Fig. 1

Fig. 2

1. Identifiers 2

(cf. Fig. 1)

Markus Demleitner
msdemlei@ari.uni-heidelberg.de

(cf. Fig. 2)

- Why?
- Miscellaneous changes
- Standard identifiers
- DIDs

2. Why?

Identifiers Version 1 almost implies 1:1 between IVO identifiers and Registry records.

Meanwhile: Dataset IDs, Standard identifiers, VOSpace IDs. . .

Consequence: Diverging practices.

Norman's critique of # in DIDs was the final wakeup call.

So: Identifiers 2 is "Identifers for everything you can find *through* in addition to *in* the Registry".

Also: Identifiers 2 is "minimal diff to RFC 3986" (hence the size of the diff to Identifiers 1)

3. IVOID versus IVORN

Over time, URIs starting with `ivo://` have had lots of names. We now propose the clarification:

- IVOID – any standards-compliant URI with a scheme of `ivo`
- IVORN ("IVOA Resource Name") – an IVOID without query or fragments part, i.e., one that references an entire registry record

Should be largely compatible with existing usage. Or is it?

Also note that the URI equivalent of IVORN, the URN, has been deprecated by W3C. So, if in the remote past it was hoped that IVORNs would be persistent in some sense (which URNs were meant to be), that insinuation is now gone one way or the other.

4. Miscellaneous Changes

- No "XML form" of identifiers any more. You probably don't care about these, but that's the formal reason for bumping the major version number
- Now allowing %-encoding in query and fragement
- empty, ., and .. path segments are now forbidden
- IVORNs now MUST be used ignoring case
- Dropped recommendation to present authorites in lower case

5. Standard Identifiers

New standard form for ids for standards (as in: capability identifers, etc):

$$\underbrace{\texttt{ivo}:\texttt{//ivoa.net/std/DataLink}}_{\text{(Service)?Standard}}\#\overbrace{\texttt{links-1.0}}^{\text{standard key}}$$

Note that the fragment *never* points to an `endorsedVersion` – these are not persistent. This also lets a single standard have multiple standardIds (think VOSI).

Consequences:

- Standards records will have to enumerate their endpoint versions
- Clients may have to query for partial identifiers (`LIKE '...links-1.%'`)

6. Dataset Identifiers

To ensure that:

- A DID is resolvable
- The registry isn't burdened with lots of datasets (its job are data collections)
- We keep IVORN fragment semantics within common URI customs

it's almost inevitable to choose:

$$\underbrace{\texttt{ivo://ex.ivoa/some/res}}_{\text{DataCollection or DataService}}\overset{\text{Resolving barrier}}{\overset{\downarrow}{?}}\underbrace{\texttt{local/part\&anything=goes}}_{\text{recommended: simple path}}$$

7. Resolving DIDs

As datasets may be published through a variety of services, there is no simple, general procedure for resolving DIDs. Non-normative *proposition* in the spec:

1. Strip IVOID to IVORN
2. Get Registry record (=R)
3. Try Datalink if capability present in R
4. Try SSA if capability present in R
5. Try ObsTAP if capability present in R
6. Try Datalink, SSA, ObsTap for all resources that are in a servedby relationship to R

8. Winding Down

There's a validator for IVOIDs to try and break:
http://dc.g-vo.org/validate-identifier[1]

What people complained about online (a.k.a. hints on what we should discuss):

- Have a canonical way from DID to data product?
- Do we want version and endpoint type in the capability id?

[1] `http://dc.g-vo.org/validate-identifier`