declaration should be considered for the capability or the interface elements defined by the registry extension.

The specific *vr:Capability* type is invoked using the *xsi:type* mechanism described in sect. 2.1. Here is an example for assigning a specialized Capability type for a standard service capability. In this example, it is assumed that *ExampleCapType* extension type is defined in a separate schema document, the target namespace of which is being bound to the *ex:* prefix in the capability element itself (it could, of course, be bound in some enclosing element just as well):

```
<capability xsi:type="ex:ExampleCapType"
  standardID="ivo://example.com/std/exampleAccess"
  xmlns:ex="http://ivoa.net/std/example-1.xsd">
   ...
</capability>
```

Note that the *xsi:type* and the *standardID* can vary independently of each other. There is no requirement to use a given capability type for an endpoint conforming to a certain standard, as indicated by the capability's *standardID*. Conversely, it may make sense to re-use a certain capability type in a different standard.

Historically, several VOResource extension types have had fixed *standardID*s. This is now discouraged, as it needlessly limits the applicability of the encoded metadata models.

If the service capability being described does not conform to any standard or if the standard does not require any specialized capability metadata for describing an implementation's behavior, then *vr:Capability* can be used as-is, possibly with the appropriate *standardID*.

Each *capability* element can contain one or more child *interface* elements, each describing how the capability can be accessed. The *interface* element's type, *vr:Interface*, is abstract; thus, the *interface* element must be accompanied by an *xsi:type* attribute that indicates a *vr:Interface* extension type. The VOResource schema defines two *vr:Interface* extension types: *vr:WebBrowser*, for describing an interface access via web browser, and *vr:WebService*, for accessing a service described by a Web Service Description Language (WSDL) document. In today's VO, the latter interface type has almost vanished with the general move away from SOAP-based architectures. VO standard capabilities currently typically use interface types defined in VODataService.[2]

As an example, a simple, browser-based interface could be declared with an untyped capability like this (this assumes the *vr* prefix has been bound in an ancestor element):

```
<capability>
  <interface xsi:type="vr:WebBrowser">
    <accessURL use="full"
      >http://example.org/browser-service</accessURL>
  </interface>
</capability>
```

Note that for less trivial interfaces, in particular *vs:ParamHTTP*, the parameters accepted should be declared in the interface element.

When a *capability* contains more than one *interface*, each *interface* should be interpreted as an alternative interface for accessing essentially the same underly-

[2]Retrospectively, *vs:ParamHTTP* should probably have been part of VOResource itself; changing this now, however, does not seem to be worth the migration effort.

ing capability. The interfaces can differ in their overall type (e.g. `vr:WebBrowser`, `vr:WebService`) or in the supported input parameters or output products.

When a standard capability is being described (by means of providing a `standardID`), then at least one of the `interface` elements should describe an interface required by the standard. The `role` attribute is used to mark the standard interfaces (typically with the value "std"). All other interfaces are considered non-standard alternatives. To best support real-world discovery strategies, it is recommended to avoid having non-standard interfaces in capabilities with well-known `standardID`s.

Another way `interface`s inside the same `capability` element can be used is if a standard forsees different versions of a protocol within the same capability. Different interfaces can then be distinguished by different values of the interface's `version` attribute. Previous versions of this document in effect recommended this pattern for VO standards.

While this pattern can still be employed, most VO standards that actually have different versions distinguish their endpoints by different standard identifiers, as described in section 4.2 of IVOA Identifiers 2.0 (Demleitner and Plante et al., 2016). In these cases – where the `capability`'s `standardID` attribute already uniquely determines the protocol spoken –, the specification of `version` on `interface` elements is optional (although encouraged).

2.3 Extending the VOResource Schema

A schema made up only of global type definitions provides great flexibility for extension. Any global type defined in the VOResource schema may be used as the base of a derived type defined in another schema. The schema containing the derived types must declare its own namespace URI or default to the null namespace; it must not adopt the VOResource namespace URI. The application must then define what schemas, identified by their namespace URIs, are supported and/or required.

A **VOResource extension** is an XML Schema document whose primary purpose is to define new types derived from those defined in the VOResource schema for use in resource descriptions. It is recommended that VOResource extensions follow the definition styles used by the core VOResource. In particular:

- *Provide semantic definitions of all types and elements within the first `xsd:documentation` element inside the type or element definition.* Subsequent `xsd:documentation` elements may provide additional comments or discussion.

- *Avoid the use of `xsd:choice` elements.* VOResource does not use the choice structure because it does not map readily into any object-oriented software language structure. Choices are handled instead as multiple derived types that can be inserted in place of a parent type; for examples, see the `vs:VOTableType` and `vs:SimpleDataType` defintions in VODataService 1.1 (Plante and Stébé et al., 2010).

- *Avoid the use of substitution groups.* VOResource prefers instead the use of `xsi:type` which are (with a few exceptions) functionally equivalent to substitution groups in terms of structure; however, `xsi:type` serves as an obvious flag in the instance document that a substitution has been made.

- *Choose semantically meaningful names for derived types.* When the derived type appears in the pattern `<elname xsi:type="derivedType">`, choose a *derivedType* name such that the sentence, "a *derivedType* is a kind of *elname*" makes natural and obvious sense. For example, "an *Organisation* is a kind of *resource*."

- *Follow the VOResource naming conventions.*

There are two types of derivation that are particularly important to the VOResource data model: derivation of the *`vr:Resource`* type, used to define specific types of resources, and the derivation of service metadata elements.

2.3.1 Defining New Resource Types

Derivation of *`vr:Resource`* to define new kinds of resources should be done by extension (i.e. using *`xsd:extension`*) rather than restriction. It is not required that the derived type change the content model from that of the *`vr:Resource`* base type; in this case, the purpose of the derivation is only to sharpen the semantic meaning of the resource description.

In VOResource itself, this mechanism is used to derive *`vr:Organisation`* and *`vr:Service`*.

2.3.2 Defining New Service Capabilities and Interfaces

As described in sect. 2.2.7, a service standard will often define a new *`vr:Capability`* extension type to allow implementations to describe how they support the standard. This definition of the *`vr:Capability`* extension should be done in a schema document with a namespace identifier that is dedicated to that standard (hereafter referred to as *the standard's extension schema*). The extension type should include elements representing the applicable Capability metadata described in section 5.2 of the RM (e.g. *Service.MaxReturnRecords*, *Service.MaxReturnSize*) but can also include other concepts that are specific to that standard.

An extension schema can define new interface types, though not necessarily in the context of any specific standard service capability. The basic *`vr:Interface`* type provides only *`accessURL`*, *`mirrorURL`*, and *`securityMethod`* as child elements. A derived *`vr:Interface`* type must indicate in the documentation how the *`accessURL`* should be interpreted and used. The derived type may also include other added metadata describing how to use the service (e.g., a description of the input arguments). If the interface extension type is expected to be referenced by a standard service capability, then it is recommended that the additional metadata be optional unless the metadata is absolutely required by clients in order to invoke the service.

Examples for extension schemas can be found in the IVOA Document Repository.[3] Sect. 1.1 enumerates the ones in Recommendation state at the time of writing. In VOResource itself, the mechanism can be inspected in the derivation of *`vr:WebBrowser`* and *`vr:WebService`*.

[3]`http://ivoa.net/documents`; TAPRegExt or SimpleDALRegExt give starting points. The actual schema files are available from `http://ivoa.net/xml`.

3 The VOResource Metadata

This section enumerates the types and elements defined in the VOResource schema.

3.1 The Base Resource Type

A resource, as defined by the RM, is any entity or component of a VO application that is describable and identifiable by an IVOA Identifier. A resource is described using VOResource by an element of the type *vr:Resource* or one of its legal extensions. The schema definition (below) includes elements that encode the identity, curation, and general content metadata for a resource (see sections 3.1 through 3.3 of the RM). The RM states that certain metadata are required in a minimally compliant resource description; this requirement is enforced by the VOResource schema definition.

vr:Resource Type Schema Documentation

Any entity or component of a VO application that is describable and identifiable by an IVOA Identifier.

vr:Resource Type Schema Definition

```
<xs:complexType name="Resource" >
  <xs:sequence >
    <xs:element name="validationLevel" type="vr:Validation" minOccurs="0"
             maxOccurs="unbounded" />
    <xs:element name="title" type="xs:token" />
    <xs:element name="shortName" type="vr:ShortName" minOccurs="0" />
    <xs:element name="identifier" type="vr:IdentifierURI" />
    <xs:element name="altIdentifier" type="xs:anyURI" minOccurs="0"
             maxOccurs="unbounded" />
    <xs:element name="curation" type="vr:Curation" />
    <xs:element name="content" type="vr:Content" />
  </xs:sequence>
  <xs:attribute name="created" type="vr:UTCTimestamp" use="required" />
  <xs:attribute name="updated" type="vr:UTCTimestamp" use="required" />
  <xs:attribute name="status" use="required" >
    <xs:simpleType >
      <xs:restriction base="xs:string" >
        <xs:enumeration value="active" />
        <xs:enumeration value="inactive" />
        <xs:enumeration value="deleted" />
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="version" type="xs:token" />
</xs:complexType>
```

vr:Resource Attributes

created

Type	string
Meaning	The UTC date and time this resource metadata description was created.
Occurrence	required

Comment This timestamp must not be in the future. This time is not required to be accurate; it should be at least accurate to the day. Any insignificant time fields should be set to zero.

updated

Type string

Meaning The UTC date this resource metadata description was last updated.

Occurrence required

Comment This timestamp must not be in the future. This time is not required to be accurate; it should be at least accurate to the day. Any insignificant time fields should be set to zero.

status

Type string with controlled vocabulary

Meaning a tag indicating whether this resource is believed to be still actively maintained.

Occurrence required

Allowed Values

active
: resource is believed to be currently maintained, and its description is up to date (default).

inactive
: resource is apparently not being maintained at the present.

deleted
: resource publisher has explicitly deleted the resource.

version

Type string: `xs:token`

Meaning The VOResource XML schema version against which this instance was written. Implementors should set this to the value of the version attribute of their schema's root (xs:schema) element. Clients may assume version 1.0 if this attribute is missing.

Occurrence optional

`vr:Resource` Metadata Elements

Element `validationLevel`

Type `vr:ValidationLevel` with optional attributes

Meaning A numeric grade describing the quality of the resource description, when applicable, to be used to indicate the confidence an end-user can put in the resource as part of a VO application or research study.

Occurrence optional; multiple occurrences allowed.

Comment See vr:Validation for an explanation of the allowed levels.

Comment Note that when this resource is a Service, this grade applies to the core set of metadata. Capability and interface metadata, as well as the compliance of the service with the interface standard, is rated by validationLevel tag in the capability element (see the vr:Service complex type).

Element `title`

Type string: `xs:token`

Meaning the full name given to the resource

Occurrence required

Element `shortName`

Type string

Meaning A short name or abbreviation given to the resource.

Occurrence optional

Comment This name will be used where brief annotations for the resource name are required. Applications may use to refer to this resource in a compact display.

Comment One word or a few letters is recommended. No more than sixteen characters are allowed.

Element ***identifier***

Type an IVOA Identifier URI: vr:IdentifierURI

Meaning Unambiguous reference to the resource conforming to the IVOA standard for identifiers

Occurrence required

Element ***altIdentifier***

Type a URI: ***xs:anyURI***

Meaning A reference to this resource in a non-IVOA identifier scheme, e.g., DOI or bibcode. Always use the an URI scheme here, e.g., doi:10.1016/j.epsl.2011.11.037. For bibcodes, use a form like bibcode:2008ivoa.spec.0222P.

Occurrence optional; multiple occurrences allowed.

Element ***curation***

Type composite: ***vr:Curation***

Meaning Information regarding the general curation of the resource

Occurrence required

Element ***content***

Type composite: ***vr:Content***

Meaning Information regarding the general content of the resource

Occurrence required

Note that the ***vr:Resource*** attributes (***created***, ***updated***, ***status***) represent a special class of metadata: they describe the resource metadata contained within the ***vr:Resource*** itself as opposed to the resource being described. Also note that with OAI-PMH 2.0, VOResource records that would have a ***status*** of `deleted` should not occur since OAI-PMH mandates that ***oai:metadata*** is empty for deleted records, and VOResource's ***status*** essentially is a refinement of OAI-PMH's ***status***. Having VOResource records with ***status*** set to `deleted` might still be useful outside of OAI-PMH.

The ***version*** attribute of an ***vr:Resource*** element gives the version of the schema it was written against, as prescribed in the IVOA Note "XML Schema Versioning Policies" (**?**). If missing, the value of this attribute can be assumed to be 1.0. This attribute should *not* be used to locate the schema file – clients should either consult ***schemaLocation*** or retrieve the latest schema for the ***vr:*** namespace from the IVOA schema repository.

The following sections discuss the complex types mentioned in the above definitions. All rules and advice given in the "Comments" portions in the RM definition for the individual items apply. In some details, in particular as regards enumerations of values, this document deviates from RM in content.

3.1.1 Identity Metadata

The identity metadata described in the RM (section 3.1) are represented as top-level children of the ***vr:Resource*** type.

Two special types, *vr:ShortName* and *vr:IdentifierURI* are defined to support identity metadata. The *vr:ShortName* definition enforces the 16-character limit on shortNames.

vr:ShortName Type Schema Documentation

A short name or abbreviation given to something.

This name will be used where brief annotations for the resource name are required. Applications may use to refer to this resource in a compact display.

One word or a few letters is recommended. No more than sixteen characters are allowed.

vr:ShortName Type Schema Definition

```
<xs:simpleType name="ShortName" >
  <xs:restriction base="xs:token" >
    <xs:maxLength value="16" />
  </xs:restriction>
</xs:simpleType>
```

vr:IdentifierURI Type Schema Documentation

A reference to a registry record.

This type should only be used if what is referenced must actually be a true Registry record; vr:IdentifierURI does not allow query or fragment parts and is hence not suitable for everything defined by IVOA Identifiers, in particular not standard keys (which are used for versions of standards, for instance) or dataset identifiers.

When something does not need to be locked down to a reference to a single registry record, xs:anyURI should be used.

vr:IdentifierURI Type Schema Definition

```
<xs:simpleType name="IdentifierURI" >
  <xs:restriction base="xs:anyURI" >
    <xs:pattern
              value="ivo://[\w\d][\w\d\-_\.!~\*'\(\)\+=]{2,}(/[\w\d\-_««
                  \.!~\*'\(\)\+=]+(/[\w\d\-_\.!~\*'\(\)\+=]+)*)?" />
  </xs:restriction>
</xs:simpleType>
```

3.1.2 Curation Metadata

The curation metadata described in the RM (section 3.2) are bundled together into the *vr:Resource* child element *curation*. Its content is defined by the *vr:Curation* complex type.

vr:Curation Type Schema Documentation

Information regarding the general curation of a resource

vr:Curation Type Schema Definition

```
<xs:complexType name="Curation" >
  <xs:sequence >
    <xs:element name="publisher" type="vr:ResourceName" />
    <xs:element name="creator" type="vr:Creator" minOccurs="0"
              maxOccurs="unbounded" />
    <xs:element name="contributor" type="vr:ResourceName" minOccurs="0"
              maxOccurs="unbounded" />
    <xs:element name="date" type="vr:Date" minOccurs="0"
              maxOccurs="unbounded" />
    <xs:element name="version" type="xs:token" minOccurs="0" />
    <xs:element name="contact" type="vr:Contact"
              maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>
```

vr:Curation Metadata Elements

Element *publisher*

Type string with ID attribute: vr:ResourceName

Meaning Entity (e.g. person or organisation) responsible for making the resource available

Occurrence required

Element *creator*

Type composite: *vr:Creator*

Meaning The entity/ies (e.g. person(s) or organisation) primarily responsible for creating the content or constitution of the resource.

Occurrence optional; multiple occurrences allowed.

Comment This is the equivalent of the author of a publication.

Element *contributor*

Type string with ID attribute: vr:ResourceName

Meaning Entity responsible for contributions to the content of the resource

Occurrence optional; multiple occurrences allowed.

Element *date*

Type *vr:UTCDateTime* with optional attributes

Meaning Date associated with an event in the life cycle of the resource.

Occurrence optional; multiple occurrences allowed.

Comment This will typically be associated with the creation or availability (i.e., most recent release or version) of the resource. Use the role attribute to clarify.

Element *version*

Type string: *xs:token*

Meaning Label associated with creation or availablilty of a version of a resource.

Occurrence optional

Element *contact*

Type composite: *vr:Contact*

Meaning Information that can be used for contacting someone with regard to this resource.

Occurrence required; multiple occurrences allowed.

Resource Names Several of the curation elements (most importantly, *publisher*) make use of the *vr:ResourceName* type. This type provides a means of referring to another resource both by name and by its IVOA identifier. Not all resources referred to using this type will necessarily be registered and, therefore, will have an identifier; thus, the identifier (which is encoded as an attribute) is optional.

vr:ResourceName Type Schema Documentation

The name of a potentially registered resource. That is, the entity referred to may have an associated identifier.

vr:ResourceName Type Schema Definition

```
<xs:complexType name="ResourceName" >
  <xs:simpleContent >
    <xs:extension base="xs:token" >
      <xs:attribute name="ivo-id" type="vr:IdentifierURI" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

vr:ResourceName Attributes

ivo-id

Type	an IVOA Identifier URI: vr:IdentifierURI
Meaning	The IVOA identifier for the resource referred to.
Occurrence	optional

The Creator Type The *creator* element is defined by the *vr:Creator* complex type which bundles together the RM metadata *Creator* and *Creator.Logo*.

vr:Creator Type Schema Documentation

The entity (e.g. person or organisation) primarily responsible for creating something

vr:Creator Type Schema Definition

```
<xs:complexType name="Creator" >
  <xs:sequence >
    <xs:element name="name" type="vr:ResourceName" />
    <xs:element name="logo" type="xs:anyURI" minOccurs="0" />
    <xs:element name="altIdentifier" type="xs:anyURI" minOccurs="0"
             maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="ivo-id" type="vr:IdentifierURI" />
</xs:complexType>
```

vr:Creator Attributes

ivo-id

Type	an IVOA Identifier URI: vr:IdentifierURI
Meaning	An IVOA identifier for the creator (typically when it is an organization).
Occurrence	optional

vr:Creator Metadata Elements

Element *name*

Type string with ID attribute: vr:ResourceName

Meaning the name or title of the creating person or organisation

Occurrence required

Comment Users of the creation should use this name in subsequent credits and acknowledgements.
This should be exactly one name, preferably last name first (as in "van der Waals, Johannes Diderik").

Element *logo*

Type a URI: *xs:anyURI*

Meaning URL pointing to a graphical logo, which may be used to help identify the information source

Occurrence optional

Comment A logo needs only be provided for the first occurrence. When multiple logos are supplied via multiple creator elements, the application is free to choose which to use.

Element *altIdentifier*

Type a URI: *xs:anyURI*

Meaning A reference to this entitiy in a non-IVOA identifier scheme, e.g., orcid. Always use a URI form including a scheme here.

Occurrence optional; multiple occurrences allowed.

The *Creator*'s *name* should contain exactly one name. Since the creator name is what Registry users use when searching for resources "by author", it should be set with care. To make sensible collation simple for clients, person names should be formatted last name first wherever appropriate.

In collaborative efforts, multiple creator elements should be used, one element per person or organisation involved.

Examples for encouraged use of creator include:

```
<creator>
  <name>van der Waals, Johannes Diderik</name>
</creator>
<creator>
  <name>de Vaucouleurs, G.</name>
</creator>
<creator>
  <name>Zhang Hailong</name>
</creator>
<creator>
  IVOA Registry Working Group
</creator>
```

While the following uses of creator do not make a resource record technically invalid, they are frowned upon:

```
<creator>
  <!-- do not include multiple names in one name element -->
  <name>Mornilov, V.G., Polkov, I.M., Bakharov, A.I</name>
</creator>
<creator>
  <!-- Do not include functions or similar -->
  <name>PI: Ita Varajedini</name>
</creator>
<creator>
```

```
  <!-- Do not abuse creator for provenance, do not include markup -->
  <name>All data is downloaded from the &lt;a href=http:...</name>
</creator>
```

The typical identifier type in *altIdentifier* in role-like elements (at this point, creator and contact) is an ORCID. In accordance with ORCID's instructions[4] (and contrary to Working Drafts of this document), ORCIDs must be given as HTTPS URIs, for instance `https://orcid.org/0000-0001-2345-6789`.

Dates and Their Roles The *Date* element's type, *vr:Date*, is an extension of the *UTCDateTime* type that adds an optional attribute called *role*.

The purpose of the role attribute is to indicate what aspect of the resource the date describes. This allows several *date* elements to be provided, each with a different role. The possible values for this attribute are not controlled by the schema. The IVOA, however, defines an RDF vocabulary at `http://www.ivoa.net/rdf/voresource/date_role` from which the *role* terms should be taken if at all possible. At the time of writing, the vocabulary consists of three old-style VOResource 1.0 terms and terms from the DataCite Metadata 4.0 (DataCite Metadata Working Group, 2016); see the RDF URL for the definitions of the terms:

> *Accepted*, *Available*, *Collected*, *Copyrighted*, *Created*, *Issued*, *Submitted*, *Updated*, *Valid creation*, *representative*, *update*,

Terms can be added to this vocabulary without changes to this specification. Applications are not expected to support semantic relationships between the terms (except for mapping the deprecated VOResource 1.0 terms to the modern DataCite Metadata ones).

vr:Date Type Schema Definition

```
<xs:complexType name="Date" >
  <xs:simpleContent >
    <xs:extension base="vr:UTCDateTime" >
      <xs:attribute name="role" type="xs:string"
                    default="representative" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

vr:Date Attributes

role

Type	string: *xs:string*
Meaning	A string indicating what the date refers to.
Occurrence	optional representative
Comment	The value of role should be taken from the vocabulary maintained at http://www.ivoa.net/rdf/voresource/date_role. This includes the traditional and deprecated strings "creation", indicating the date that the resource itself was created, and "update", indicating when the resource was updated last, and the default value, "representative", meaning the date

[4]`https://support.orcid.org/knowledgebase/articles/116780`

is a rough representation of the time coverage of the resource. The preferred terms from that vocabulary are the DataCite Metadata terms. It is expected that the vocabulary will be kept synchronous with the corresponding list of terms in the DataCite Metadata schema.

Comment Note that this date refers to the resource; dates describing the metadata description of the resource are handled by the “created” and “updated” attributes of the Resource element.

The Contact Type The `contact` element is defined by the `vr:Contact` type which bundles together several component pieces of information, including the RM metadata *Contact.Name* and *Contact.Email.*

`vr:Contact` Type Schema Documentation

Information allowing establishing contact, e.g., for purposes of support.

`vr:Contact` Type Schema Definition

```
<xs:complexType name="Contact" >
  <xs:sequence >
    <xs:element name="name" type="vr:ResourceName" />
    <xs:element name="address" type="xs:token" minOccurs="0" />
    <xs:element name="email" type="xs:token" minOccurs="0" />
    <xs:element name="telephone" type="xs:token" minOccurs="0" />
    <xs:element name="altIdentifier" type="xs:anyURI" minOccurs="0"
              maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="ivo-id" type="vr:IdentifierURI" />
</xs:complexType>
```

`vr:Contact` Attributes

ivo-id

Type an IVOA Identifier URI: vr:IdentifierURI

Meaning An IVOA identifier for the contact (typically when it is an organization).

Occurrence optional

`vr:Contact` Metadata Elements

Element `name`

Type string with ID attribute: vr:ResourceName

Meaning the name or title of the contact person.

Occurrence required

Comment This can be a person’s name, e.g. “John P. Jones” or a group, “Archive Support Team”.

Element `address`

Type string: `xs:token`

Meaning the contact mailing address

Occurrence optional

Comment All components of the mailing address are given in one string, e.g. “3700 San Martin Drive, Baltimore, MD 21218 USA”.

Element *email*

Type	string: *xs:token*
Meaning	the contact email address
Occurrence	optional

Element *telephone*

Type	string: *xs:token*
Meaning	the contact telephone number
Occurrence	optional
Comment	Complete international dialing codes should be given, e.g. "+1-410-338-1234".

Element *altIdentifier*

Type	a URI: *xs:anyURI*
Meaning	A reference to this entitiy in a non-IVOA identifier scheme, e.g., orcid. Always use a URI form including a scheme here.
Occurrence	optional; multiple occurrences allowed.

3.1.3 General Content Metadata

The general content metadata described in the RM (section 3.3) are bundled together into the *vr:Resource* child element *content*. Its content is defined by the *vr:Content* complex type.

vr:Content Type Schema Documentation

Information regarding the general content of a resource

vr:Content Type Schema Definition

```
<xs:complexType name="Content" >
  <xs:sequence >
    <xs:element name="subject" type="xs:token"
            maxOccurs="unbounded" />
    <xs:element name="description" type="xs:string" />
    <xs:element name="source" type="vr:Source" minOccurs="0" />
    <xs:element name="referenceURL" type="xs:anyURI" />
    <xs:element name="type" type="xs:token" minOccurs="0"
            maxOccurs="unbounded" />
    <xs:element name="contentLevel" type="xs:token" minOccurs="0"
            maxOccurs="unbounded" />
    <xs:element name="relationship" type="vr:Relationship" minOccurs="0"
            maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>
```

vr:Content Metadata Elements

Element *subject*

Type	string: *xs:token*
Meaning	a topic, object type, or other descriptive keywords about the resource.
Occurrence	required; multiple occurrences allowed.
Comment	Terms for Subject should be drawn from the Unified Astronomy Thesaurus (http://astrothesaurus.org).

Element `description`

Type string: `xs:string`

Meaning An account of the nature of the resource

Occurrence required

Comment The description may include but is not limited to an abstract, table of contents, reference to a graphical representation of content or a free-text account of the content.
Note that description is xs:string-typed, which means that whitespace is considered significant. Clients should render empty lines as paragraph boundaries and ideally refrain from reflowing material that looks formatted (i.e., is broken to about 80-character lines).

Element `source`

Type a string with optional attributes

Meaning a bibliographic reference from which the present resource is derived or extracted.

Occurrence optional

Comment This is intended to point to an article in the published literature. An ADS Bibcode is recommended as a value when available.

Element `referenceURL`

Type a URI: `xs:anyURI`

Meaning URL pointing to a human-readable document describing this resource.

Occurrence required

Element `type`

Type string: `xs:token`

Meaning Nature or genre of the content of the resource. Values for type should be taken from the controlled vocabulary http://www.ivoa.net/rdf/voresource/content_type

Occurrence optional; multiple occurrences allowed.

Element `contentLevel`

Type string: `xs:token`

Meaning Description of the content level or intended audience. Values for contentLevel should be taken from the controlled vocabulary http://www.ivoa.net/rdf/voresource/content_level.

Occurrence optional; multiple occurrences allowed.

Element `relationship`

Type composite: `vr:Relationship`

Meaning a description of a relationship to another resource.

Occurrence optional; multiple occurrences allowed.

The content of `contentLevel` should conform to the values from a vocabulary. At the time of writing, this vocabulary contained the terms:

Research, *Amateur*, *General*

For definitions of these terms, their relationships, and the complete, up-to date list of the legal terms in HTML, turtle, or RDF-X formats, please refer to `http://www.ivoa.net/rdf/voresource/content_level`.

The content of `type` should conform to the values from a vocabulary. At the time of writing, this vocabulary contained the terms:

Other, *Archive*, *Bibliography*, *Catalog*, *Journal*, *Library*, *Simulation*, *Survey*, *Transformation*, *Education*, *Outreach*, *EPOResource*, *Animation*, *Artwork*, *Background*, *BasicData*, *Historical*, *Photographic*, *Press*, *Organisation*, *Project*, *Registry*

For definitions of these terms, their relationships, and the complete, up-to date list of the legal terms in HTML, turtle, or RDF-X formats, please refer to `http://www.ivoa.net/rdf/voresource/content_type`.

The *`source`* element's type, *`vr:Source`*, is an extension of the *`xs:token`* type that adds an optional attribute called *`format`*.

`vr:Source` Type Schema Definition

```
<xs:complexType name="Source" >
  <xs:simpleContent >
    <xs:extension base="xs:token" >
      <xs:attribute name="format" type="xs:string" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

`vr:Source` Attributes

format

Type	string: *`xs:string`*
Meaning	The reference format. Recognized values include “bibcode”, referring to a standard astronomical bibcode (http://cdsweb.u-strasbg.fr/simbad/refcode.html).
Occurrence	optional

The *`relationship`* is defined by the *`vr:Relationship`* complex type which bundles together the RM metadata *Relationship* and *RelationshipID*.

The nature of the relationship is given by relationshipType. The values of this are taken from a vocabulary that is available in RDF form at `http://www.ivoa.net/rdf/voresource/relationshipType`. This vocabulary at the time of writing consists of

- the terms from VOResource 1.0 (mirror-of, service-for, served-by, derived-from, and related-to); these are guaranteed to remain within VOResource 1.x
- selected terms from the DataCite Metadata relationType enumeration.

The resulting mix of term styles is an aesthetic defect that may actually lead to the construction of invalid terms. While, for backward compatibility, we will not drop the VOResource 1.0 terms in VOResource 1.x, they should not be used in new records. Instead, resource records should use the terms given as *Preferred* in the vocabulary retrievable at the URL given above.

At the time of writing, the full vocabulary consists of these terms:

Cites, *Continues*, *HasPart*, *IsContinuedBy*, *IsDerivedFrom*, *IsIdenticalTo*, *IsNewVersionOf*, *IsPartOf*, *IsPreviousVersionOf*, *IsServedBy* *IsServiceFor*, *IsSourceOf*, *IsSupplementTo*, *IsSupplementedBy*, *derived-from*, *mirror-of*, *related-to*, *served-by*, *service-for*,

NOte

validationLevel is of type *vr:Validation*; somewhat confusingly, VOResource also has a type *vr:ValidationLevel*. The latter is extended by *vr:Validation* with a *validatedBy* attribute. The choice of names in this triple certainly is somewhat confusing and can be considered an editorial oversight in version 1.0. However, the effort of changing element and/or type names would seem unproportional to the merely aesthetic benefit in a part of VOResource that is rarely visible to common VO users.

For definitions of these terms, their relationships, and the complete, up-to date list of the legal terms in HTML, turtle, or RDF-X formats, please refer to http://www.ivoa.net/rdf/voresource/relationship_type.

vr:Relationship Type Schema Documentation

A description of the relationship between one resource and one or more other resources.

vr:Relationship Type Schema Definition

```
<xs:complexType name="Relationship" >
  <xs:sequence >
    <xs:element name="relationshipType" type="xs:token" />
    <xs:element name="relatedResource" type="vr:ResourceName"
              minOccurs="1"
              maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>
```

vr:Relationship Metadata Elements

Element *relationshipType*

Type	string: *xs:token*
Meaning	the named type of relationship
Occurrence	required
Comment	The value of relationshipType should be taken from the vocabulary at http://www.ivoa.net/rdf/voresource/relationship_type.

Element *relatedResource*

Type	string with ID attribute: vr:ResourceName
Meaning	the name of resource that this resource is related to.
Occurrence	required; multiple occurrences allowed.

3.1.4 Resource Record Quality with validationLevel

The RM's *ResourceValidationLevel* is encoded in the VOResource schema with the *validationLevel* element, which can appear in multiple places within a *vr:Resource* type or sub-type. It may appear zero or more times as direct children of a *vr:Resource* type and/or, if the resource is a *vr:Service* type or sub-type, zero or more times as the first children of any *capability* element.

The *validationLevel* element's attribute *validatedBy* takes an IVOID as a value. This IVOID must refer to a registered organisation or registry that assigned the numerical value. The *validationlevel* element can appear multiple times, each with a different *validatedBy* value, to reflect the code assigned by different organisations.

vr:Validation Type Schema Documentation

a validation stamp combining a validation level and the ID of the validator.

vr:Validation Type Schema Definition

```
<xs:complexType name="Validation" >
  <xs:simpleContent >
    <xs:extension base="vr:ValidationLevel" >
      <xs:attribute name="validatedBy" type="xs:anyURI" use="required" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

vr:Validation Attributes

validatedBy

Type a URI: *xs:anyURI*

Meaning The IVOA ID of the registry or organisation that assigned the validation level.

Occurrence required

vr:ValidationLevel Type Schema Documentation

The allowed values for describing the resource descriptions and interfaces.

See the RM (v1.1, section 4) for more guidance on the use of these values.

vr:ValidationLevel Type Allowed Values

0

The resource has a description that is stored in a registry. This level does not imply a compliant description.

1

In addition to meeting the level 0 definition, the resource description conforms syntactically to this standard and to the encoding scheme used.

2

In addition to meeting the level 1 definition, the resource description refers to an existing resource that has demonstrated to be functionally compliant.

3

In addition to meeting the level 2 definition, the resource description has been inspected by a human and judged to comply semantically to this standard as well as meeting any additional minimum quality criteria (e.g., providing values for important but non-required metadata) set by the human inspector.

4
: In addition to meeting the level 3 definition, the resource description meets additional quality criteria set by the human inspector and is therefore considered an excellent description of the resource. Consequently, the resource is expected to operate well as part of a VO application or research study.

`vr:ValidationLevel` Type Schema Definition

```
<xs:simpleType name="ValidationLevel" >
  <xs:restriction base="xs:integer" >
    <xs:whiteSpace value="collapse" />
    <xs:enumeration value="0" />
    <xs:enumeration value="1" />
    <xs:enumeration value="2" />
    <xs:enumeration value="3" />
    <xs:enumeration value="4" />
  </xs:restriction>
</xs:simpleType>
```

3.2 Resource Type Extensions: Organisation and Service

The VOResource schema defines two extensions of the base *vr:Resource* type for describing two specific types of resources: *vr:Organisation* and *vr:Service*. In addition to providing more refined semantic meanings over *vr:Resource*, they add additional metadata for describing the resource which do not necessarily apply in the generic case.

3.2.1 The Organisation Resource Type

The *vr:Organisation* resource type is used to describe an organisation in the sense defined by the RM:

> An organisation is a specific type of resource that brings people together to pursue participation in VO applications. Organisations can be hierarchical and range greatly in size and scope. At a high level, an organisation could be a university, observatory, or government agency. At a finer level, it could be a specific scientific project space mission, or individual researcher.

The Organisation type extends the Resource type by adding two additional elements to the core set of metadata, *facility* and *instrument*:

`vr:Organisation` Type Schema Documentation

A named group of one or more persons brought together to pursue participation in VO applications.

According to the Resource Metadata Recommendation, organisations "can be hierarchical and range in size and scope. At a high level, an organisation could be a university, observatory, or government agency. At a finer level, it could be a specific scientific project, mission, or individual researcher."

The main purpose of an organisation as a registered resource is to serve as a publisher of other resources.

vr:Organisation Type Schema Definition

```
<xs:complexType name="Organisation" >
  <xs:complexContent >
    <xs:extension base="vr:Resource" >
      <xs:sequence >
        <xs:element name="facility" type="vr:ResourceName" minOccurs="0"
                maxOccurs="unbounded" />
        <xs:element name="instrument" type="vr:ResourceName" minOccurs="0"
                maxOccurs="unbounded" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

vr:Organisation Extension Metadata Elements

Element *facility*

Type	string with ID attribute: vr:ResourceName
Meaning	the observatory or facility used to collect the data contained or managed by this resource.
Occurrence	optional; multiple occurrences allowed.

Element *instrument*

Type	string with ID attribute: vr:ResourceName
Meaning	the Instrument used to collect the data contain or managed by a resource.
Occurrence	optional; multiple occurrences allowed.

The main role of an organisation in the VO (that is, the main reason for describing organisations in a registry) is as a provider or publisher of other resources. In particular, an organisation description in a registry declares the association of an IVOA identifier with the organisation. The organisation can then be referred to in other resource descriptions. For example, an organisation identifier will appear as the publisher identifier of service resource (as illustrated in the opening example in sect. 2).

3.2.2 The Service Resource Type

The *vr:Service* resource type is used to describe a service–a resource that actually does something – in the sense defined by the RM:

> A service is any VO resource that can be invoked by the user to perform some action on their behalf.

The general data model is described in sect. 2.2.7. The Service type extends the Resource type by adding two elements: *rights* which indicates the modalities of service use and access, and *capability* which describes how the service behaves and how it is invoked.

vr:Service Type Schema Documentation

a resource that can be invoked by a client to perform some action on its behalf.

vr:Service Type Schema Definition

```
<xs:complexType name="Service" >
  <xs:complexContent >
    <xs:extension base="vr:Resource" >
      <xs:sequence >
        <xs:element name="rights" type="vr:Rights" minOccurs="0"
                  maxOccurs="unbounded" />
        <xs:element name="capability" type="vr:Capability" minOccurs="0"
                  maxOccurs="unbounded" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

vr:Service Extension Metadata Elements

Element *rights*

Type	a string with optional attributes
Meaning	Information about rights held in and over the resource.
Occurrence	optional; multiple occurrences allowed.
Comment	Mainly for compatibility with DataCite, this element is repeatable. Resource record authors are advised that within the Virtual Observatory clients will typically only display and/or use the rights element occurring first and ignore later elements.

Element *capability*

Type	composite: *vr:Capability*
Meaning	a description of a general capability of the service and how to use it.
Occurrence	optional; multiple occurrences allowed.
Comment	This describes a general function of the service, usually in terms of a standard service protocol (e.g. SIA), but not necessarily so.
Comment	A service can have many capabilities associated with it, each reflecting different aspects of the functionality it provides.

Declaration of copyrights, usage limitations, and licenses The *rights* element contains free text, where it is recommended that standard licenses are named and special circumstances like embargoes are mentioned. The *rightsURI* attribute can be used to reference the full license text. By re-using standard URIs (e.g., provided by the creators of the licenses), machines can identify usage conditions.

While the schema allows multiple *rights* elements and thus multiple license URIs, within the VO such use is discouraged. This means that clients that discard information from additional *rights* elements are not considered to be in violation of this specifiation. In practice, resource record authors should place all usage limitations, citation recommendations, etc., directed at VO users in the first *rights* element.

vr:Rights Type Schema Documentation

A statement of usage conditions. This will typically include a license, which should be given as a full string (e.g., Creative Commons Attribution 3.0 International). Further free-text information, e.g., on how to attribute or on embargo periods is allowed.

vr:Rights Type Schema Definition

```
<xs:complexType name="Rights" >
  <xs:simpleContent >
    <xs:extension base="xs:token" >
      <xs:attribute name="rightsURI" type="xs:anyURI" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

vr:Rights Attributes

rightsURI

Type	a URI: *xs:anyURI*
Meaning	A URI identifier for a license
Occurrence	optional
Comment	Where formal licenses are available, this URI can reference the full license text. The IVOA may define standard URIs for a set of recommended licenses, in which case these should be used here.

An example could look like this:

```
<rights rightsURI="http://creativecommons.org/licenses/by/3.0">
  The images from the X survey are copyright 2016, the X project.
  They are published under the creative commons attribution 3.0 unported
  license. If you use this data, please cite doi:10.5072/7273288.

  Images are under embargo for one year after their addition to the
  repository.
</rights>
```

Service capabilities As described in sect. 2.2.7, multiple *capability* elements may appear, each describing a different capability of the service.

vr:Capability Type Schema Documentation

a description of what the service does (in terms of context-specific behavior), and how to use it (in terms of an interface)

vr:Capability Type Schema Definition

```
<xs:complexType name="Capability" >
  <xs:sequence >
    <xs:element name="validationLevel" type="vr:Validation" minOccurs="0"
             maxOccurs="unbounded" />
    <xs:element name="description" type="xs:string" minOccurs="0" />
    <xs:element name="interface" type="vr:Interface" minOccurs="0"
             maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="standardID" type="xs:anyURI" />
</xs:complexType>
```

vr:Capability Attributes

standardID

Type a URI: *xs:anyURI*

Meaning A URI identifier for a standard service.

Occurrence optional

Comment This provides a unique way to refer to a service specification standard, such as a Simple Image Access service. The use of an IVOA identifier here implies that a VOResource description of the standard is registered and accessible.

vr:Capability Metadata Elements

Element *validationLevel*

Type *vr:ValidationLevel* with optional attributes

Meaning A numeric grade describing the quality of the capability description and interface, when applicable, to be used to indicate the confidence an end-user can put in the resource as part of a VO application or research study.

Occurrence optional; multiple occurrences allowed.

Comment See vr:ValidationLevel for an explanation of the allowed levels.

Element *description*

Type string: *xs:string*

Meaning A human-readable description of what this capability provides as part of the over-all service

Occurrence optional

Comment Use of this optional element is especially encouraged when this capability is non-standard and is one of several capabilities listed.

Element *interface*

Type composite: *vr:Interface*

Meaning a description of how to call the service to access this capability

Occurrence optional; multiple occurrences allowed.

Comment Since the Interface type is abstract, one must describe the interface using a subclass of Interface, denoting it via xsi:type.

Comment Multiple occurences can describe different interfaces to the logically same capability, i.e. data or functionality. That is, the inputs accepted and the output provides should be logically the same. For example, a WebBrowser interface given in addition to a WebService interface would simply provide an interactive, human-targeted interface to the underlying WebService interface.

The *capability* element is sufficient for describing a *custom service capability*, i.e., a service that is particular to one provider and does not conform to a specific standard (be it recognized by the IVOA or some other sub-community). However, service standards will often create a *vr:Capability* extension that adds additional metadata that are specific to the behavior of that particular type of service.

More on interfaces The *interface* element is of the complex type *vr:Interface*; its usage, in particular as regards the mandatory use of its *xsi:type* attribute and typical values thereof, is discussed sect. 2.2.7.

Note

The RM defines three pieces of metadata that may be important for several standard query services: *Service.MaxSearchRadius*, *Service.MaxReturnRecords*, and *Service.MaxReturnSize*. These are examples of service-specific metadata that should be encoded as child elements in a type derived from *vr:Capability*.

vr:Interface Type Schema Documentation

A description of a service interface.

Since this type is abstract, one must use an Interface subclass to describe an actual interface.

Additional interface subtypes (beyond WebService and WebBrowser) are defined in the VODataService schema.

vr:Interface Type Schema Definition

```
<xs:complexType name="Interface" abstract="true" >
  <xs:sequence >
    <xs:element name="accessURL" type="vr:AccessURL" minOccurs="1"
             maxOccurs="unbounded" />
    <xs:element name="mirrorURL" type="vr:MirrorURL" minOccurs="0"
             maxOccurs="unbounded" />
    <xs:element name="securityMethod" type="vr:SecurityMethod"
             minOccurs="0"
             maxOccurs="1" />
    <xs:element name="testQueryString" type="xs:token" minOccurs="0"
             maxOccurs="1" />
  </xs:sequence>
  <xs:attribute name="version" type="xs:string" />
  <xs:attribute name="role" type="xs:NMTOKEN" />
</xs:complexType>
```

vr:Interface Attributes

version

Type string: *xs:string*

Meaning The version of a standard interface specification that this interface complies with. Most VO standards indicate the version in the standardID attribute of the capability. For these standards, the version attribute should not be used.

Occurrence optional

role

Type *xs:NMTOKEN*

Meaning A tag name that identifies the role the interface plays in the particular capability. If the value is equal to "std" or begins with "std:", then the interface refers to a standard interface defined by the standard referred to by the capability's standardID attribute.

Occurrence optional

Comment For an interface complying with some registered standard (i.e. has a legal standardID), the role can be matched against interface roles enumerated in standard resource record. The interface descriptions in the standard record can provide default descriptions so that such details need not be repeated here.

vr:Interface Metadata Elements

Element *accessURL*

Type a URI with optional attributes

Meaning The URL (or base URL) that a client uses to access the service. How this URL is to be interpreted and used depends on the specific Interface subclass

Occurrence required; multiple occurrences allowed.

Comment Although the schema allows multiple occurrences of accessURL, multiple accessURLs are deprecated. Each interface should have exactly one access URL. Where an interface has several mirrors, the accessURL should reflect the "primary" (fastest, best-connected, best-maintained) site, the one that non-sophisticated clients will go to.
Additional accessURLs should be put into mirrorURLs. Advanced clients can retrieve the mirrorURLs and empirically determine interfaces closer to their network location.

Element *mirrorURL*

Type a URI with optional attributes

Meaning A (base) URL of a mirror of this interface. As with accessURL, how this URL is to be interpreted and used depends on the specific Interface subclass

Occurrence optional; multiple occurrences allowed.

Comment This is intended exclusively for true mirrors, i.e., interfaces that are functionally identical to the original interface and that are operated by the same publisher. Other arrangements should be represented as separate services linked by mirror-of relationships.

Element *securityMethod*

Type composite: *vr:SecurityMethod*

Meaning The mechanism the client must employ to authenticate to the service.

Occurrence optional

Comment Services not requiring authentication must provide at least one interface definition without a securityMethod defined.

Element *testQueryString*

Type string: *xs:token*

Meaning Test data for exercising the service.

Occurrence optional

Comment This contains data that can be passed to the interface to retrieve a non-empty result. This can be used by validators within test suites.
Exactly how agents should use the data contained in the testQueryString depends on the concrete interface class. For interfaces employing the HTTP GET method, however, this will typically be urlencoded parameters (as for the application/x-www-form-urlencoded media type).

Exactly how a client uses the value of the *accessURL* element depends on the specific type derived from *vr:Interface*. Extension schemas that define non-abstract types derived from *vr:Interface* MUST provide documentation that explains the exact use of the *accessURL*; this documention should follow the documention conventions described in sect. 2.2.

In version 1.0, operators of services with multiple mirrors were encouraged to give multiple *accessURL* elements. This feature was not used in practice, presumably for concerns users might end up on remote mirrors. In version 1.1, we therefore deprecate multiple *accessURL* within an interface. Access URLs of actual mirrors can now be declared using *mirrorURL* elements. As before, neither *accessURL* nor *mirrorURL* must not point to an installation that is outside the administrative control of the service's listed publisher; such a mirror should be described in a separate resource record. To aid in user interfaces for mirror selection, *mirrorURL* admits a *title* attribute:

vr:MirrorURL Type Schema Documentation

A URL of a mirror (i.e., a functionally identical additional service interface) to

vr:MirrorURL Type Schema Definition

```
<xs:complexType name="MirrorURL" >
  <xs:simpleContent >
    <xs:extension base="xs:anyURI" >
      <xs:attribute name="title" type="xs:token" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

vr:MirrorURL Attributes

title

Type	string: *xs:token*
Meaning	A terse, human-readable phrase indicating the function or location of this mirror, e.g., "Primary Backup" or "European Mirror".
Occurrence	optional

Declaring access restrictions The *vr:SecurityMethod* type is defined as a complex type but with empty content:

vr:SecurityMethod Type Schema Documentation

a description of a security mechanism.

This type only allows one to refer to the mechanism via a URI. Derived types would allow for more metadata.

vr:SecurityMethod Type Schema Definition

```
<xs:complexType name="SecurityMethod" >
  <xs:sequence />
  <xs:attribute name="standardID" type="xs:anyURI" />
</xs:complexType>
```

vr:SecurityMethod Attributes

standardID

Type	a URI: *xs:anyURI*
Meaning	A URI identifier for a standard security mechanism.

Occurrence optional

Comment This provides a unique way to refer to a security specification standard. The use of an IVOA identifier here implies that a VOResource description of the standard is registered and accessible.

While this simple element (when the *standardID* attribute is provided) may on its own be sufficient to describe the security mechanism used, it is expected that some future VOResource extension schema will define additional types derived from *vr:SecurityMethod*. If such a sub-type is available, it may be employed at *securityMethod* location within a *vr:Interface*-typed element, in which case it should be invoked via a *xsi:type* attribute to the *securityMethod* element.

Web browser interfaces *vr:WebBrowser* is one of the two *vr:Interface* sub-types defined by the VOResource schema. This type indicates that the service is intended to be accessed interactively by a user through a web browser. The *accessURL*, then, represents the URL of a web page containing one or more forms used to invoke the service.

vr:WebBrowser Type Schema Documentation

A (form-based) interface intended to be accesed interactively by a user via a web browser.

The accessURL represents the URL of the web form itself.

vr:WebBrowser Type Schema Definition

```
<xs:complexType name="WebBrowser" >
  <xs:complexContent >
    <xs:extension base="vr:Interface" >
      <xs:sequence />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

As can be seen in the schema definition above, the *vr:WebBrowser* type does not define any additional interface metadata (though other *vr:Interface* derivations may). Thus, this type is provided purely for its semantic meaning.

vr:WebService is the second *vr:Interface* sub-type available from the VOResource schema:

vr:WebService Type Schema Documentation

A Web Service that is describable by a WSDL document.

The accessURL element gives the Web Service's endpoint URL.

vr:WebService Type Schema Definition

```
<xs:complexType name="WebService" >
  <xs:complexContent >
    <xs:extension base="vr:Interface" >
      <xs:sequence >
        <xs:element name="wsdlURL" type="xs:anyURI" minOccurs="0"
```

```
            maxOccurs="unbounded" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

vr:WebService Extension Metadata Elements

Element *wsdlURL*

Type a URI: *xs:anyURI*

Meaning The location of the WSDL that describes this Web Service. If not provided, the location is assumed to be the accessURL with "?wsdl" appended.

Occurrence optional; multiple occurrences allowed.

Comment Multiple occurrences should represent mirror copies of the same WSDL file.

Legacy web service interfaces The *vr:WebService* interface is one that is described by a Web Service Description Language (Booth and Liu, 2007) document. This is typically realized as one that employs the Simple Object Access Protocol (Box and Ehnebuske et al., 2000); however, like WSDL, this interface type is not restricted to it. With this interface, the *vr:accessURL* must refer to the service endpoint URL.

Note that *vr:WebService* must *not* be used when no WSDL for the service is available. This is true for all modern VO standard services. It is thus unlikely that modern services will use this interface. See sect. 2.2.7 for details.

A Change History

A.1 Changes from PR-20171107

- Editoral changes after DAL and Semantics RFC comments.

A.2 Changes from PR-20170425

- *interface* now only allows zero or one *securityMethod* elements (rather than zero to unlimited in version 1.0). This is to keep discovery patterns simple in the presence of services requiring authentication. While this is a **potentially incompatible change**, no actual records have ever used multiple security methods, and the change has been agreed upon with the stakeholders at the time of the change.

- creator and contact can now have ivo-id attributes (analogous to publisher and contributor); these are intended for when the roles are filled by organizations.

- @updated and @created on resource are now vr:UTCTimestamp rather than xs:datetime. There has always been a textual requirement that these are UTC. This is now enforced by the schema.

- The schema example for an ORCID used a non-standard URI form. It is now deleted, and the text gives an example for an https URI.

A.3 Changes from WD-20170123

- Explanation for mismatch between namespace URI and actual version.
- `testQueryString` has erroneously had multiplicity 0. . . n. It is now defined as 0. . . 1.
- Adopting DataCite 4.0 as upstream, explanation of the relationship between VOResource 1.0 and DataCite terms.
- altIdentifier is now allowed on Contact now (mainly for symmetry with Creator).

A.4 Changes from WD-20161010

- Now recommending the Unified Astronomy Thesaurus instead of the old IAU Thesaurus.
- date/role terms are now flat (rather than deprecated terms being specialisations of their successors).

A.5 Changes from REC-1.03

- Adopting DataCite as another "upstream" in addtion to RM.
- While we still reference RM, it is now no longer informally authoritative for VOResource (we'd have to change RM before we change VOResource otherwise)
- References to the RM terms in the metadata definition dropped (could add support in ivoatex/schemadoc if we want them back).
- Adding altIdentifier elements to creator and resource.
- Adding a version attribute to vr:Resource for compliance with XML Schema Versioning Policies
- The vocabulary of date/@role is now managed as a separate resource; DataCite terms have been added to it, and the old 1.0 terms deprecated.
- The vocabulary of relationshipType is now managed as a separate resource; some DataCite terms have been added to it.
- Service/rights now is free text rather than the tree-term enumeration. In addition, it now has a rightsURI attribute, DataCite-style.
- content/type is now xs:token rather than a special restriction. The vocabulary is now managed as an external resource. Consequently, the vr:Type type vanishes from the XSD.
- content/level is now xs:token rather than a special restriction. The vocabulary is now managed as an external resource. Consequently, the vr:ContentLevel type vanishes from the XSD.

- New testQueryString and mirrorURL children of interface.
- Now discouraging fixing standardID in VOResource extensions. Also, removed indication that capability/@xsi:type should be used for service discovery.
- Now allowing a Z timezone marker in UTCTimestamp, indeed discouraging non-timezone use. This is in line with OAI-PMH use. Therefore, while technically changing the schema such that legacy clients might be confused, we do not expect incompatibilites.
- resource/description and capability/description are now xs:string to enable simple plain text markup.
- Adding language on using capability/@standardID rather than interface/@version to distinguish between different protocol versions in the normal case, removed skynode example.
- Consequently, removing 1.0 default on interface/@version.
- Added advice about the use of creator.name
- Removed SOAP/WSDL example and a bit of the accompanying language.
- Section 3 needed some refactoring since the schema documentation is now generated from the schema document. To accommodate this, some text manually embedded within the schema documentation had to be moved out of the generated material or removed.
- Strongly advising the use of the vr: prefix, removing some duplicated advice regarding prefixes
- Removed example for deriving a SIA capability (this is now in the document repository)
- Ported document source to IvoaTeX.

A.6 Changes from v1.02

- converted to Recommendation

A.7 Changes from v1.01

- *status* attribute is now required.
- added this Change History appendix.

A.8 Changes from v1.0

- dropped *PaddedString*, *PaddedURI* and replaced with *xs:token*.
- made *Validation*'s *validatedBy* attribute required.
- reference citation correction for SOAP, WSDL

References

Arviset, C., Gaudet, S. and the IVOA Technical Coordination Group (2010), 'IVOA architecture', IVOA Note.
http://www.ivoa.net/documents/Notes/IVOAArchitecture

Benson, K., Plante, R., Auden, E., Graham, M., Greene, G., Hill, M., Linde, T., Morris, D., O'Mullane, W., Rixon, G., Stébé, A. and Andrews, K. (2009), 'IVOA Registry Interfaces Version 1.0', IVOA Recommendation 04 November 2009, arXiv:1110.0513.
http://adsabs.harvard.edu/abs/2009ivoa.spec.1104B

Berners-Lee, T., Fielding, R. and Masinter, L. (2005), 'Uniform Resource Identifier (URI): Generic syntax', RFC 3986.
http://www.ietf.org/rfc/rfc3986.txt

Booth, D. and Liu, C. K. (2007), 'Web services description language (wsdl) version 2.0 part 0: Primer', W3C Recommendation.
http://www.w3.org/TR/wsdl20-primer

Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Nielsen, H. F., Thatte, S. and Winer, D. (2000), 'Simple object access protocol (soap) 1.1'.
http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

Bradner, S. (1997), 'Key words for use in RFCs to indicate requirement levels', RFC 2119.
http://www.ietf.org/rfc/rfc2119.txt

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E. and Yergeau, F. (2008), 'Extensible markup language (XML) 1.0 (fifth edition)', W3C Recommendation.
http://www.w3.org/TR/REC-xml/

DataCite Metadata Working Group (2016), 'DataCite metadata schema – documentationfor the publication and citation of research data version 4.0', DataCite publication.
https://schema.datacite.org/meta/kernel-4.0/doc/DataCite-MetadataKernel_v4.0.pdf

Demleitner, M., Dowler, P., Plante, R., Rixon, G. and Taylor, M. (2012), 'TAPRegExt: a VOResource Schema Extension for Describing TAP Services Version 1.0', IVOA Recommendation 27 August 2012, arXiv:1402.4742.
http://adsabs.harvard.edu/abs/2012ivoa.spec.0827D

Demleitner, M., Harrison, P., Molinaro, M., Greene, G., Dower, T. and Perdikeas, M. (2014), 'IVOA Registry Relational Schema Version 1.0', IVOA Recommendation 08 December 2014, arXiv:1510.02275.
http://adsabs.harvard.edu/abs/2014ivoa.spec.1208D

Demleitner, M., Plante, R., Linde, T., Williams, R. and Noddle, K. (2016), 'IVOA Identifiers Version 2.0', IVOA Recommendation 23 May 2016, arXiv:1605.07501.
http://adsabs.harvard.edu/abs/2016ivoa.spec.0523D

Dowler, P., Bonnarel, F., Michel, L. and Demleitner, M. (2015), ‘IVOA DataLink Version 1.0’, IVOA Recommendation 17 June 2015, arXiv:1509.06152.
http://adsabs.harvard.edu/abs/2015ivoa.spec.0617D

Graham, M., Rixon, G. and Grid andWeb Services Working Group (2011), ‘IVOA Support Interfaces Version 1.0’, IVOA Recommendation 31 May 2011, arXiv:1110.5825.
http://adsabs.harvard.edu/abs/2011ivoa.spec.0531G

Hanisch, R., IVOA Resource Registry Working Group and NVO Metadata Working Group (2007), ‘Resource Metadata for the Virtual Observatory Version 1.12’, IVOA Recommendation 02 March 2007, arXiv:1110.0514.
http://adsabs.harvard.edu/abs/2007ivoa.spec.0302H

Harrison, P., Burke, D., Plante, R., Rixon, G., Morris, D. and IVOA Registry Working Group (2012), ‘StandardsRegExt: a VOResource Schema Extension for Describing IVOA Standards Version 1.0’, IVOA Recommendation 08 May 2012, arXiv:1402.4745.
http://adsabs.harvard.edu/abs/2012ivoa.spec.0508H

Harrison, P., Demleitner, M., Major, B. and Dowler, P. (2018), ‘XML Schema Versioning Policies Version 1.0’, IVOA Endorsed Note 29 May 2018.
http://adsabs.harvard.edu/abs/2018ivoa.spec.0529H

International Organization for Standardization (2004), ‘Data elements and interchange formats – information interchange – representation of dates and times’.
http://www.iso.org/iso/catalogue_detail?csnumber=40874

Lagoze, C., de Sompel, H. V., Nelson, M. and Warner, S. (2002), ‘The open archives initiative protocol for metadata harvesting, version 2.0’.
http://www.openarchives.org/OAI/openarchivesprotocol.html

Plante, R., Delago, J., Harrison, P., Tody, D. and IVOA Registry Working Group (2013), ‘Describing Simple Data Access Services Version 1.0’, IVOA Recommendation 25 November 2013, arXiv:1402.4747.
http://adsabs.harvard.edu/abs/2013ivoa.spec.1125P

Plante, R., Stébé, A., Benson, K., Dowler, P., Graham, M., Greene, G., Harrison, P., Lemson, G., Linde, T. and Rixon, G. (2010), ‘VODataService: a VOResource Schema Extension for Describing Collections, Services Version 1.1’, IVOA Recommendation 02 December 2010, arXiv:1110.0516.
http://adsabs.harvard.edu/abs/2010ivoa.spec.1202P

Thompson, H. S., Beech, D., Maloney, M. and Mendelsohn, N. (2004), ‘XML schema part 1: Structures second edition’, W3C Recommendation.
http://www.w3.org/TR/xmlschema-1/