Astronomy and Computing 10 (2015) 88–98

Contents lists available at ScienceDirect

Astronomy and Computing

journal homepage: www.elsevier.com/locate/ascom

Full length article

Client interfaces to the Virtual Observatory Registry

M. Demleitner [a,*], P. Harrison [b], M. Taylor [c], J. Normand [d]

[a] Universität Heidelberg, Zentrum für Astronomie, Astronomisches Rechen-Institut, Mönchhofstraße 12-14, 69120 Heidelberg, Germany
[b] Jodrell Bank Centre for Astrophysics, Jodrell Bank Observatory, Macclesfield, SK11 9DL, UK
[c] H. H. Wills Physics Laboratory, Tyndall Avenue, University of Bristol, UK
[d] Observatoire de Paris VOPDC-IMCCE, 61 Av de l'Observatoire 75014 Paris, France

ARTICLE INFO

Article history:
Received 27 October 2014
Accepted 20 January 2015
Available online 30 January 2015

Keywords:
Virtual Observatory
Registry
Standards

ABSTRACT

The Virtual Observatory Registry is a distributed directory of information systems and other resources relevant to astronomy. To make it useful, facilities to query that directory must be provided to humans and machines alike. This article reviews the development and status of such facilities, also considering the lessons learnt from about a decade of experience with Registry interfaces. After a brief outline of the history of the standards development, it describes the use of Registry interfaces in some popular clients as well as dedicated UIs for interrogating the Registry. It continues with a thorough discussion of the design of the two most recent Registry interface standards, RegTAP on the one hand and a full-text-based interface on the other hand. The article finally lays out some of the less obvious conventions that emerged in the interaction between providers of registry records and Registry users as well as remaining challenges and current developments.

1. Introduction

In Demleitner et al. (2014a), henceforth Paper I, we described the design and maintenance of the Virtual Observatory (VO) Registry as a distributed information system. Conceptually, it is a collection of, by now, about 15 000 registry records. To give the Registry's users – astronomers, the library community, or even the general public – access to this collection, facilities have to be provided that allow focused queries against it. This includes common bibliographic constraints (by author, title or abstract term, year, etc.), but also constraints specific to a registry mainly concerned with data services (e.g., supported protocols or query parameters, metadata of published tables). In the design of such facilities, several challenges have to be addressed:

1. different users have very different expectations and requirements
2. the underlying data collection (i.e., the set of registry records) is changing over time
3. the underlying data structure is fairly complex, and evolves itself as new standards and techniques are introduced in the VO
4. as many uses require only a small subset of the types of metadata contained, partial resource descriptions should be retrievable
5. the total data set cannot efficiently be transferred to clients as a whole
6. registry records are frequently authored by persons not entirely familiar with the data model, resulting in inconsistent quality.

In consequence, no single *user* interface to the Registry can be sufficient. Instead, the VO community designed *client* interfaces, i.e., network endpoints with rigorously defined behavior and semantics, designed for use by programs that then present the actual user interfaces to Registry data.

We will begin this paper with a brief review of the various client interfaces that are or were used in the VO (Section 2). In Section 3, we proceed to describe the use some selected clients make of these facilities and the ways they apply and expose information obtained from the registry. A major part of the paper, Section 4, is devoted to a thorough discussion of the Registry Relational Model (RegTAP for short), one of the two registry interfaces currently being developed and deployed in response to the deficiencies of previous standards. In Section 5, the other new-generation interface is described.

While laying out some common use cases of Registry data in Section 6 we also point out common query patterns. Section 7 concludes with some speculation about probable future developments.

* Corresponding author.
E-mail address: msdemlei@ari.uni-heidelberg.de (M. Demleitner).

http://dx.doi.org/10.1016/j.ascom.2015.01.008
2213-1337/

In the following, we refer to common Registry standard texts by their abbreviated names as introduced in Paper I, and again the capitalized word "Registry" refers to the abstract concept, while concrete services are written in lower case (e.g., a "publishing registry"). Concepts from VOResource and its extensions are written in SMALL CAPS.

2. History

Although only explicitly written down in 2011, the use cases collected on the IVOA wiki (IVOA Registry WG, 2011) outline some of the challenges faced by the designers of the first client interfaces to the registry in the mid-2000s—finding tables containing columns with certain physics, locating services implementing certain protocols, and the like.

While on the maintenance side of the registry the ecosystem around OAI-PMH (Open Archives Initiative, 2002) provided guidance for many technology choices, in developing the client interfaces much more new ground had to be broken. For instance, the OPACs (Online Public Access Catalogs; see Kani-Zabihi et al. (2008) for a treatment from about the time of RI1 design) established in the library community, while comparable for the purpose of locating information resources, could not efficiently address the use cases, and no broadly accepted standard for client, rather than user, interfaces to OPACs, lent itself to adoption by the VO community.

Given that the interface to be designed was expected to be expressive enough for requests of the type "find all TAP services exposing a table having some word in the description and a column with a given UCD,[1]" it was determined fairly early on that an interface based on simple, atomic parameters would not be sufficient, and Registry information crucial to certain discovery tasks would not be queryable through it. Client interfaces making explicit too much of the underlying data model would also unduly restrict future developments of that data model. Thus, at least one interface to the Registry would have to support a full query language. Since the Registry data model was defined in XML Schema, an obvious choice for the query language was XQuery (Robie et al., 2014), a language that essentially extends SQL concepts to querying XML trees.

However, factors against the adoption of XQuery included:

- the heavy use VOResource makes of XML namespaces, which tended to make queries hard to write by hand;
- the much larger installed base of relational databases compared to XQuery-capable engines (compounded by the fact that translating XQuery to a given relational schema is hard);
- the desire to open up the full registry data model to queries written by end users, i.e., astronomers. As it was expected that many of these would familiarize themselves with the VO's SQL dialect ADQL (Astronomical Data Query Language; Ortiz et al. (2008)), requiring yet another query language for Registry access appeared undesirable.

With these considerations, it was decided to base the primary Registry interface on conventional relational technology.

While the complex queries XQuery and ADQL allow were needed for identified use cases, it was also acknowledged that "Google-like" searches – more or less loose matching of words in documents modeled as bags of words – was the dominant mode of searching for resources outside of the VO in the targeted user base. At least if common "comfort" features like stemming or phrase searches are desired, this type of search is hard or impossible to simulate through plain ADQL given its very basic set of text search capabilities. Therefore, a keyword search operation with significant freedom for implementors was also defined.

The result of these considerations was Section 2 of RI1 (Benson et al., 2009). It defines two required search operations *Search* (with constraints in ADQL) and *KeywordSearch* (with operator-defined matching of keywords against an operator-extensible minimal set of fields) as well as an optional *XQuerySearch* operation. All search operations return either identifier lists or sequences of full resource records in OAI-PMH style. In addition, two OAI-PMH-like operations were defined, *GetResource* to obtain a resource record from an identifier, and *GetIdentity* to discover metadata about the registry service itself.

Several implementations of the standards are available; services are provided by STScI, ESA, and AstroGrid.

As the RI1 design significantly predates the final standardizations of both ADQL (Ortiz et al., 2008) and the transport protocol for queries and results – that was eventually defined in the TAP standard (Dowler et al., 2010) –, RI1 further defined an ad-hoc transport based on the RPC mechanism SOAP, and it adopted ADQL at a time when experiments were underway with passing ADQL statements to client interfaces in parsed (XML) form. In consequence, modern TAP clients cannot use registry endpoints, and writing queries in the aging XML serialization of ADQL became at least difficult as software components translating SQL expressions into the XML forms went unmaintained.

Further critique came from implementor feedback (e.g., Taylor, 2010) and was collected together with the use cases (IVOA Registry WG, 2011). For instance, in practice the use of a restricted set of XPath to specify constraints instead of defining an actual relational schema lead to severe interoperability problems between different registries, which were further exacerbated by not specifying rules for case folding. The apparent flexibility towards registry extensions provided by the XPath-based column references also did not pay off as originally expected since registries still needed to do internal mapping as registry extensions were developed. In contrast to the (optional) XQuery interfaces, the (mandatory) ADQL interfaces frequently lagged behind standards deployment.

In this situation, the most advanced Registry clients relied on the optional XQuery interface or even used entirely proprietary interfaces.

As TAP services entered the registry in the early 2010s, RI1's response format also became a liability. Registry records contain table metadata, and with TAP services exposing many tables, resource records of several megabytes are not exceptional. This made relatively common queries like "Retrieve basic metadata on all TAP services" expensive in terms of transfer time and processing required.

Therefore, starting in 2011, it was decided to design a new Registry interface, dubbed "RESTful" to contrast it from the RI1 SOAP-based protocol. With TAP and ADQL now available, a replacement of the RI1 *Search* operation was mainly a matter of designing a schema and a mapping to this schema from VOResource. This can be seen as creating a second serialization of an abstract data model implicit in VOResource's XML schema files.

The combination of a defined schema and a TAP service had a model in ObsCore (Louys et al., 2011). The resulting new standard ("RegTAP"), discussed in Section 4, is in the last phases of IVOA peer review as this article is written.

A replacement for the *KeywordSearch* operation is also being developed. Here, the wide availability of feature-rich fulltext engines such as Apache Lucene offers the possibility of enriching the bag-of-words model and allows some advanced operators as well. We will revisit this development in Section 5.

[1] Unified Content Descriptors or UCDs in the VO denote physical concepts like "angular distance" or "radio flux" in a simple formal language (Derriere et al., 2004).