Comparison between the proposed serialisations by Jiri, Francois and Laurent

Under: http://volute.g-vo.org/svn/trunk/projects/time-domain/time-series/
README
data_sample/ Mixed information in xml, dat, vot formats, but this is the data itself which we use for testing the proposed serialisations:

- BetaLyr_Vizier.xml
- GAPS_sample_kp7.dat
- GaiaSample3Tables.vot
- SDSS_J080434.20+510349.2_VizieR.xml
- README.txt —> empty

standardized_votables/

- **Jiri**: two examples, one for BetaLyr and another of a SDSS source
- **Francois**: same two examples, a README file and a TimeSeries-TAPschema.xml
- **Laurent**: same two examples

I start by comparison between the proposed serialisations by Jiri and by Francois.

- BetaLyr_Vizier_jiri.xml VS BetaLyr_Vizier_complete_utypes.xml

Light curve of BetaLyr, one target, several filters, several magnitudes and errors, several times associated to each magnitude
For both serialisations topcat opens four tables, with content as follows:

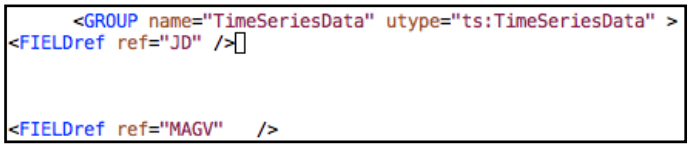
1. Information on the product: product type, calib:evel, pubID, creator, contributor, Target. Both serialisations have the **same information** in this table.
2. Characterisation: **Jiri's serialisation contains information on position and spectral** coverage, while Francois contains information only on the position (RA,DEC)
3. Coordinate system information: both cases are the **same**.
    1. For time we have:
        1. TimeScale: TT
        2. refPositionT: Barycenter
    2. For space we have:
        1. SpaceRefFrame: ICRS
        2. refPositionS: BARYCENTER
    3. For the wavelength, for each filter we have:
        1. wavelength: 1250 (value)
        2. filter: J_BAND (name)
4. light curve it self: both cases are the same

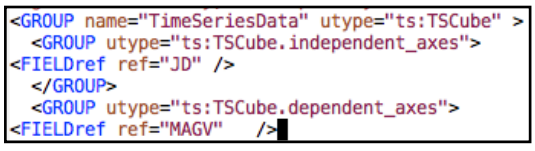
- SDSS_J080434.20+510349.2_VizieR_complete_utypes_jiri.xml VS SDSS_J080434.20+510349.2_VizieR_complete_utypes.xml

One target, one filter.
The only difference with the previous case is that table 2 for Jiri's serialisation has no information on the spectral part.

Main differences in the following utypes:

- ts:TimeSeriesData or ts:TSCube:





Comparison between the proposed serialisations by Jiri, Francois and Laurent

- ts:TimeSeriesData.NDPoint.TimeObservable.TimeMeasure.MJD or stc:TimeMeasure.MJD
- ts:TimeSeriesData.NDPoint.dependantObservedObject.CoordMeasure.PhotometryPoint or phot:PhotometryPoint

```
   <FIELD ID="JD" datatype="double" name="JD" ucd="time;obs.exposure" unit="d"  utype="ts:TimeSeriesData.ND
Point.TimeObservable.TimeMeasure.MJD"  ref="tif">
        <DESCRIPTION>Epoch at midpoint of observation in julian date</DESCRIPTION>
   </FIELD>
        <FIELD ID="MAGV" datatype="float" name="MAGV" ucd="phot.flux" unit="mag" utype="ts:TimeSeriesData.N
DPoint.dependantObservedObject.CoordMeasure.PhotometryPoint" ref="phot" >
```

```
   <FIELD ID="JD" datatype="double" name="JD" ucd="time;obs.exposure" unit="d"  utype="stc:TimeMeasure.MJD
"  ref="tif">
        <DESCRIPTION>Epoch at midpoint of observation in julian date</DESCRIPTION>
   </FIELD>
        <FIELD ID="MAGV" datatype="float" name="MAGV" ucd="phot.flux" unit="mag" utype="phot:PhotometryPoi
nt" ref="phot" >
```

Content of the README file: README_SDSS_complete_utypes.txt
In the file SDSS we adopted the simplest model mapping technic we could imagine.

The model classes are grouped in 4 different packages which are mapped into 4 different tables : one for generic Dataset metadata (dataset direct attributes, dataID attributes and observation attributes), one for characterisation, one for coordinate system and Photometric filters features, and of course the data table.

In order to build utypes for the data and metadata fields in these different tables, we start from the TimeSeries root class and follow the pathes until the leaves attributes. We concatenate the classes and attributes names until we reach the FIELD. This defines unambiguously utypes. They can include pieces from different models mainly the TimeSeries model and imported models such as characterisation, STC, photometry, etc...

IN this approach the role of a FIELD with respect to this IVOA models is defined entirely by the utype string and available at the FIELD level without any depndance from other FIELDS or VOTABLE structures. GROUPs of FIELDS are only there for facilitating reading. This is the simplest method we can imagine in the case the TimeSeries provides for each Time Stamp a measurement consistent with one STC or Photometry axis (or a combination of those). We think that this way applications can infer a lot of feasible actions from the sole parsing of each FIELD attributes values.

- This method will become more difficult to use if all or part of the measurements are represented by a new or non-IVOA model.
- It has been objected that the method may be unstable in the case of changes in the model affecting the names and attributes in the main TimeSeries model or in imported data models. This issue can be partially solved by attaching utypes permanently to columns in a TAP schema. That's why we provide a **TAP schema for a given family of TimeSeries.**

See:
TimeSeries-BetaLyr-TAPschema.xml
TimeSeries-SDSS-TAPschema.xml