



StandardsRegExt: a VOResource Schema Extension for Describing IVOA Standards Version 1.0

IVOA Working Draft
19 May 2010

This version:

<http://www.ivoa.net/Documents/StandardsRegExt/20100519/>

Latest version:

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

Previous versions

n/a

Authors:

[Paul Harrison](#), Editor

[Douglas Burke](#)

[Ray Plante](#)

[Guy Rixon](#)

[Dave Morris](#)

and the IVOA Registry Working Group.

Abstract

This document describes an XML encoding standard for metadata about IVOA standards themselves, referred to as StandardsRegExt. It is intended to allow for the discovery of a standard via an IVOA identifier that refers to the standard. It also allows one to define concepts that are defined by the standard which can themselves be referred to via an IVOA identifier (augmented with a URL fragment identifier). Finally, it can also provide a machine interpretable description of a standard service interface. We describe the general model for the schema and explain its intended use by interoperable registries for discovering resources.

Status of this document

This is an IVOA Working Draft for review by IVOA members and other interested parties. It is a draft document and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use IVOA Working Drafts as reference materials or to cite them as other than "works in progress".

Pre-1.0 versions of this document were known as [VOStandard](#).

Comments on this document for consideration in the next version are welcome. They can be sent to registry@ivoa.net, a mailing list with a [public archive](#) or on the [StandardsRegExt twiki discussion page](#). When this document becomes a Proposed Recommendation, an official public Request For Comment period will begin.

A list of [current IVOA Recommendations and other technical documents](#) can be found at <http://www.ivoa.net/Documents/>.

Editor's Note:

Parts that the editor considers should be removed from the document are marked in ~~red with a strike through line~~, and sections that need more work are indicated in green.

Acknowledgements

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 \(OPTICON\)](#).

This document contains text lifted verbatim, with small changes, and with substantial changes from the previously published

VODataService specification [VDS] (e.g. sections 2.0). This has been done without specific attribution as a means for providing consistency across similar documents. We acknowledge the authors of that document for this text.

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, RFC 2119 [RFC 2119].

The **Virtual Observatory (VO)** is 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.

XML document **validation** is a software process that checks that an XML document is not only well-formed XML but also conforms to the syntax rules defined by the applicable schema. Typically, when the schema is defined by one or more XML Schema [Schema] documents (see [next section](#)), validation refers to checking for conformance to the syntax described in those Schema documents. This document describes additional syntax constraints that cannot be enforced solely by the rules of XML Schema; thus, in this document, use of the term validation includes the extra checks that goes beyond common Schema-aware parsers which ensure conformance with this document.

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 the World Wide Web Consortium (W3C) Recommendation, [XML 1.0 \[XML\]](#). 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 [Schema].

This document defines the StandardsRegExt schema using XML Schema. The full Schema document is listed in [Appendix A](#). Parts of the schema appear within the main sections of this document; however, documentation nodes have been left out for the sake of brevity.

Reference 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). Reference to specific elements and types defined in the StandardsRegExt schema include the namespaces prefix, *vstd*, as in *vstd:ServiceStandard* (a type defined in the StandardsRegExt schema). Use of the *vstd* prefix in compliant instance documents is strongly recommended, particularly in the applications that involve IVOA Registries (see [RI], section 3.1.2). Elsewhere, the use is not required.

Contents

- [Abstract](#)
- [Preface](#)
 - [Status of this document](#)
 - [Acknowledgements](#)
 - [Conformance-related definitions](#)
 - [Syntax Notation Using XML Schema](#)
- [1. Introduction](#)
- [2. The StandardsRegExt Data Model](#)
 - [2.1. The Schema Namespace and Location](#)
 - [2.2. Summary of Metadata Concepts](#)
 - [2.3. Defining Enumerations of Identifiers](#)
- [3. The StandardsRegExt Metadata](#)
 - [3.1. Resource Type Extensions](#)
 - [3.2. Defining Keys: StandardKey and StandardKeyURI](#)
- [Appendix A: The complete StandardsRegExt Schema](#)
- [Appendix B: Change History](#)
- [References](#)

1. Introduction

An important goal of the IVOA is to publish standards for services which can interoperate to create a Virtual Observatory (VO). Central to the coordination of these services is the concept of a registry ([RI]) where resources can be described and thus discovered by users and applications in the VO. The standard [Resource Metadata for the Virtual Observatory \[Hanisch et al. 2004\]](#) (hereafter referred to as **RM**) defines metadata terms for services and other discoverable resources. A specific XML encoding of these resources is described in the IVOA standard [VOResource: an XML Encoding Schema for Resource Metadata \[Plante et al. 2006\]](#) (hereafter referred to as **VOResource**). In this schema, support for a standard service protocol is described as a service's *capability*; the associated metadata is contained within the service resource description's *<capability> element*. The specific standard protocol supported is uniquely identified via an attribute of the *<capability> element* called *standardID* whose value is a URI. The [VOResource standard](#) does not place a formal validation requirement on the *standardID* other than it be a legal URI; however, it was intended that IVOA-endorsed standards would be represented via an IVOA identifier. As per the IVOA Identifier standard [ID], an IVOA identifier must be registered as a resource in an IVOA-compliant registry.

This document defines a VOResource extension schema call **StandardsRegExt** which allows one to describe a standard and register it with an IVOA registry. By doing so, a unique IVOA identifier becomes "attached" to the standard which can be referred to in other resource descriptions, namely for services that support the standard. Not only does this aid in the unambiguous discovery of compliant service instances but also in validating their descriptions and support for the standard. Another benefit of

associating an IVOA identifier with a standard is that allows registry users who discover services that conform to particular standard to also discover the document that describes that standard.

StandardsRegExt has two other purposes. First, it allows a service protocol description to communicate specifics about the standard input parameters and output formats specified by the standard. Such a machine-readable description of the interface can assist intelligent portals and applications to build GUI interfaces to standard services and manage workflows built around them. Second, it allows for the definition of small controlled sets of standardized names (referred to as *keys* in this document) which might be used to identify, for example, specific features of a standard protocol (such as supported data transport protocols). By virtue of being defined within the context of a VOResource description, one can refer to the key using a globally unique URI by adding the key name as a URI fragment [[URI](#)] onto IVOA identifier associated with the descriptions.

It is envisaged that StandardsRegExt instances that describe standards endorsed or otherwise in development by the IVOA will be published in the IVOA Registry of Registries [[RofR](#)] using the authority identifier [[ID](#)], [ivoa.net](#). However, other standards, be they ad hoc or endorsed by another body, may be published in any compliant registry.

2. The StandardsRegExt Data Model

The StandardsRegExt extension in general enables the description of three types of resources:

- a generic standard (specified by an external document)
- a standard specifically defining a service protocol
- a set of related, standardized names called *keys*.

Here's an example of defining a controlled list of computer languages that might be referred to in other descriptions of applications. Should other languages become prevalent, this resource description could be updated to add the new names.

An example of defining a list of programming languages

```
<ri:Resource xsi:type="vstd:StandardKeyEnumeration" created="2001-12-31T12:00:00"
  updated="2001-12-31T12:00:00" status="active">
  <title>application languages</title>
  <identifier>ivo://ivoa.net/std/application/languages</identifier>
  <curator>
    <publisher>IVOA</publisher>
    <creator>
      <name>IVOA</name>
      <logo>http://www.ivoa.net/icons/ivoa_logo_small.jpg</logo>
    </creator>
    <date role="representative">2006-07-17</date>
    <version>1.0</version>
    <contact>
      <name>IVOA Grid and Web Services WG</name>
      <email>grid@ivoa.net</email>
    </contact>
  </curator>
  <content>
    <subject>IVOA Standard: registry</subject>
    <description>
      This resource defines keys for commonly used computer languages.
    </description>
    <referenceURL>http://www.ivoa.net/twiki/bin/view/IVOA/IvoaResReg</referenceURL>
  </content>
  <key>
    <name>C</name>
    <description>The C programming language</description>
  </key>
  <key>
    <name>CPP</name>
    <description>The C++ programming language</description>
  </key>
  <key>
    <name>CSharp</name>
    <description>The C# programming language</description>
  </key>
  <key>
    <name>FORTRAN</name>
    <description>The FORTRAN programming language</description>
  </key>
  <key>
    <name>Java</name>
    <description>The Java programming language</description>
  </key>
  <key>
    <name>Perl</name>
    <description>The Perl programming language</description>
  </key>
  <key>
    <name>Python</name>
    <description>The Python programming language</description>
  </key>
</ri:Resource>
```

2.1. The Schema Namespace and Location

The namespace associated with StandardsRegExt extensions is "http://www.ivoa.net/xml/StandardsRegExt/v1.0". Just like the

namespace URI for the VOResource schema, the StandardsRegExt namespace URI can be interpreted as a URL. Resolving it will return the XML Schema document (given in [Appendix A](#)) that defines the StandardsRegExt schema.

Authors of VOResource instance documents may choose to provide a location for the VOResource XML Schema document and its extensions using the `xsi:schemaLocation` attribute. While the choice of the location value is the choice of the author, this specification recommends using the StandardsRegExt namespace URI as its location URL (as illustrated in the example above), as in,

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

Note:

The IVOA Registry Interface standard [RI] actually *requires* that the VOResource records it shares with other registries provide location URLs via `xsi:schemaLocation` for the VOResource schema and all legal extension schemas that are used in the records. This rule would apply to the StandardsRegExt schema.

The prefix, `vstd`, is used by convention as the prefix defined for the StandardsRegExt schema; however, instance documents may use any prefix of the author's choosing. In applications where common use of prefixes is recommended (such as with the Registry Interface specification [RI]), use of the `vstd` prefix is recommended. Note also that in this document, the `vr` prefix is used to label, as shorthand, a type or element name that is defined in the VOResource schema, as in `vr:Resource`.

As recommend by the VOResource standard [VR], the StandardsRegExt schema sets `elementFormDefault="unqualified"`. This means that it is not necessary to qualify element names defined in this schema with a namespace prefix (as there are no global elements defined). The only place it is usually needed is as a qualifier to a StandardsRegExt type name given as the value of an `xsi:type` attribute.

2.2. Summary of Metadata Concepts

The StandardsRegExt extension defines three new types of resources. Two are specifically for independently documented standards:

vstd:Standard

This resource describes a general standard (e.g. data model, schema, protocol, etc.). The most important piece of metadata associated with this resource is the `<referenceURL>` (from the core VOResource schema) which should point to the human-readable specification document that defines the standard. It also allows one to provide the recommended version of the standard to use.

vstd:ServiceStandard

This resource type, which extends from `vstd:Standard`, is specifically for describing a standard service protocol (e.g. Simple Cone Search). It differs from `vstd:Standard` in that it also allows one to describe specific constraints on the service interface via its `<interface>` element.

vstd:StandardKeyEnumeration

This resource type allows for the description of a related set controlled names (referred to as *keys*) and their meanings. While keys can be defined as part of a `vstd:Standard` or `vstd:ServiceStandard` resource, the `vstd:StandardKeyEnumeration` allows a set of key definitions stand as a resource on its own, regardless of whether it is part of a documented standard or not.

2.3. Defining Enumerations of Identifiers

A common practice when defining metadata to restrict certain string values to a controlled sets of defined names, each with a well-defined meaning. With XML Schema, the controlled set can be enforced by a validating parser (using the `xsd:enumeration` construct [Schema]). One disadvantage of locking in the vocabulary in an XML Schema document is that growing list of allowed names requires a revision of the XML Schema document, which can be a disruptive change. To avoid this, it is the practice VOResource and its extensions to avoid "hard-coded" enumerations in the XML Schema document for sets of defined values that will likely change over time.

The StandardsRegExt schema provides an alternative to XML Schema-based definitions of controlled names. Instead, a controlled list of names, called *standard keys*, can be defined as part of any of the three StandardsRegExt resource types. Updating a resource description is much less disruptive than a Schema document, and as a resource available via an IVOA-compliant registry, it is still possible for a (non-Schema-based) application to validate the use of the vocabulary.

The StandardsRegExt specification also defines a mapping from a key name to a URI. This allows these keys--and their underlying meaning--to be referenced in a globally unique way in a variety of contexts, not restricted to XML.

A key is defined using the `vstd:StandardKey` type which consists simply of a name and a description. The key is mapped to a URI by attaching the name as the "fragment"--i.e., appending after a pound sign, `#`--to the IVOA identifier for the resource description that defines the key:

```
ivoa-identifier#key-name
```

where *ivoa-identifier* is the resource's IVOA identifier and *key-name* is the name of a key defined in that resource.

For example with a key named `case-insensitive` defined within a resource description with an IVOA identifier given by `<identifier>` `ivo://ivoa.net/std/QueryProtocol` `<identifier>`, the URI identifying this key would be:

```
ivo://ivoa.net/std/QueryProtocol#case-insensitive
```

This form of defining multiple keys, each with its own mapping to a URI, all in one resource has several advantages:

- The enumerations are naturally grouped under a single registry resource, and so can be retrieved with one registry query and need no further metadata to assert the association.
- The "Dublin core" metadata that is associated with a resource need only be entered once for the whole enumeration, rather than for each member of the enumeration - this saves both curation effort and space in the registry.
- If it is necessary to expand the list of controlled names (or shrink it), it is simple and fairly undistruptive to update the VOResource record.

Note:

When these enumerations are presented to a user in a GUI it is expected that the only the "fragment" part that distinguishes the various members of the enumeration will be used as a choice value, as the full IVO ID is not usually particularly "user-friendly".

Some applications may wish to publish additional metadata associated with a key definition through further extension of VOResource metadata. This can be done by deriving a new key metadatum type derived by extension from the `vstd:StandardKey`.

3. The StandardsRegExt Metadata

3.1. Resource Type Extensions

This specification defined three new resource types. As is spelled out in the VOResource specification, a resource description indicates that it refers to one of these types of resources by setting the `xs:type` attribute to the namespace-qualified type name. Doing so implies that the semantic meaning of that type applies to the resource.

3.1.1. Standard

The `vstd:Standard` resource type describes any general standard specification. This typically refers to an IVOA standard but is not limited to such. Generally, the `vstd:Standard` type is intended for standards *other than* standard protocols (which should use the `vstd:ServiceStandard` type instead). It extends the generic `vr:Resource` type as follows.

vstd:Standard Type Schema Definition

```
<xs:complexType name="Standard" >
  <xs:complexContent >
    <xs:extension base="vr:Resource" >
      <xs:sequence >
        <xs:element name="endorsedVersion" type="vstd:EndorsedVersion"
          maxOccurs="unbounded" />
        <xs:element name="deprecated" type="xs:token" minOccurs="0" />
        <xs:element name="key" type="vstd:StandardKey" minOccurs="0"
          maxOccurs="unbounded" />
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

As one of the purposes of this resource type is to enable users to discover the documentation that defines the standard that the resource describes, the `<referenceURL>` should point either to the standard's specification document or to summary information about the standard that can lead one to the specification document.

The `vstd:Standard` resource type adds two metadata terms to the core set:

vstd:Standard Extension Metadata Elements	
Element	Definition
endorsedVersion	<p><i>Value type:</i> a string with optional attributes</p> <p><i>Semantic Meaning:</i> the version of the standard that is recommended for use.</p> <p><i>Occurrences:</i> required; multiple occurrences allowed.</p> <p><i>Comments:</i> More than one version can be listed, indicating that any of these versions are recommended for use.</p>
deprecated	<p><i>Value type:</i> string: <code>xs:token</code></p> <p><i>Semantic Meaning:</i> when present, this element indicates that all versions of the standard are considered deprecated by the publisher. The value is a human-readable explanation for the designation.</p> <p><i>Occurrences:</i> optional</p> <p><i>Comments:</i> The explanation should indicate if another standard should be preferred.</p>
key	<p><i>Value type:</i> composite: <code>vstd:StandardKey</code></p> <p><i>Semantic Meaning:</i> a defined key associated with this standard.</p> <p><i>Occurrences:</i> optional; multiple occurrences allowed.</p>

The child `<key>` elements define terms with special meaning to the standard; see [section 3.2](#).

The purpose of the required `<endorsedStandard>` element is to point potential users of the standard to the version that is most preferred by the standard's publisher. If multiple versions are relevant or in use, multiple elements may be given; in this case, the `use` attribute can further help steer the users to the preferred version.

vstd:EndorsedVersion Type Schema Definition

```
<xs:complexType name="EndorsedVersion" >
  <xs:simpleContent >
    <xs:extension base="xs:string" >
      <xs:attribute name="status" default="n/a" >
        <xs:simpleType >
          <xs:restriction base="xs:string" >
            <xs:enumeration value="rec" />
            <xs:enumeration value="prop" />
            <xs:enumeration value="wd" />
            <xs:enumeration value="n/a" />
          </restriction>
        </simpleType>
      </attribute>
      <xs:attribute name="use" >
        <xs:simpleType >
          <xs:restriction base="xs:string" >
            <xs:enumeration value="preferred" />
            <xs:enumeration value="deprecated" />
          </restriction>
        </simpleType>
      </attribute>
    </extension>
  </simpleContent>
</complexType>
```

vstd:EndorsedVersion Attributes

Attribute	Definition								
status	<p><i>Value type:</i> string with controlled vocabulary</p> <p><i>Semantic Meaning:</i> the IVOA status level of this version of the standard.</p> <p><i>Occurrences:</i> optional</p> <p><i>Allowed Values:</i></p> <table> <tr> <td>rec</td> <td>an IVOA Recommendation</td> </tr> <tr> <td>prop</td> <td>an IVOA Proposed Recommendation</td> </tr> <tr> <td>wd</td> <td>an IVOA Working Draft</td> </tr> <tr> <td>n/a</td> <td>not an IVOA standard or protostandard at this time.</td> </tr> </table> <p><i>Default Value:</i> n/a</p>	rec	an IVOA Recommendation	prop	an IVOA Proposed Recommendation	wd	an IVOA Working Draft	n/a	not an IVOA standard or protostandard at this time.
rec	an IVOA Recommendation								
prop	an IVOA Proposed Recommendation								
wd	an IVOA Working Draft								
n/a	not an IVOA standard or protostandard at this time.								
use	<p><i>Value type:</i> string with controlled vocabulary</p> <p><i>Semantic Meaning:</i> A designation of preference for the version compared to other versions in use.</p> <p><i>Occurrences:</i> optional</p> <p><i>Allowed Values:</i></p> <table> <tr> <td>preferred</td> <td>the most preferred version.</td> </tr> <tr> <td>deprecated</td> <td>a versions whose use is now discouraged because a newer version is preferred.</td> </tr> </table>	preferred	the most preferred version.	deprecated	a versions whose use is now discouraged because a newer version is preferred.				
preferred	the most preferred version.								
deprecated	a versions whose use is now discouraged because a newer version is preferred.								

When all versions of the standard are considered deprecated by the resource publisher, the `<deprecated>` child element should appear. The explanation given as a value should mention the standard that the current standard is deprecated by when relevant.

Note:

An example where the `<deprecated>` element might be used in the VO is in the case of the SkyNode standard [[SkyNode](#)]. As of this writing, there are many instances of SkyNode services available in the VO, and where they are used, version 1.01 is endorsed; however, the IVOA is deprecating this standard in favor of the Table Access Protocol.

An example of a Standard resource that summarizes this specification

```
<ri:Resource xsi:type="vstd:Standard" created="2001-12-31T12:00:00"
  updated="2001-12-31T12:00:00" status="active">
  <title>StandardsRegExt: a VOResource Schema Extension for Describing IVOA Standards</title>
  <identifier>ivo://ivoa.net/std/StandardsRegExt</identifier>
  <curator>
    <publisher>IVOA</publisher>
    <creator>
      <name>IVOA</name>
      <logo>http://www.ivoa.net/icons/ivoa_logo_small.jpg</logo>
    </creator>
    <date role="representative">2006-07-17</date>
    <version>1.0</version>
    <contact>
      <name>IVOA Registry WG</name>
      <email>registry@ivoa.net</email>
    </contact>
  </curator>
```

```

<content>
  <subject>standard language identifiers</subject>
  <subject>virtual observatory</subject>
  <description>
    This document describes an XML encoding standard for metadata about
    IVOA standards themselves, referred to as StandardsRegExt. We describe the
    general model for the schema and explain how it may be included in
    other schema as a methodology of avoiding XML enumerations. This
    schema is primarily intended to support interoperable registries used
    for discovering resources.
  </description>
  <referenceURL>http://www.ivoa.net/Documents/latest/StandardsRegExt.html</referenceURL>
</content>
<endorsedVersion status="wd">1.0</endorsedVersion>
</ri:Resource>

```

3.1.2. ServiceStandard

The `vstd:ServiceStandard` resource type extends `vstd:Standard` to describe more specifically a standard protocol. It adds an `<interface>` element to allow the interface defined by the standard to be described in a machine-readable way. Its type is defined to be `vr:Interface`, which is defined in the VOResource schema [VR].

vstd:ServiceStandard Type Schema Definition

```

<xs:complexType name="ServiceStandard" >
  <xs:complexContent >
    <xs:extension base="vstd:Standard" >
      <xs:sequence >
        <xs:element name="interface" type="vr:Interface" minOccurs="0"
          maxOccurs="unbounded" />
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

vstd:ServiceStandard Extension Metadata Elements

Element	Definition
interface	<p><i>Value type:</i> composite: <code>vr:Interface</code></p> <p><i>Semantic Meaning:</i> an abstract description of one of the interfaces defined by this service standard.</p> <p><i>Occurrences:</i> optional; multiple occurrences allowed.</p> <p><i>Comments:</i> This element can provide details about the interface that apply to all implementations. Each interface element should specify a <code>role</code> attribute with a value starting with "std:", or, if there is only one standard ce, <code>role</code> is set to "std".</p>

Even though the `vr:Interface` type requires an `<accessURL>` child element, the `<interface>` element in a `vstd:ServiceStandard` is intended to describe a service in the abstract--i.e. without reference to a particular installation of the service. Consequently, the `accessURL` should contain a bogus URL; applications should not expect it to be resolvable.

An applications can, in principle, get a complete machine-readable description of a particular instance of a standard service (to, say, create a GUI for that service on-the-fly) by combining the general description in the `vstd:ServiceStandard` record with the service resource description for the specific instance. The intended process for building that description is as follows:

1. The application obtains a VOResource resource record for the service instance (e.g. from a registry).
2. The application extracts the `standardID` attribute for the desired service capability.
3. The `standardID` is resolved (via a registry) to a `vstd:ServiceStandard` record for the service. This description would capture the required and optional (but standard) components of the service interface.
4. The specific instance's interface description is merged into the standard one. The service's support of optional components as well as its allowed customizations would override the generic description from the `vstd:ServiceStandard` record.

Note:

A number of IVOA standard services (e.g. [SIA]) are registered using the `vs:ParamHTTP` interface type to describe its interface. This interface type allows one to list input parameters accepted by the service. Each parameter can be marked as *required* or *optional*. A `vstd:ServiceStandard` record for the SIA protocol can list the required and optional parameters defined by the SIA specification. A resource record for a real SIA service, then, need only list the optional parameters that it supports plus any custom parameters. Between the two records, an application can, in principle, build a GUI to the SIA service without knowing anything special about the SIA standard protocol.

3.1.3. StandardKeyEnumeration

The `vstd:StandardKeyEnumeration` resource type is available for collecting definitions of related, standard keys. Each key defined within this resource can then be referred to by a unique IVOA Identifier URI (see [section 3.2](#)). To support this, the `vstd:StandardKeyEnumeration` resource simply adds the `<keys>` element to the standard core metadata.

vstd:StandardKeyEnumeration Type Schema Definition

```

<xs:complexType name="StandardKeyEnumeration" >
  <xs:complexContent >
    <xs:extension base="vr:Resource" >
      <xs:sequence >
        <xs:element name="key" type="vstd:StandardKey" maxOccurs="unbounded"
          minOccurs="1" />
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

vstd:StandardKeyEnumeration Extension Metadata Elements

Element	Definition
key	<p><i>Value type:</i> composite: vstd:StandardKey</p> <p><i>Semantic Meaning:</i> the name and definition of a key—a named concept, feature, or property.</p> <p><i>Occurrences:</i> required; multiple occurrences allowed.</p>

The contents of the <key> element is described in the next section.

3.2. Defining Keys: StandardKey and StandardKeyURI

The `vstd:StandardKey` type provides the means to define keys (as defined in [section 2.3](#)) within a VOResource record.

vstd:StandardKey Type Schema Definition

```

<xs:complexType name="StandardKey" >
  <xs:sequence >
    <xs:element name="name" type="vstd:fragment" />
    <xs:element name="description" type="xs:token" />
  </sequence>
</complexType>

<xs:simpleType name="fragment" >
  <xs:restriction base="xs:string" >
    <xs:pattern value="[A-Za-z0-9;\/?:@&=+\$, \_ \! \- \* \(\ \)]|[A-Za-f0-9]{2}+" />
  </restriction>
</simpleType>

```

vstd:StandardKey Metadata Elements

Element	Definition
name	<p><i>Value type:</i> string with a form restricted to a legal URI fragment [URI].</p> <p><i>Semantic Meaning:</i> The property identifier which would appear as the fragment (string after the pound sign, #) in an IVOA identifier.</p> <p><i>Occurrences:</i> required</p>
description	<p><i>Value type:</i> string: <code>xs:token</code></p> <p><i>Semantic Meaning:</i> A human-readable definition of this property.</p> <p><i>Occurrences:</i> required</p>

Defining a key via a <key> element within a VOResource record implies the definition of a unique URI formed according to the syntax described in [section 2.3](#) that represents the semantics given by the value of the <description> child element. Because the URI must be globally unique, the key name (given by the <name> child element) must be unique within the VOResource record.

Though it is not needed by StandardsRegExt resource records, the StandardsRegExt schema further defines a convenience type, `vstd:StandardKeyURI`, which defines the legal pattern for a full standard key identifier (as defined in [section 2.3](#)). Applications that wish to use XML Schema to validate the form of a key URI may import the StandardsRegExt schema and use this type.

Note:

It is worth noting that just using or otherwise referencing a standard key URI in an application does not require importing the StandardsRegExt nor need there be any reference to the StandardsRegExt namespace. The role of the StandardsRegExt schema is simply to provide a way of documenting the definitions in a VOResource record. Thus, an application may dereference the URI for display or user help purposes; however, dereferencing is not necessary to use the URI.

Appendix A: The complete StandardsRegExt Schema

Note that this schema can be found on-line at <http://www.ivoa.net/xml/StandardsRegExt/v0.2> (i.e. the target namespace can also be used as a URL for the schema **not yet uploaded for this draft**) This location should represent the definitive source, the schema is only copied below for completeness of this document.

The Complete StandardsRegExt Schema

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.ivoa.net/xml/StandardsRegExt/v0.4"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:vr="http://www.ivoa.net/xml/VOResource/v1.0"
  xmlns:vstd="http://www.ivoa.net/xml/StandardsRegExt/v0.4"
  xmlns:vm="http://www.ivoa.net/xml/VOMetadata/v0.1"
  elementFormDefault="unqualified" attributeFormDefault="unqualified"
  version="0.4rc0" >

  <xs:annotation>
    <xs:appinfo>
      <vm:schemaName>StandardsRegExt</vm:schemaName>
      <vm:schemaPrefix>xs</vm:schemaPrefix>
      <vm:targetPrefix>vstd</vm:targetPrefix>
    </xs:appinfo>
    <xs:documentation>
      This is a core schema for describing IVOA Standards themselves
    </xs:documentation>
  </xs:annotation>

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

  <xs:complexType name="Standard">
    <xs:annotation>
      <xs:documentation>
        a description of a standard specification.
      </xs:documentation>
      <xs:documentation>
        This typically refers to an IVOA standard but is not
        limited to such.
      </xs:documentation>
    </xs:annotation>

    <xs:complexContent base="vr:Resource">
      <xs:sequence>

        <xs:element name="endorsedVersion" type="vstd:EndorsedVersion"
          maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>
              the version of the standard that is recommended for use.
            </xs:documentation>
            <xs:documentation>
              More than one version can be listed, indicating
              that any of these versions are recognized as
              acceptable for use.
            </xs:documentation>
          </xs:annotation>
        </xs:element>

        <xs:element name="deprecated" type="xs:token" minOccurs="0">
          <xs:annotation>
            <xs:documentation>
              when present, this element indicates that all
              versions of the standard are considered
              deprecated by the publisher. The value is a
              human-readable explanation for the designation.
            </xs:documentation>
            <xs:documentation>
              The explanation should indicate if another
              standard should be preferred.
            </xs:documentation>
          </xs:annotation>
        </xs:element>

        <xs:element name="key" type="vstd:StandardKey"
          minOccurs="0" maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>
              a defined key associated with this standard.
            </xs:documentation>
          </xs:annotation>
        </xs:element>

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

  <xs:complexType name="EndorsedVersion">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="status" default="n/a">
          <xs:annotation>
            <xs:documentation>
              the IVOA status level of this verison of the standard.
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

```

```

    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="rec">
          <xs:annotation>
            <xs:documentation>
              an IVOA Recommendation
            </xs:documentation>
          </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="prop">
          <xs:annotation>
            <xs:documentation>
              an IVOA Proposed Recommendation
            </xs:documentation>
          </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="wd">
          <xs:annotation>
            <xs:documentation>
              an IVOA Working Draft
            </xs:documentation>
          </xs:annotation>
        </xs:enumeration>
        <xs:enumeration value="n/a">
          <xs:annotation>
            <xs:documentation>
              not an IVOA standard or protostandard at
              this time.
            </xs:documentation>
          </xs:annotation>
        </xs:enumeration>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>

  <xs:attribute name="use">
    <xs:annotation>
      <xs:documentation>
        A designation of preference for the version compared
        to other versions in use.
      </xs:documentation>
    </xs:annotation>

    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="preferred">
          <xs:annotation>
            <xs:documentation>
              the most preferred version.
            </xs:documentation>
          <xs:documentation>
            Only one version should have this designation.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
        <xs:enumeration value="deprecated">
          <xs:annotation>
            <xs:documentation>
              a versions whose use is now discouraged
              because a newer version is preferred.
            </xs:documentation>
          </xs:annotation>
        </xs:enumeration>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>

</xs:extension>
</xs:simpleContent>
</xs:complexType>

<!-- I am not sure if this can really be used with the interface defined as below, as the that element has an accessURL, which can really only refer to a pa
Paul Harrison -->
<!-- See the 4th comment to the interface element in the documentation below
Ray Plante -->
  <xs:complexType name="ServiceStandard">
    <xs:annotation>
      <xs:documentation>
        a description of a standard service protocol.
      </xs:documentation>
      <xs:documentation>
        This typically refers to an IVOA standard but is not
        limited to such.
      </xs:documentation>
    </xs:annotation>

    <xs:complexContent>
      <xs:extension base="vstd:Standard">
        <xs:sequence>
          <xs:element name="interface" type="vr:Interface"
            minOccurs="0" maxOccurs="unbounded">

```

```

    <xs:annotation>
      <xs:documentation>
        an abstract description of one of the interfaces defined
        by this service standard.
      </xs:documentation>
    </xs:annotation>
    <xs:documentation>
      This element can provide details about the interface
      that apply to all implementations. Each interface
      element should specify a role with a value starting
      with "std:" or, if there is only one standard interface,
      is equal to "std".
    </xs:documentation>
    <xs:documentation>
      Applications that, for example, wish to build a GUI
      to the service on-the-fly would first access this generic
      description. Site-specific variations, such
      as supported optional arguments or site specific
      arguments, would be given in the actual implemented
      service description and overrides any common information
      found in this generic description. This generic interface
      description can be matched with the site-specific one
      using the role attribute.
    </xs:documentation>
    <xs:documentation>
      Even though the Interface type requires an
      accessURL child element, this element is
      intended to describe a service in the
      abstract--i.e. without reference to a particular
      installation of the service. Consequently,
      the accessURL may contain a bogus URL;
      applications should not expect it to be resolvable.
    </xs:documentation>
  </xs:annotation>
</xs:element>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>

<xs:complexType name="StandardKeyEnumeration">
  <xs:annotation>
    <xs:documentation>
      A registered set of related keys. Each key can be
      uniquely identified by combining the IVOA identifier of
      this resource with the key name separated by the URI
      fragment delimiter, #, as in: ivoa-identifiery#key-name
    </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="vr:Resource">
      <xs:sequence>
        <xs:element name="key" type="vstd:StandardKey"
          maxOccurs="unbounded" minOccurs="1">
          <xs:annotation>
            <xs:documentation>
              the name and definition of a key--a named concept,
              feature, or property.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="StandardKey">
  <xs:annotation>
    <xs:documentation>
      The name and definition of a key--a named concept,
      feature, or property.
    </xs:documentation>
    <xs:documentation>
      This key can be identified via an IVOA identifier
      of the form ivo://authority/resource#name where name is
      the value of the child name element.
    </xs:documentation>
    <xs:documentation>
      This type can be extended if the key has
      other metadata associated with it.
    </xs:documentation>
  </xs:annotation>

  <xs:sequence>
    <xs:element name="name" type="vstd:fragment">
      <xs:annotation>
        <xs:documentation>
          The property identifier which would appear as the
          fragment (string after the pound sign, #) in an IVOA
          identifier.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>

```

```

<xs:element name="description" type="xs:token">
  <xs:annotation>
    <xs:documentation>
      A human-readable definition of this property.
    </xs:documentation>
  </xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>

<xs:simpleType name="StandardKeyURI">
  <xs:annotation>
    <xs:documentation>
      reference that forces an IVOA ID with a # fragment part
      appended to match the standard pattern for registering
      enumeration values with a vstd:StandardKeyList
    </xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:anyURI">
    <xs:pattern value="ivo://[\w\d][\w\d\-\_\.!~*\'(\)\+=]{2,}/([\w\d\-\_\.!~*\'(\)\+=]+/([\w\d\-\_\.!~*\'(\)\+=]+)*)?#[A-Za-z0-9;/\?:@&=\+\$, \-\_\.!~*\'(\)\+=]">
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="fragment">
  <xs:annotation>
    <xs:documentation>
      the allowed characters for a fragment identifier taken
      from rfc2396
    </xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:string">
    <xs:pattern
      value="([A-Za-z0-9;/\?:@&=\+\$, \-\_\.!~*\'(\)\)]|[A-Fa-f0-9]{2})+/">
  </xs:restriction>
</xs:simpleType>
</xs:schema>

```

Appendix B: Change History

Changes since WD-v1.0 20100514:

- short name changed from VOStandard to StandardsRegExt

Changes since WD-v0.4:

- removed App. B. (Sample instance) as examples appear throughout the document.

References

[RI]

Benson, Kevin, Plante, Ray, Auden, Elizabeth, Graham, Matthew, Greene, Gretchen, Hill, Martin, Linde, Tony, Morris, Dave, O'Mullane, Wil, Rixon, Guy, Andrews, Kona 2008, *IVOA Registry Interfaces*, v1.0, IVOA Recommendation, <http://www.ivoa.net/Documents/RegistryInterface/20091104/>

[URI]

Berners-Lee, Tim, Fielding, R., Irvine, U.C., Masinter, L. 1998. *Universal Resource Identifier (URI): Generic Syntax*, IETF RFC 2396, <http://www.ietf.org/rfc/rfc2396.txt>

[RFC 2119]

Bradner, S. 1997. *Key words for use in RFCs to Indicate Requirement Levels*, IETF RFC 2119, <http://www.ietf.org/rfc/rfc2119.txt>

[XML]

Bray, Tim, Paoli, Jean, Sperberg-McQueen, C. M., Maler, Eve, Yergeau, Francois (editors) 2004, *Extensible Markup Language (XML) 1.0 (Third Edition)*, W3C Recommendation 04 February 2004, <http://www.w3.org/TR/REC-xml>

[Schema]

Fallside, David C., Walmsley, Priscilla (editors) 2004, *XML Schema Part 0: Primer Second Edition*, W3C Recommendation 28 October 2004, <http://www.w3.org/TR/xmlschema-0/>

[RM]

Hanisch, Robert (ed.) 2004. *Resource Metadata for the Virtual Observatory, Version 1.01*, IVOA Recommendation, <http://www.ivoa.net/Documents/REC/ResMetadata/RM-20040426.htm>

[RofR]

Plante, R. 2007, *The Registry of Registries v1.00*, IVOA Note, <http://www.ivoa.net/Documents/Notes/RegistryOfRegistries/RegistryOfRegistries-20070628.html>

[VR]

Plante, R., Benson, K., Graham, M., Greene, G., Harrison, P., Lemson, G., Linde, T., Rixon, G., Stébé, A. 2008, *VOResource: an XML Encoding Schema for Resource Metadata*, v1.03, IVOA Recommendation, <http://www.ivoa.net/Documents/REC/ReR/VOResource-20080222.html>

[ID]

Plante, R., Linde, T., Williams, R., Noddle, K. 2005, *IVOA Identifiers v1.1*, <http://www.ivoa.net/Documents/REC/Identifiers/Identifiers-200505XX.html>.

[VDS]

Plante, R., Stébé, A., Benson, K., Dowler, P., Graham, M., Greene, G., Harrison, P., Lemson, G., Linde, T., Rixon, G., Stébé, A. 2009, *VODataService: a VOResource Schema Extension for Describing Collections and Services*, v1.1, IVOA Proposed

Recommendation, <http://www.ivoa.net/Documents/V0DataService/20090903/>

Last modified: Wed May 19 00:06:00 2010