# SED Construction

DS5 – Cycle 2 review meeting

*Brice GASSMANN*
*Sebastien DERRIERE*
*Bernd VOLLMER*
*Thomas BOCH*



# Summary

- Scope and goal of SED Construction
- Registry query tool
- Data extraction tool

# Scope and goal of SED Construction



# Scope

- Definition:
  - *SED (Spectral Energy Distribution) Construction provides associations between wavelength and intensity of emission for objects in the sky.*
- SED Construction from existing catalogues are of great interest for astronomers.
- Three steps:
  - Find relevant VO resources according to some restriction like « *I want all resources referring to radio emission* ».
  - Extract relevant and homogenized data from the obtained resources.
  - Merge data at different wavelengths

# The need for a registry

- The registry contains resources metadata that can help us to find relevant resources.
- Excerpt from the registry (tabular resource):

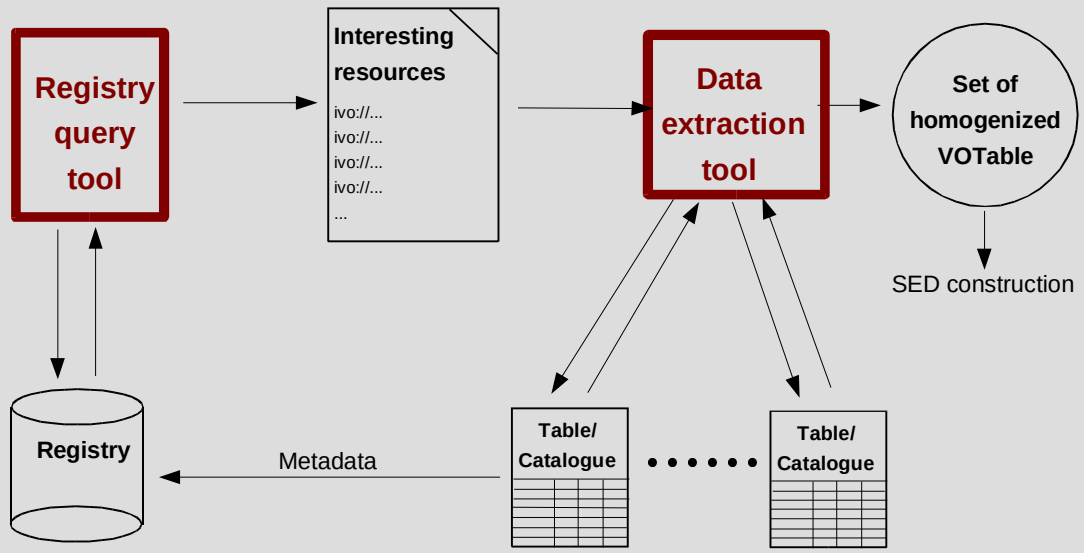
```
<vs:Resource>
  <vs:table xmlns="http://www.ivoa.net/xml/VODataService/v0.5">
    <vs:name>I/239/hip_main</vs:name>
    <vs:description>The Hipparcos Main Catalogue\vizContent{timeSerie}</vs:description>
    <vs:column>
      <name>recno</name>
      <description>Record number within the original table (starting from 1)</description>
      <unit/>
      <ucd>meta.record</ucd>
    </vs:column>
    <vs:column>
      <name>HIP</name>
      <description>Identifier (HIP number) (H1)</description>
      <unit/>
      <ucd>meta.id;meta.main</ucd>
    </vs:column>
  </vs:table>
</Resource>
```



# Our goal

- To perform intelligent resources discovery in the scope of SED Construction
  - Exploits the registry to filter VO resources according to some criteria on its content (wavelength...).
  - Automatically extracts relevant data from the obtained resources.
- To comply with VO standards (as far as possible)
- To make a prototype tool to demonstrate the feasibility

# Overview






# Registry query tool

## *Registry query tool* - Introduction

- A java tool for finding VO resources based on UCDs matching a query
  - Resource -> table -> column -> ucd
- Uses the *XMLDB* API to get data from an XML registry (performing Xquery...)
- Can interact with:
  - A local or remote XML registry (we use Carnivore)
  - Any remote XML database (XMLDB compatible) containing VO resources XML records



## *Registry query tool* – functionalities (1)

- VO resources selection:
  - Finding all VO resources satisfying a logical condition on UCDs
- VO resources description:
  - XML source of the registry record
  - Link to Web page thanks to the *ReferenceURL* element
- VO resources characterization:
  - Input: form for filling characterization data of a VO resource
  - Output: an XML record conforming to the characterization schema is stored locally

## *Registry query tool* – functionalities (2)

- Relevant VO resources mining:
  - Possiblity to pick important VO resources identifiers and store them in a seperate basket
  - The current workspace (list of VO resources, state of the basket, and UCDs request) can be saved and reloaded.
  - A saved workspace can be used in the *data extraction tool* (see next part...)



# Data extraction tool

## *Data extraction tool* - Introduction

- A java tool for extracting homogenized data from various catalogues
- Uses the *XMLDB* API to communicate with an XML registry (performing Xquery...)
- Works with:
  - A local or remote XML registry (we use Carnivore)
  - Any remote XML database (XMLDB compatible) containing VO resources records

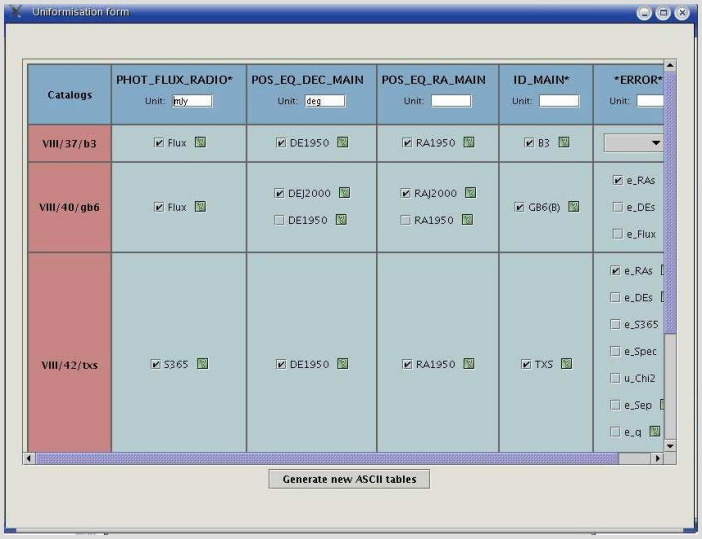


## *Data extraction tool* – Step 1

- Select a list of catalogues
- Select a list of UCDs
  - All columns having such UCDs will be proposed for the extraction
- Or: load a workspace
  - It is possible to load a workspace produced by registry query tool to avoid setting catalogues and UCDs
- Data extraction form
  - A data extraction form is generated for the list of catalogues and UCDs

# *Data extraction tool* – Step 2

- Snapshot of the data extraction form:




- Which columns in the output ?
- Which unit for the columns of a given UCD ?



# *Data extraction tool* – Step 3

- VOTable loading
  - VOTable of each catalogue is loaded thanks to *SAVOT* library
  - The interface of type *ParamHTTP* of the catalogue is used to get the VOTable base URL (registry -> generic). The parameters of the request are for the moment VizieR specific -> It could be generic with future Registry DM for services.
- Unit conversion
  - *cds.astro* java library is used for unit conversion
  - Loaded VOTables are parsed and modified thanks to *SAVOT*

# *Data extraction tool* – Result

- VOTable output
  - Obtain a set of homogenized VO tables, ready for *cross-match* and *SED Construction*