1. Practical Interoperability in the VO

Markus Demleitner
msdemlei@ari.uni-heidelberg.de

VO = Virtual Observatory.

A global data infrastructure in Astronomy comprising

- $\sim$ 50 data centers in $\sim$ 20 countries
- $\sim 3 \times 10^4$ data collections
- hundreds of millions of data sets (spectra, images,...)
- hundreds of billions of table rows

(cf. Fig. 1)

2. What's the VO?

It is...

not a website ("platform"),

not a bunch of websites,

not a programme that does all things astronomy.

Instead...

3. The VO is...

Standards for finding, accessing, using, and describing data (more on which in a moment)

plus

$\sim$ 50 data centers worldwide adhering to these standards (which includes almost all the major players like ESO, NASA, ESA, etc)

plus

a few volunteers operating some $\pm$ central infrastructure (these are things like searchable Registry endpoints, our document repository, and the various bodies working on the standards)

plus

authors of client software, libraries, and web pages making these resources available to astronomers and the public (newcomers might want to look at TOPCAT and Aladin on the desktop, pyVO and STIL as libraries, or ESA Sky or Aladin lite in the browser).

4. Why no Platform?

We couldn't do a platform if we wanted to. To start with, getting the sort of international funding we would need is probably close to impossible, and then if we got that, there are so many different sub-disciplines that we'd need a large establishment that would start mainly to deal with itself.

But more importantly: With multiple

interoperable

(i.e., they can be used in a well-defined, uniform way by machines) providers the VO can

grow from the edges:

Users control

their end of processing, operators can adapt services to their needs and

evolve the standards.

No single part can dictate what happens. Not to mention it saves a lot of money if people don't have to write elaborate web pages per project, and if software written for one data collection just works (adapting for differences between instruments, that is) for another data collection.

But then, of course, people can build platforms *on top* of our standards ("APIs").

5. How do we do it?

In the VO data discovery conceptually is a two-step process:

1. Ask the Registry which services *might* have data of interest
2. Have a machine ask these services one by one.

Example: "Give me images of Barnard's star in X-rays".

6. Barnard in X: Step 1

A client programme asks the Registry: What "resources" (services, data collections, whatever) are out there that:

- serve images
- have data in the X-ray part of the spectrum
- have data around $\alpha = 269.45$, $\delta = 4.693$ (that's where Barnard's star is these days)?

The registry responds with a list of access URLs for such services, and the protocols to speak when talking to them.

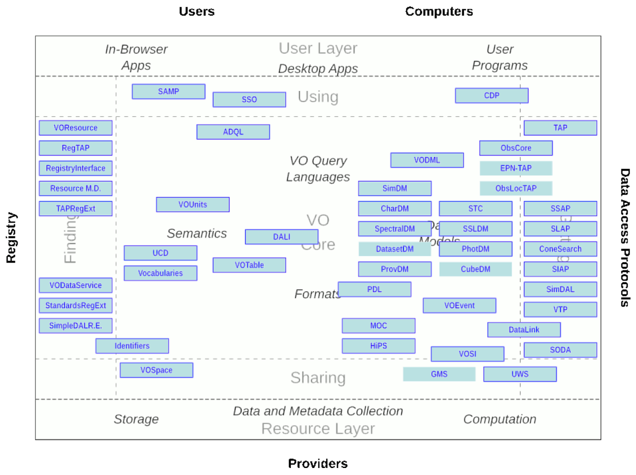

Fig. 2

7. Barnard in X: Step 2

The client now goes to each service in the list and asks it for data...

- covering the position $\alpha = 269.45$, $\delta = 4.693$,
- intersecting the spectral range $0.1 \cdots 120\,\mathrm{keV}$ of photon energy.

Each server responds with one set of metadata per matched image. The client turns this into some representation for the user.

8. Barnard in X: Step 3

The user picks datasets of interest based on their metadata.

The client retrieves images (or parts thereof) for further processing.

In practice, that's a lot more natural than it sounds here.

9. What standards?

(cf. Fig. 2)

To make this happen, a lot of components have to work together, and so quite a few standards have to play together. Our "architecture diagram" is one way to organise this; there's no need to understand this deluge of acronyms – but it turns out that this is about what it takes to build an interoperable data infrastructure pretty much across disciplines.

Looks scary, but trying to dodge this effort (that, by the way, is helped a lot by looking at other communities that have similar things already) will only make things worse by accumulating non-interoperable legacy that your standards will later have to deal with.

10. What?

Here is a run-down of what we've had to standardise:

Searching for data: Images (SIAP), spectra (SSAP), objects (SCS), spectral lines (SLAP), generic datasets (ObsCore).

Remote manipulation: SODA lets you do cutouts, rescaling, etc., to avoid pulling data you don't need.

Interacting with databases: Access using TAP, common query language ADQL. and...

11. What? (cont'd)

...and

Formats: Table exchange using VOTable, complex spherical geometries with MOC, multiscale images with HiPS.

Registry: Registry Interfaces for the architecture, VOResource, VODataService, TAPRegExt, SimpleDALRegExt for the metadata format, RegTAP for how to search it.

Semantics: Light semantics of physical quantities (UCD), Unit syntax, Vocabulary maintenance.

SAMP: Assembling complex environments from simple building blocks.

12. Case Study: Registry

The Registry is our metadata catalogue, and our architecture might be useful as a blueprint beyond astronomy.

The backbone is OAI-PMH, which is used by searchable registries (of which we currently have three major ones worldwide, run by volunteer institutions) to harvest publishing registries, typically once a day.

The searchable registries learn which publishing registries are there from the only central component, the Registry of Registries.

Clients talk to the searchable registries, so there's no single point of failure, since the RofR is only required for publishing registry enrollment.

If I had to design it again, the only thing I'd think hard about is if there's any chance to build in a trigger that would let a publishing registry send out a "harvest me now" signal to all interested harvesters (perhaps via XMPP? or OStatus?). Oh, and I'd do OAI-PMH incrementals based on tokens rather than dates.

(cf. Fig. 3)

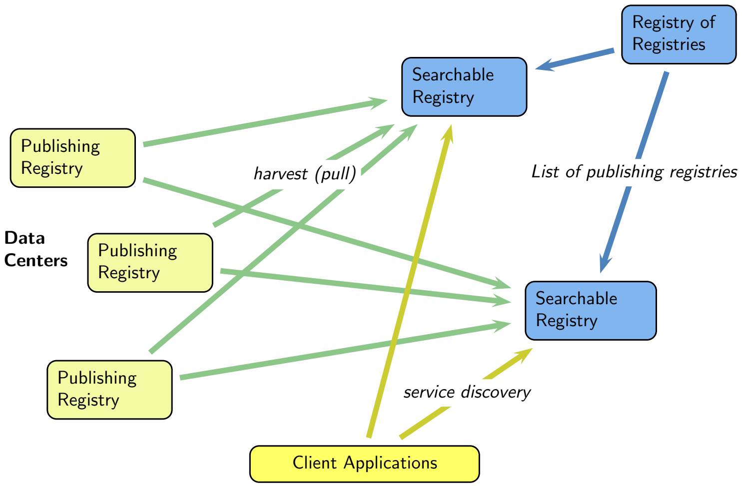

Fig. 3

13. Learning More

- General VO info: https://ivoa.net
- Use cases and tutorials: https://dc.g-vo.org/VOTT
- Registry architecture: Demleitner et al, The Virtual Observatory Registry. Astronomy and Computing, 7 pp. 101-107, astro-ph 1407.3083.

Thanks!