

User Defined Functions for the ISO Post Mission Archive

Ekkehard Wieprecht

*Max Planck Institut für extraterrestrische Physik, Giessenbachstrasse 1,
D-85748 Garching, Germany*

Bart Vandenbussche

*Katholieke Universiteit Leuven, Instituut voor
Sterrenkunde, Celestijnenlaan 200 B, B-3001 Heverlee, Belgium*

Erich Wiezorrek, Helmut Steinle, Otto H. Bauer

*Max Planck Institut für extraterrestrische Physik, Giessenbachstrasse 1,
D-85748 Garching, Germany*

Christophe Arviset, Jose Hernandez, Arno M. Plug, Richard Saxton

ISO Data Centre, ESA SSD, Villafranca, 28080 Madrid, Spain

Abstract. A major problem with designing archives in astronomy is defining the data resolution. In most cases, the smallest element in an archive is the object “observation”. Attributes of this object contain meta-information about the observation that should be quickly queryable and are thus stored in the database engine. The data from the observation are stored external to the database engine as one big object. During the design of the archive, it is often difficult or even impossible to anticipate what meta-information of an observation should be queryable.

We present a solution for this problem as adopted for the Post Mission Archive of the Infrared Space Observatory (ISO) mission. Expert users can create User Defined Functions (UDFs) that extract virtual attributes from files external to the database engine. These virtual attributes can then further be used in queries inside the database engine.

We discuss the concept of User Defined Functions, system design, and implementation.

1. The ISO Post Mission Archive

During ISO’s (Kessler et al. 1996) operational period (November 1995 to May 1998), its four instruments obtained a wide variety of astronomical data at infrared wavelengths from 2.4-240 microns. About 25,000 observations are stored in the ISO Post Mission Archive (Saxton et al. 1998). This archive provides on-line access to the data, support documentation, and software through a WWW interface.

The user interface is HTML and JAVA based and issues SQL queries to the database via jConnect¹ for JDBC. The basic interface allows for selection of observations defined by standard criteria such as coordinates, source name, observing mode, exposure time, wavelength, etc. Users can also select data by engineering, trend, and housekeeping parameters.

2. Virtual Data Resolution with User Defined Functions

As with all astronomical databases, data resolution is a critical issue in the ISO Post Mission Archive. The smallest object in the database is often an “observation” which is stored as a file external to the Database Management System (DBMS) or as a Binary Large Object (BLOB) in the DBMS. Only meta-information such as coordinates, average flux, pointing, etc., and relations to other objects like observer, instrument, etc. are stored and queryable (SQL, OQL, etc.) inside the DBMS.

For many instrument analysis and scientific purposes, queries and correlations on a much smaller scale are needed, for example, finding trends in detector noise for a specific instrument status, measuring line fluxes of one particular line of many sources in a vast set of broadband spectra, or comparing dark current with glitch activities. Anticipating all queries that will be needed during the design phase of the database is impossible. A scheme where all relevant data are implemented as attributes in the database is therefore impossible. This need could be accommodated in a database system where not only meta-data is stored in a queryable form, but also the data proper. In the case of spectra, a table containing wavelengths and fluxes would then be defined in the database.

Present “off the shelf” RDBMSs (Relational Database Management Systems) seem not to be robust enough to handle typical sizes of present astronomical databases. Therefore, for the ISO Post Mission Archive, the concept of virtual data resolution with user defined functions has been designed and implemented as a novel database paradigm.

The traditional division is kept where only meta-information about observations and related files is stored in the relational database; the data files themselves are stored external to the database. But the user is allowed to define functions which extract attributes from any data file in the archive. This includes the possibility of running Unix commands (sed, awk, grep, etc.), TCL, or Perl scripts on the ASCII files. Also, Interactive Data Language (IDL²) functions can be run on the FITS files. Therefore, it is possible to use features provided by the instrument interactive analysis systems which are coded in IDL.

¹jConnect is a trademark of Sybase Inc.

²IDL is a trademark of Research Systems Inc.

3. System Design

The definition of a User Defined Function is entered in the system via a web interface. A UDF takes one file as input, optional additional input parameters, and outputs the virtual attributes for that file.

A defined UDF can be executed on a number of data files. The list of data files on which the UDF has to be run and the additional input parameters are prepared by the user in a temporary database table. The UDF is executed on every filename in the table and the extracted virtual attributes are stored in a second table. The second table can be either inserted again in the database where it can be used for further queries or it can be e-mailed to the user.

4. Implementation - How the ISO PMA User Uses UDFs

If an expert user needs additional attributes of files (e.g., observations) that are not provided by the standard RDBMS scheme he might use UDFs. The interface to define and execute user defined functions is web-based. Figure 1 gives an overview of the interface structure. In order to take advantage of UDF processing, the user needs a UDF user identification and password.

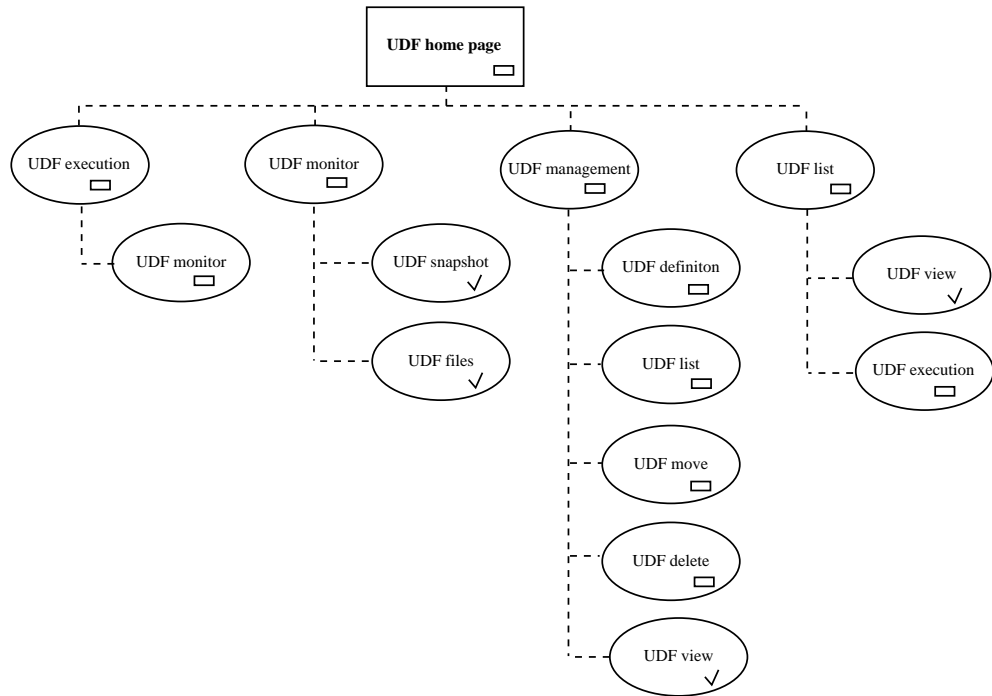


Figure 1. UDF web pages.

The UDF list page gives an overview of existing UDFs in the public and private areas. But the user can also write his/her own new UDF and install it in his/her private area on the server side. Further specifications of the UDF (e.g., input parameters) have to be given using the UDF definition page. This specifications can be displayed by the UDF view page. If a UDF is stable and useful to the community, it can be moved from the private UDF area to the public area by the UDF manager using a web page interface, UDF `move`.

The UDF execution page is used to start the UDF process. The user must specify the fixed input parameters. He also must specify if he wants to select required files via an SQL statement creating a temporary RDBMS table or use an existing database table. The required files are ordered to guarantee most efficient file access and are copied to the work directory, taking into account internal database limitations. A UDF works on one file at a time. The result (table with filenames and virtual attributes) is available as a temporary RDBMS table, which can be used for further SQL processing or sent via e-mail. The UDF process can be monitored using the UDF monitor page where the user might get additional information such as a snapshot of the result or which files have been copied from the archive at a given time. The UDF manager can monitor all the processes and start, stop, or even kill them.

5. Conclusion

We presented concept, design, and implementation of User Defined Functions on the ISO Post-Mission archive. We showed that UDFs allow users to query the archive using virtual attributes, some of which cannot always be anticipated during the archive design phase.

Acknowledgments. The ISO Post Mission Archive (ISO PMA) was developed at the ISO Data Centre, ESA Astrophysics Division, Villafranca del Castillo, Spain. Bart Vandebussche acknowledges financial support from the Belgian Federal Services for Scientific, Technological and Cultural Affairs and from the Onderzoeksfonds K.U.Leuven grant OT/94/10

References

- Kessler, M. F., et al. 1996, *A&A*, 315, L27
- Saxton, R. D., et al. 1998, in *ASP Conf. Ser.*, Vol. 145, *Astronomical Data Analysis Software and Systems VII*, ed. R. Albrecht, R. N. Hook, & H. A. Bushouse (San Francisco: ASP), 438