Fig. 1

Fig. 2

Fig. 3

1. TAP 1.1: Registry Woes

(cf. Fig. 1)

Markus Demleitner
msdemlei@ari.uni-heidelberg.de

(cf. Fig. 2)

- TAP discovery now
- Current TAP 1.1 proposal
- Consequences
- A way out?

(cf. Fig. 3)

2. TAP Discovery Now

The RegTAP pattern to enumerate all available TAP services currently is:

```
SELECT ivoid, access_url
FROM rr.capability
NATURAL JOIN rr.interface
WHERE standard_id like ’ivo://ivoa.net/std/tap%’
  AND intf_type=’vs:paramhttp’
```

TL;DR: I'd like to keep it that way for 1.x (or not make it a lot more complicated).

Actual discovery queries by clients tend to get more complicated, but this basic pattern holds. True: clients should already weed out multiple access URLs for a given service, but even that isn't necessary in practice right now (no service gives more than one).

3. Enter TAP 1.1

The current TAP 1.1 PR shows the following:

```
<capability standardID="ivo://ivoa.net/std/TAP">
  <interface xsi:type="vs:ParamHTTP" role="std" version="1.0"> ...
  <interface xsi:type="vs:ParamHTTP" role="std" version="1.0"> ...
    <securityMethod standardID="ivo://ivoa.net/sso#tls-with-certificate"/>
  </interface>
</capability>

<capability standardID="ivo://ivoa.net/std/TAP#async-1.1">
  <interface xsi:type="vs:ParamHTTP" role="std" version="1.1"> ...
  <interface xsi:type="vs:ParamHTTP" role="std" version="1.1"> ...
    <securityMethod standardID="ivo://ivoa.net/sso#BasicAA"/>
  </interface>
  <interface xsi:type="vs:ParamHTTP" role="std" version="1.1"> ...
    <securityMethod standardID="ivo://ivoa.net/sso#tls-with-certificate"/>
  </interface>
</capability>

<capability standardID="ivo://ivoa.net/std/TAP#sync-1.1">
  ...
```

4. Registry Woes

There's at least two of them:

1. With the current RegTAP pattern, this declaration will result in *eight* access URLs. Of these, only one will actually work as expected.
2. Clients interested in public services [my expectation: the wide majority] have to filter out access-controlled services (which is reasonably painful in RegTAP 1.0)

Actually. . .

5. Discovery of Restricted Services

Well: How urgent is it?

The main use case I can see right now for having these in the Registry in the first place is:

> Which services (of kind X, with columns Y, described like Z, etc) can I access with the credentials I have?

Do we have an idea how that might work? [I don't]

6. Proposed Solution

The base of the trouble apparently is restricted services.

For them, securityMethod probably isn't sufficient to cover the one identified use case for them.

What if we defined (in a quick Note?) a special interface type ("sec:SecuredHTTP"?) in which whatever is required can evolve?

7. Resulting Capability Output

```
<capability standardID="ivo://ivoa.net/std/TAP#query-1.1">
  <interface xsi:type="vs:ParamHTTP" role="std">
    <accessURL use="base">http://example.com/tap<accessURL>
  </interface>
  <interface xsi:type="sec:SecuredHTTP">
    <accessURL use="base">http://example.com/rtap<accessURL>
    <authMethod standardId="ivo://ivoa.net/sso#tls-with-certificate"/>
    <authMethod standardId="ivo://ivoa.net/sso#BasicAA"/>
    <authz registrar="ivo://example/project">
      <authorizedGroup>projectA_masters</authorizedGroup>
      <authorizedGroup>projectB_peasants</authorizedGroup>
    </authz>
  </interface>
</capability>
```

(`sec:SecuredHTTP` being a complete fantasy at this point)

8. Well?

Pro:

- Old clients will just continue to work
- Fairly clear path ahead for non-conflicting support for restricted services

Against:

- Need to work out sec:SecuredHTTP schema
- TAP sync and async are modeled as one capability, which may not scale to other standards

And: What about VOSI, examples endpoints?

Now: You.