TAP and Authentication

- approach #1: it's horrible and it breaks stuff
- approach #3 works, BUT: introduces subtle use of interface types and mixed interface style in a single capability
- approach #2: separate #sync-1.1 and #async-1.1
  - matches the VOResource/VOSI-capabilities design
  - works the same way as all other IVOA services
- I am convinced that:
  - we made a mistake in TAP-1.0 when we specified one standardID for two features
  - when we make mistakes we have to admit it and stop doing it in order to improve our standards
  - sometimes that makes things a little more complex

TAP 1.1 Capabilities: Bundles

Why I don't much like TAP 1.1 authenticated service specification

- TAP service interaction uses several endpoints:

  `sync`, `async`, `tables`, `capabilities`, `examples`

- Clients like TOPCAT/STILTS need to work with a *bundle* of these, not just pick one
- TAP 1.0: service defined by base URL
  - Bundle specification: base URL + well-known subpaths

    `http://dc.g-vo.org/tap/sync`
    `http://dc.g-vo.org/tap/async`
    `...`
- TAP 1.1: service defined by `capabilities` document
  - Capabilities doc provides an unstructured list of (endpoint, securityMethodID) pairs
  - Bundle specification: capabilities doc + securityMethodID + bundle-assembly rules
  - There is no base URL
- TAP 1.1 is hard work for bundle-oriented clients:
  - Fairly difficult to list available bundles to offer to users
    (but not impossible — I implemented it)
  - Can't specify a bundle by just giving a URL e.g. on the command line or in an email
    (except by fallback to TAP 1.0 rules)
  - Very difficult to deal with services that may be mirrored as well as authenticated
    (I gave up)

TAP 1.1 Capabilities: Example and discussion

PR-TAP-1.1-20181024 Section 2.4 "VOSI-capabilities":

- Explains use of capabilities file to specify endpoints
- Includes text explaining how to interpret it as bundles
  - Text supplied by me —
    as simple as I could make it for e.g. TOPCAT requirements, but still a bit involved
- Relies on (draft) UWSRegExt Note, so provisional and non-normative
  - PR-TAP-1.1-20181024 therefore does not tell clients how to use authenticated services (just guesses how it might work in future)
  - If UWSRegExt doesn't work out as per its current draft,
    this section will be unhelpful/misleading

TAP 1.1 Capabilities: Suggestions

Bundles: return to Base-URL-based system

- Markus suggests new DALI subtype of `vr:interface` (see registry mailing list 17 Oct):

```
<capability standardID="ivo://ivoa.net/std/TAP">
  <interface xsi:type="vs:DALIInterface">
    <accessURL>http://dc.g-vo.org/tap"</accessURL>
    <endpoint>sync</endpoint>
    <endpoint>async</endpoint>
    <endpoint>capabilities</endpoint>
    <endpoint>tables</endpoint>
    <endpoint>examples</endpoint>
  </interface>
</capability>
```

- I think this would reduce the complexity of Sec 2.4 and remove the need for UWSRegExt
- `<mirrorURL>` elements would become usable
- Looks reasonable to me? But I'm not a registry expert

Capabilities discussion: extract to TAPRegExt

- Capabilities section is not really core to TAP 1.1, and is anyway subject to change and non-normative
- TAPRegExt 1.1 is currently in WD

⇒ Punt capabilities details to TAPRegExt?

... blame Markus for this suggestion too

TAP and Authentication

- approach #1: it's horrible and it breaks stuff
- approach #3 works, BUT: introduces subtle use of interface types and mixed interface style in a single capability
- approach #2: separate #sync-1.1 and #async-1.1
  - matches the VOResource/VOSI-capabilities design
  - works the same way as all other IVOA services
- I am convinced that:
  - we made a mistake in TAP-1.0 when we specified one standardID for two features
  - when we make mistakes we have to admit it and stop doing it in order to improve our standards
  - sometimes that makes things a little more complex

NRC·CNRC

TAP 1.1 Capabilities: Bundles

Why I don't much like TAP 1.1 authenticated service specification

- TAP service interaction uses several endpoints:

  `sync`, `async`, `tables`, `capabilities`, `examples`

- Clients like TOPCAT/STILTS need to work with a *bundle* of these, not just pick one
- TAP 1.0: service defined by base URL
  - ▷ Bundle specification: base URL + well-known subpaths

    `http://dc.g-vo.org/tap/sync`
    `http://dc.g-vo.org/tap/async`
    ...
- TAP 1.1: service defined by `capabilities` document
  - ▷ Capabilities doc provides an unstructured list of (endpoint, securityMethodID) pairs
  - ▷ Bundle specification: capabilities doc + securityMethodID + bundle-assembly rules
  - ▷ There is no base URL
- TAP 1.1 is hard work for bundle-oriented clients:
  - ▷ Fairly difficult to list available bundles to offer to users
    (but not impossible — I implemented it)
  - ▷ Can't specify a bundle by just giving a URL e.g. on the command line or in an email
    (except by fallback to TAP 1.0 rules)
  - ▷ Very difficult to deal with services that may be mirrored as well as authenticated
    (I gave up)

TAP 1.1 Capabilities: Suggestions

Bundles: return to Base-URL-based system

- Markus suggests new DALI subtype of `vr:interface` (see registry mailing list 17 Oct):

```
<capability standardID="ivo://ivoa.net/std/TAP">
  <interface xsi:type="vs:DALIInterface">
    <accessURL>http://dc.g-vo.org/tap"</accessURL>
    <endpoint>sync</endpoint>
    <endpoint>async</endpoint>
    <endpoint>capabilities</endpoint>
    <endpoint>tables</endpoint>
    <endpoint>examples</endpoint>
  </interface>
</capability>
```

- I think this would reduce the complexity of Sec 2.4 and remove the need for UWSRegExt
- `<mirrorURL>` elements would become usable
- Looks reasonable to me? But I'm not a registry expert

Capabilities discussion: extract to TAPRegExt

- Capabilities section is not really core to TAP 1.1, and is anyway subject to change and non-normative
- TAPRegExt 1.1 is currently in WD

⇒ Punt capabilities details to TAPRegExt?

... blame Markus for this suggestion too

TAP 1.1 Capabilities: Example and discussion

PR-TAP-1.1-20181024 Section 2.4 "VOSI-capabilities":

- Explains use of capabilities file to specify endpoints
- Includes text explaining how to interpret it as bundles
  - ▷ Text supplied by me —
    as simple as I could make it for e.g. TOPCAT requirements, but still a bit involved
- Relies on (draft) UWSRegExt Note, so provisional and non-normative
  - ▷ PR-TAP-1.1-20181024 therefore does not tell clients how to use authenticated services (just guesses how it might work in future)
  - ▷ If UWSRegExt doesn't work out as per its current draft,
    this section will be unhelpful/misleading