*Fig. 1*

*Fig. 2*

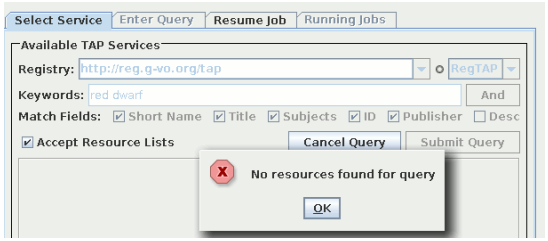

*Fig. 3*

# 1. Touch TAPRegExt

(cf. Fig. 1)

Markus Demleitner
*msdemlei@ari.uni-heidelberg.de*

(cf. Fig. 2)

(cf. Fig. 3)

This is the problem we're trying to solve: There's quite a few tables searchable by TAP talking about red dwarfs. However, there's nothing in the registry that lets you find them.

# 2. Proposed Solution

After much experimenting, we came up with this pattern:

- capability/@standardId `ivo://ivoa.net/std/tap`: Look for those if you want to enumerate all TAP services (for validation, all-VO searches, DM discovery...)
- capability/@standardId `ivo://ivoa.net/std/tap#sync-1.0-aux`: These just give an access URL, a single service may have thousands of those



# 3. Does it work?

Sure. (We're just using aux as fragment identifier yet, but that's a detail).

GAVO's DC already puts out aux records, and NVO Validation has learned to ignore them (enumerate-all use case)

In WIRR (http://dc.g-vo.org/WIRR), you'll find something if you look for TAP services about "neutrinos", and you can send the TAP access URL to TOPCAT (the find-single use case).

# 4. But – VizieR?

VizieR would just have to add

```
<capability standardID=
    "ivo://ivoa.net/std/TAP#sync-1.0-aux">
  <interface xsi:type="vs:ParamHTTP" role="std">
    <accessURL use="base"
      >http://vizier.u-strasbg.fr/viz-bin/VizieR<accessURL>
  </interface>
</capability>
```

to each of the Cone Search records they're already putting out (plus probably another for async, if we follow this pattern).

If they want to be fancy, they'll set a served-by relationship to their TAP record, too.

# 5. Have a main rec?

Should VOResource get an AuxCapability type for such capabilities?

Should that type have a mainRecord child with the IVORN of a record having the "full" capability?

[My take: not much more useful than just setting a relationship]

# 6. Extra trouble for TAP

In services with 100+ tables, being sent to the endpoint may not be enough –

The client may want to "open" the table in question, too.

Good news: No extra Registry features necessary – clients can discover the table from the record's tableset element.

(Almost, at least. VODataService isn't as clear as TAP yet on having query-ready table names in tableset, and that's what clients would need – but that needs to be fixed in VODataService, and it needs to be fixed anyway)

# 7. Left to be done

- Write a note to explain the principle?
- Add a section to TAPRegExt 1.1 discussing #aux capabilities
- Teach clients to search for capid
- Update Registry extensions for SIA, SSA, Cone Search, too

Should I finally go ahead?