Fig. 1

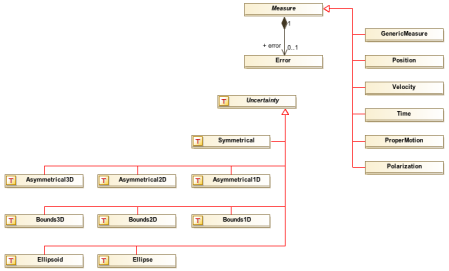

Fig. 2

1. DM Posture Review

Markus Demleitner
msdemlei@ari.uni-heidelberg.de

Basic question: Should we have many small, isolated DMs or fewer larger, entangled ones?

- How it came up: Measurements
- Why entangled DMs are operationally difficult
- An escape route...
- ...and another one.

(cf. Fig. 1)

2. A Measurements Ouch

Measurements ought to model the distribution-ness of experimental results: You don't have "$v_r = 40\,\mathrm{km/s}$", you have "v_r is drawn from $N(40, 3)$" or $v_r = 40 \pm 3\,\mathrm{km/s}$ (I'm aware it's even more complicated than that, but this is what almost all catalogues make it look, so it ought to be enough for the present considerations.)

Against that, here's a compact representation current model:

(cf. Fig. 2)

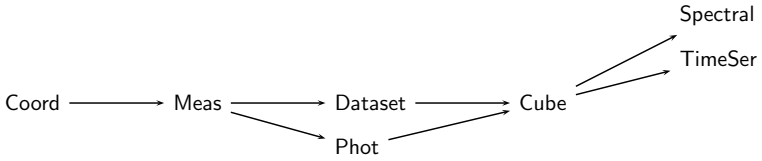

Fig. 3

3. Domain Mismatch

Much of the complexity of the model is the result of attempting to entangle it with coordinates where the domain doesn't require it: Errors on energy are no different from errors on time or on distance.

And, in particular: Errors are usually not correlated because things are vector components but because of how they're measured. For instance, when doing stellar modelling, the errors on T_{eff}, $\log(g)$, and Z will usually be correlated although of course they don't form a vector. Or, for Gaia, all parts of the astrometric solution, including parallax, proper motion, and possibly radial velocity, are correlated.

So: Disentangle Meas and Coords, and you'll get a simpler, more expressive Meas model better adapted to the domain.

4. But That's Beside the Point Here

All that is material for the WG review.

But discussion there quickly turned to the general DM architecture. Mark C-D pointed to this passage of the VO architecture:

> Some of the VO Data Models are specific to a type of data collection (e.g. Spectrum DM, SSLDM, ObsCoreDM, ObsProvDM, PhotDM, SimDM), while others (e.g. STC, Units, Utypes, CharDM) are more foundational and are components of the more specific Data Models. In general, each Data Model can be considered as a building block which can be referred from some other Data Models.

Given that with our VO-DML plans, DMs have become operationally important, I disagree with this basic stance.

5. Model Evolution

Even if entangling models were desirable for some reason, it's a time bomb.

Consider Mark's model dependency graph:

(cf. Fig. 3)

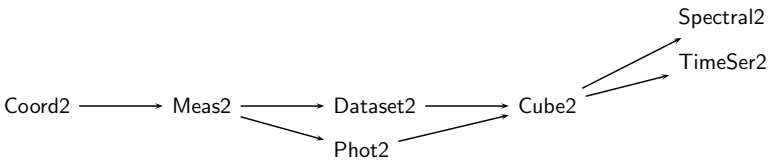

Fig. 4

6. Model Evolution Catastrophe

Suppose we want to evolve Coord to Coord2 (and we will want to!). This will break Meas, so we'll have to have Meas2, and so on, until:

(cf. Fig. 4)

We have touched *all* our data models and broke *all* instance document although only one model changed.

Let's not do that. By the way, we can't.

7. Small and Isolated is Good Sense

There are other good reasons not to needlessly entangle DMs:

- Avoid the horror pluriboxii. That's when people don't even consider adopting a data model because they see so many boxes and arrows that they don't know where to start; I'm saying this as someone who's been trying to get people to adopt SimDM and STC-1 for the past 10 years. Instead, offer tangible benefits just for adding a few pieces of annotation. As in: Look, TOPCAT has automatically figured out how to plot error bars. Or: Hey, Aladin's epoch slider works, and no guesswork was necessary.
- Validation is meaningful (rather than the "80% failures" situation we show in parts of the Registry). This is because a validator can easily figure out "this dataset can be placed on the sky" if the special annotation is present, while perhaps its product type annotation is broken. With a single, complex DM, it'd just look broken entirely. And, based on the Registry experience: anything as complex as a full annotation of non-trivial data will almost always be broken in some detail. If not now, then after the next software update.
- Separate evolvability: coord1 and coord2 annotation can co-exist indefinitely if necessary, without affecting dataset annotation in any way.
- Isolation and encapsulation are what let us grow our large software systems in the first place.

8. So, We Need to Scrap It All?

No.

There are fewer necessary cross-links in the existing draft DMs than one would think.

Much can be disentangled by just un-typing references (and possibly allowing multiple of them).

But: **We finally need to get buy-in into these models.**

9. History repeats itself

As far as I can tell, the current MCT isn't in use *anywhere*, and very few people have looked at it.

It feels like it's 2007 again, when STC-1 was passed in a mixture of exhaustion and a feeling of unconcernedness.

That's not a way to build a critical component of the VO.

And that's why 20 years into the VO we still can't bring two catalogues to the same epoch. Not even roughly.

10. Proposal

Let's stop the RFC of MCT and instead

- assemble a team of one person per major data center
- that works out an annotation scheme for the data they serve until the next interop,
- meanwhile spelling out exact *use* cases for each model (so it's clear what kind of thing the DM should enable and what's out of scope for now)
- and then use and revise MCT as they go.

This will both fix the decade-old annotation crisis *and* prevent a repetition of the STC-1 non-adoption.

11. An alternative

Or: We return DM to a state of innocence and take it out of operations.

If it's just boxes and arrows, technicalities don't matter.

I'd be in on an effort to make COOSYS cover the epoch transformation use case.