The ALMA Data Mining Toolkit II: Using ADMIT on Data Mined from the ALMA Archive

Douglas N. Friedel[1], Leslie Looney[1], Lee Mundy[2], Marc Pound[2], & Peter Teuben[2]

[1]University of Illinois [2]University of Maryland

Abstract

Following the accompanying ADMIT (the ALMA Data Mining Toolkit) poster by Teuben et al. (P22), we explore the power of the ADMIT schema. ADMIT could be used to data mine across multiple ALMA datasets, projects, or sources, adding value to the ALMA archive. ADMIT will be capable of defining a set of sources in a local virtual project. This project could have the ADMIT analyses recomputed in a common fashion, or to a common level, and add new data products. The sources in the virtual project could also be grouped based on similarities, either by the user or due to decisions made by ADMIT, and have their data abstracted for further analysis and visualization inside or outside of ADMIT in a native Python environment. In this poster we describe the design and techniques of this capability.

ADMIT Products

All products from an ADMIT run are stored in an admit.zip file. These products include an XML file with nodes for all data product members. These members include peak spectra, moment maps, p-v diagrams, etc. The nodes that are for images point to PNG and/or FITS format images of the relevant data products (e.g., moment maps). An example pseudo-XML structure follows:

```
<project name="c01" uid="c01">
   <summary>
      <atask name="task name" parameters ="params"/>
   </summary>
   <source name="Source 1" uid="c01.source_1">
      <ra>20:15:36.25</ra>
      <dec>-22:36:16.7</dec>
      <velocity type="lsr" unit="kms">2.5</velocity>
      <band number="1" uid="c01.source_1.b1">
         <stats type="peak" uid="c01.source1.b1.s1">
            VOTable of peak spectra
         </stats>
         <map type="mom0" uid="c01.source_1.b1.m1">
            <imageURI file="s1.b1.mom0.png"/>
            <atask name="task" parameters="params"/>
         </map>
      </band>
      <band number="2" uid="c01.source_1.b2">
            .
            .
            .
   </source>
   <source name="Source 1" uid="c01.source_1">
      <ra>20:15:36.25</ra>
      <dec>-22:36:16.7</dec>
      <velocity type="lsr" unit="kms">2.5</velocity>
      <band number="1" uid="c01.source1.b1>
            .
            .
            .
      </band>
   </source>
</project>
```

Each of these nodes will have a unique identifier (UID), base on the project name, source, and band. The uniqueness will be not only in a project but across all projects, sources, and bands produced by ALMA.

Virtual Projects

A virtual project is a user or ADMIT defined structure that can span multiple projects, sources, bands, and even other virtual projects. The members of the virtual project can be grouped into one or more subgroups that can be processed in similar manners or to a similar depth.

The virtual projects will have their own XML structure, not saved in the admit.zip file(s), but in a separate XML file. This file will have links to the members of the virtual project. The virtual project will contain nodes for dividing up the individual parts so that they can be grouped by the user or ADMIT, for further analysis.

A sample virtual project XML structure follows:

```
<project type="virtual" uid="user_vp_02">
   <group name="group1">
      <common_item="source"/>
      <member>
         <file name="c001.admit" path="/home/user"/>
         <uid name="c001.source_1"/>
      </member>
      <member>
         <file name="c003.admit" path="/home/bob"/>
         <uid name="c006.source_3"/>
      </member>
         .
         .
         .
   </group>
   <group name="group2">
      <common_item="molecule"/>
      <member>
         <file name="c001.admit" path="/home/user"/>
         <uid name="c001.source_1.b2"/>
      </member>
         .
         .
         .
   </group>
</project>
```

The UID for each project/source/band will be used by the virtual project so that it can link to all associated data with no ambiguity. The link would include both the absolute file name and path and the UID for each member of the virtual project. These links will behave like the Unix hard link, i.e., they will point to the original data members unless a change is made inside the virtual project and the data reprocessed. At this point the links would be replaced by a local version of the data product, leaving the original intact.

The overall virtual project structure will be similar to that of a traditional project, thus to ADMIT it will appear to be just another project and any operation that ADMIT can do on traditional projects can be done on virtual projects as well.

Figure 1 shows an example of how a virtual project could be constructed. There are two groups in the virtual project. The first (blue) associates data based on similar characteristics, which may be the same astronomical source or different sources that the users wants processed identically. The second (red) associates data of the same molecule (i.e., CO or HCN) from several different sources, again to all be processed in the same way.

What can be Done with Virtual Projects?

The purpose of the virtual project is to allow the user to group data together, based on some similarity (source, molecule, distance, etc.), and have the associated members processed in a similar manner (same depth, same noise level, etc.). This will allow for easier comparison and/or combining of the different data. The products of the reprocessing will be exportable (FITS or PNG format) for further analysis in outside packages. The data may also be available as numpy arrays for analysis in Python or casapy.

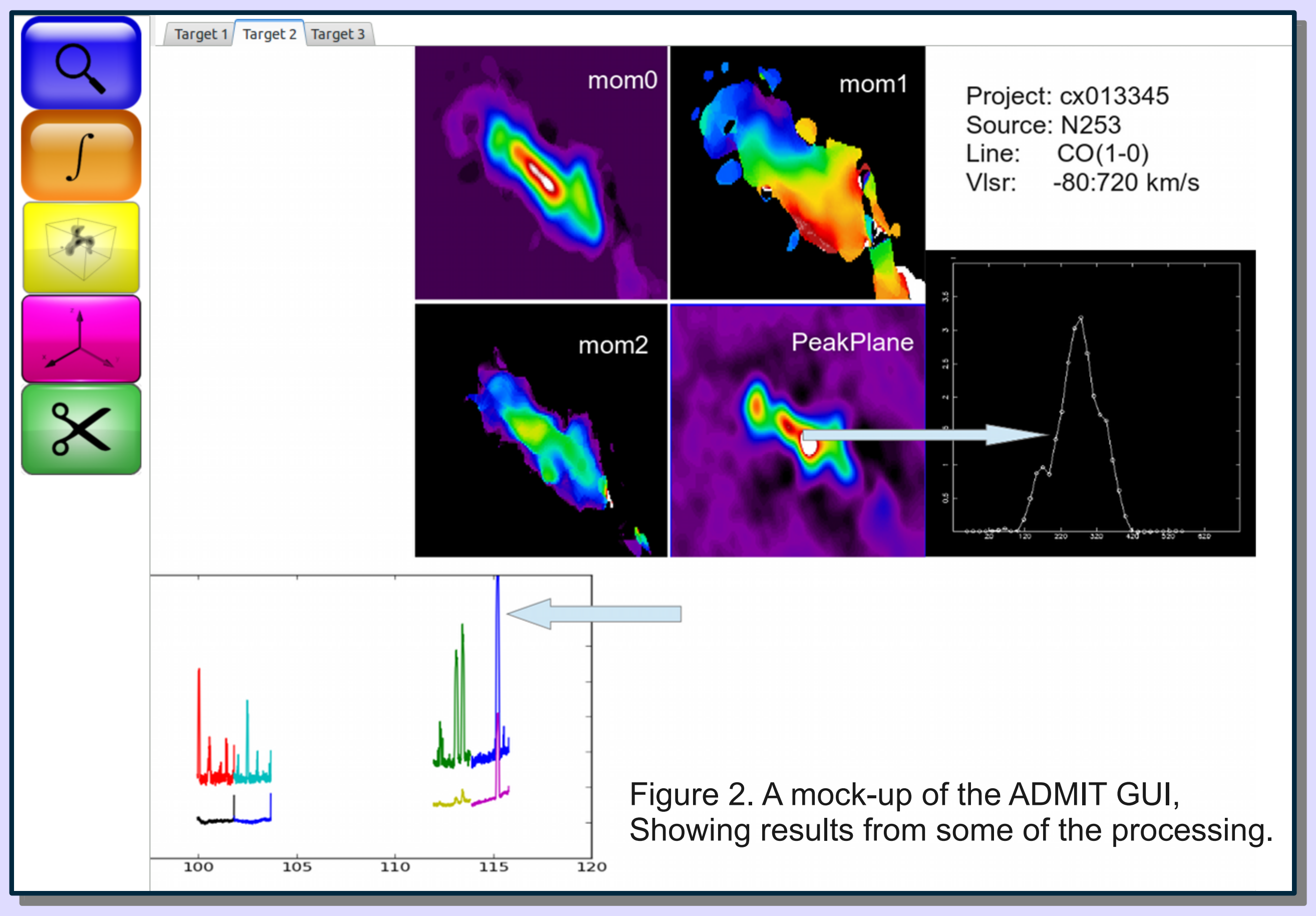
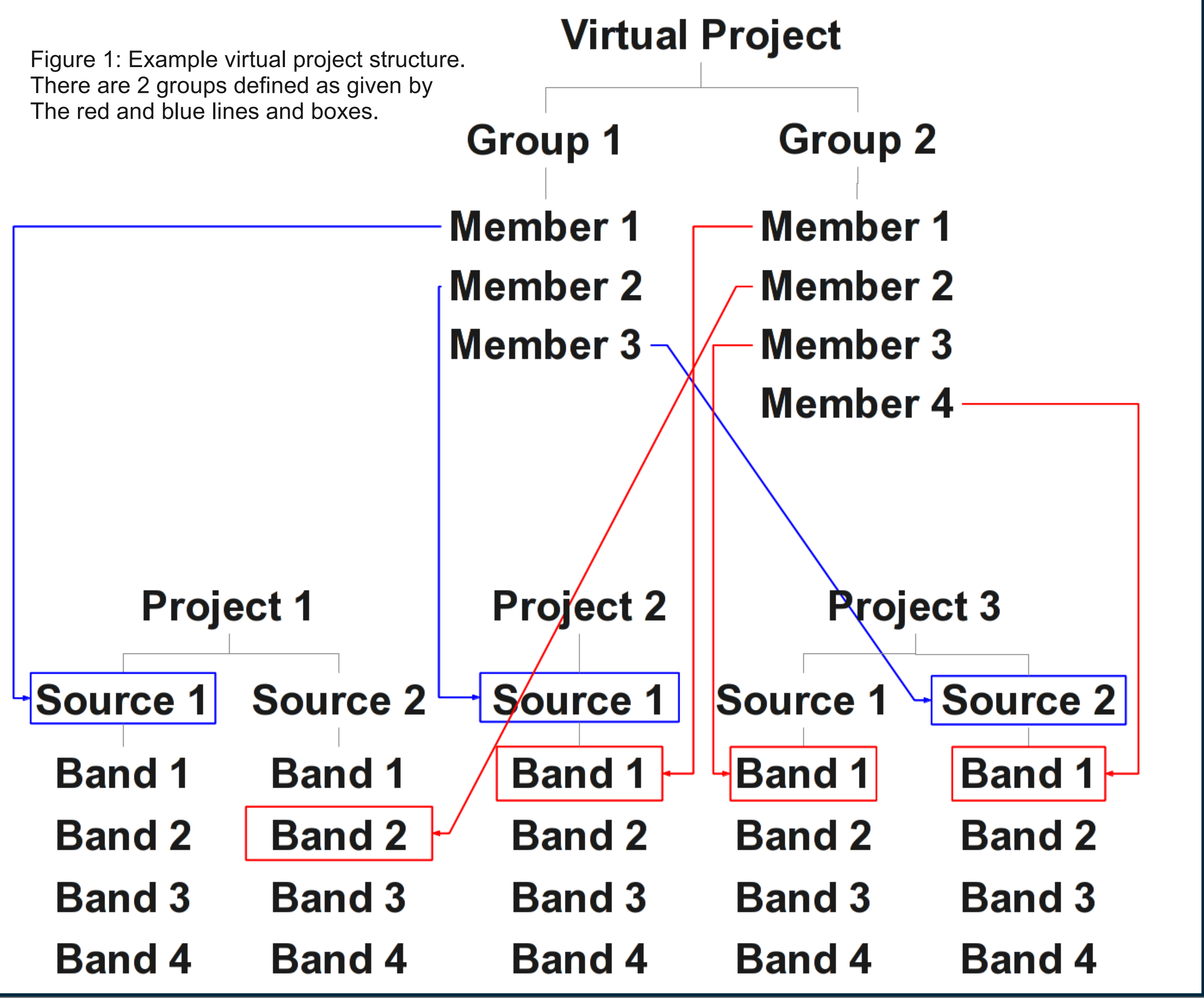


Figure 2. A mock-up of the ADMIT GUI, Showing results from some of the processing.

How are Virtual Projects Created?

The user will be able to use a menu item from the ADMIT GUI to create a virtual project. Figure 2 shows a mock-up of the ADMIT GUI. Once created, the user can add groups to the virtual project and members to the groups, based on their research goals. The creation of the virtual project will create an object in memory that links all the associated members together. The virtual project can be saved in its own admit.zip file for future re-opening and processing. Since the typical virtual project contains links to the members, and not the actual files themselves, or a mixture of links and files, there will be an option for the user to export the virtual project in its entirety to an admit.zip file. In this instance all links would be followed and copies made of their destinations in the output file. This export file would be fully self-contained and portable.

Figure 1: Example virtual project structure. There are 2 groups defined as given by The red and blue lines and boxes.



Credit: ALMA (ESO/NAOJ/NRAO)

Douglas Friedel
University of Illinois
Department of Astronomy
friedel@astro.illinois.edu
www.astro-chemistry.com