1. Coping with Major Version Changes

Markus Demleitner
msdemlei@ari.uni-heidelberg.de

Starting point: Add `servicetype=’sia2’` to pyvo’s registry.search.

So far, `’image’` was an alias for `’sia’`.

What should it be now? More generally: What’s sane client behaviour on major version changes?

Funded by e-inf-astro, BMBF FKZ 05A20VH5

2. Opening Remark

Don’t use servicetype constraints going forward.

Data product types are not a property of the service protocol (think obscore, think timeseries in ssa). They are a property of the data collection (which can be accessible through multiple protocols).

Let’s further move to *data discovery* and add a product-type declaration to VODataService

3. Drop image?

Option I: `servicetype=’image’` will give a DeprecationWarning and keep current behavoiur otherwise.

If we are serious about getting rid of servicetype as fast as possible, that’s the way to go. It’s also the simplest options implementationally.

4. Break image?

Option II: Make `servicetype=’image’` equivalent to `servicetype=’sia2’`.

This would be right if we expected SIAP2 to take over all of SIAP. Eight years after SIAP2: 271 SIAP1 vs. 99 SIAP2. Also: SIAP2 services have a rather different interface from SIAP1 services. That’s not even considering that people really don’t like it when the matches for their registry queries *totally* change after a software update. So, I’m pretty sure that’s the worst option of them all.

5. Fix image?

Option III: Make `servicetype=’image’` return some sort of “union” of sia and sia2.

Problem: What interface would that have? A program has know what parameters it can pass to a service object.

The only solution I can see is to return adapter objects that support POS and returns obscore-like records for both sia and sia2.

Note that that breaks programs that use `servicetype=’image’` and then use non-pos SIAP1 constraints unless we were to do a really fancy SIAP1 emulation layer (I’ve not looked at the feasiblity).

6. De-duplicate?

But many resources have both sia and sia2 capabilities.

- Option IIIa: produce service objects for both That’s actually rather significant pain, because we then have to make two pyvo service objects from one RegTAP results, which is at least unusual. Also, if all works well, it would result in all products from such resources would be present twice in the full result, which probably never is useful.
- Option IIIb: prefer SIAP1 (perhaps under the assumption the services are more mature)
- Option IIIc: prefer SIAP2 (on grounds that that’s where folks should be migrating to anyway)

7. And Obscore

Of course, there’s also Obstap that people can use to discover images. Why shouldn’t that come back for the servicetype image?

…ignoring for a moment the difficulty of figuring out the pertinent value of obs_collection. Background: You probably want one record per image collection, and you’ll have multiple image collections per obscore service in general. There’ll be a servedBy relationship from collection to service. But to do SIAP1-style querying of an obscore service, you must constrain the obs_collection column to whatever is appropriate for the collection, and we have no way to say what to put there at the moment.

8. In the End

What’s your preference for limping on with service discovery?

But I’d still say: Let’s move to data discovery and have per-resource decisions what protocol to use.