Transient Server

1. Change RAMultiplier to whatever Tom said to -- .0667 or something. Don’t really know what that is used for.
2. Change the Obj\_Type, somehow. For TNS, just make it 60 for Supernova. For AAVSO, don’t know.
3. Figure out how to get VSX to deliver HET listings.
4. Add VOTable converter for ExoPlanets.
5. RA Multiplier: I have to admit that I really just left this in the header (albeit incorrectly) to avoid screwing anything up as I was vomiting out code to reverse engineer the SDB interface. I didn’t really think about it. In retrospect I see it causes a conversion of input RA from decimal degrees to decimal hours (which TSX expects). I will put in the correct value as you suggest.
6. Search Prefix: Consider it done -- I simply had not figured out what the Search Prefix was used for, yet. As with TNS, I will assign the database name as you suggest as a rule: TNS, VSX, MPC, IVOA (ExoPlanet).
7. Obj\_Types: When I (rapid)prototyped the app, I lazily did a direct assignment of the VSX VType to the TSX Obj\_Type, not really knowing what was going to happen. It turns out, as you described, that TSX picked up on the first letter of the VSX VType and used that as its designation for Object Type. AAVSO has what seems like a hundred different VTypes. They have maybe five groupings, but no really hierarchical naming conventions – it’s basically flat, and, to make things worse, they allow groupings of VTypes to describe odd or poorly understood objects, e.g. NM+S. Fortunately, for the nova query, most nova designations start with “N”. Unfortunately, a VSX query requires the URI to spec each VType wanted, so I do a generic query then cull all the non-“N” VTypes. I’m still working on that.

I really do understand the drawbacks of unrestrained Obj\_Type proliferation. I had considered using the TSX “Type X” Obj-Type, but I thought the SuperNova icon was so cool that perhaps more types meant more cool symbols to pick from. But, at the end of the day, TSX has the same problem that AAVSO has: a flat object type structure – no subtypes, or the ability to make subtypes, it seems. Normally, the architectural solution to this type of problem is hierarchical object classes: i.e. Object Type: SuperNova -> Subtype IA, IB, IIA, etc. Object Type: Planet; SubType -> Major, Exo, Minor -> Comet, Kyper, Asteroid -> Jupiter, etc. Of course, even hierarchical object classes have their implementation challenges, e.g. how deep do you allow? All that said, I suspect that’s a huge structural change for TSX and I wouldn’t expect it was even up for discussion. I think whatever you come up with will be just fine. I will stub in a translation class in the app that maps individual database nomenclature to TSX nomenclature. Then I’ll only have to construct a new mapping vector each time a database is added to the fold. That’s essentially what I’ve done for the input parsing: So far there have been only three types of database output to handle: CSV, TSV and VOTable, and the first two are essentially the same (i.e. headers and data). Better yet, first blush on the ExoPlanet database says that it can be output in VOTable; the MPC Ephemeris database in CSV. No worries – happy days.

* Rick