1. StandardsRegExt: Review?

Markus Demleitner
msdemlei@ari.uni-heidelberg.de

- What's StandardsRegExt?
- Problem 1: What about Endorsed Notes?
- Problem 2: What to register when?
- Problem 3: . . . and who does it?
- Problem 4: And who gets to add StandardKeys?
- These ways out.

(cf. Fig. 1)

2. What's StandardsRegExt?

It's a vintage 2012 Registry extension for

- Registering things in the IVOA document repository (RECs, but potentially also Notes, PRs, WDs)
- with current versions, schema files, and
- StandardKey-s.

See the ServiceStandard type[1] for its content schema.

Most important actual use: standardIDs (& friends). Almost everything starting with ivo:// in a TAP capability[2] points to/into a StandardsRegExt record.

[1] `http://docs.g-vo.org/schemadoc/schemas/StandardsRegExt_xsd/complexTypes/ServiceStandard.html`

[2] `http://dc.g-vo.org/tap/capabilities`

3. Funky Example

DaCHS' ADQL has a few extensions declared in TAPRegExt `feature`-s like this:

```
<languageFeatures type="ivo://org.gavo.dc/std/exts#extra-adql-keywords">
  <feature>
    <form>TABLESAMPLE</form>
    <description>Written after a...
  <feature>
    <form>MOC</form>
    <description>A geometry function creating MOCs....
```

If you retrieve that Registry resource referenced, you'll see an explanation of what is going on here:

```
<endorsedVersion status="n/a" use="preferred">1.0</endorsedVersion>
<key>
  <name>extra-adql-keywords</name>
  <description>This is a TAPRegExt languageFeature group ...
```

This is certainly a somewhat exotic use of StandardsRegExt, but it shows how it can even help evolve standards smoothly and transparently; note how every other TAP implementor can re-use these identifiers, and anyone running a publishing registry could come up with them.

Disclaimer: These days, I'd probably use vocabularies for many of the things we've chosen standard keys for in the past: They're smoother and have well-defined processes out of the box.

4. Problem 1: ENs

This whole thing started by me wanting to register the Endorsed Note on UAT adoption in the IVOA. StandardRegExt's `endorsedVersion` has:

`status = ("rec" | "pr" | "wd" | "iwd" | "note" | "n/a")`

That is: No code for Endorsed Notes.

What do we now now? Options:

1. Touch StandardsRegExt and add `en`? Perhaps even `pen`? Of course, that's work, and we'll have to find someone who does it – and then perhaps tackle a number of other problems in StandardsRegExt, too.
2. Don't register ENs at all? That's not a big deal at this point, but it would be once an EN wants to define standard keys. Perhaps ENs simply should not do that and turn into RECs if they need StandardsRegExt StandardKey-s?
3. Register ENs using n/a? I am not aware that anything is using endorsedVersion/@status operationally, so there is probably no technical reason not to do that. But it doesn't feel right either.

Assuming we go for (1), in the following I am collecting a few additional problem we ought to fix when we go for StandardsRegExt 1.1.

5. Problem 2: What Do We Register?

StandardsRegExt does not come with a section "Use Cases" so far, so we don't know what we should register when; current practice is… not totally consistent.

Options:

1. Only RECs (and perhaps ENs) requiring StandardKeys and/or standardID-s? As long as we have these references into the Registry, we can't do without this, as an ivoid needs a Registry record it resolves to.
2. All RECs? (this is about what we think we are doing now)
3. All RECs and PRs (this is what ivoatexDoc recommends in order to let standard keys resolve during the review; it's rarely been done that way, though)
4. Everything on the REC/EN tracks?
5. Everything in the Doc repo? (which would have the advantage that potentially, we could generate the document repo from the Registry – but there are so many technicalities involved that *if* we wanted that, we'd have to develop the standard together with the software that would generate the doc repo from the Registry)

Me, I'm leaning towards (1) or perhaps (3) at this point, unless we actually wanted to use the Registry to manage the document repository – which I'd only consider if we merged the RofR (where our standards are registered) and the document repo.

6. Problem 3: Who Writes them?

So far, it was mainly I and sometimes a few other enthusiasts who have written the StandardsRegExt records. When we happened to notice something was missing.

If we think we want the records for all standards, we have to improve this.

With ivoaTEX, making these records isn't terribly hard any more. We could make default "make test" rules. SRE management could then become part of the upload procedure.

But that only leads back to the question: Is it worth the effort? If there are no use cases for standards without keys or references to the standard itself, should we bother?

7. Problem 4: Creating Keys

So far, we are not sure whether StandardKey-s can be entered at will, only by the standard (in a REC process) itself or perhaps in other standards, too. The extreme example here is the TAPRegExt/ADQL combo. TAPRegExt describes things in ADQL and TAP, but it was thought it shouldn't regulate into their Registry records, so most of the TAP-related ids now point into TAPRegExt's record. The late consequence: ADQL 2.1 will create a plethora of keys in TAPRegExt…

Options:

1. Have a global procedure like the vocabularies' VEPs? Given the diverse use of StandardKeys, I don't think we should do that; if you want VEPs, use vocabularies.
2. Tell standard authors to define (simple) management procedures/rules per keys-using standard.
3. Leave it to RofR maintainers/the TCG/the Exec on a case-by-case basis?

8. Final Problem: Human Resources

In other words: Who will do the changes?

I'm in (whatever we decide), but only when it's not me alone.

…Thanks!