Astronomical Data Analysis Software and Systems XIII
ASP Conference Series, Vol. 314, 2004
F. Ochsenbein, M. Allen, and D. Egret, eds.

The Common Pipeline Library - a silver bullet for standardising pipelines?

Klaus Banse, Pascal Ballester, Carlo Izzo, Yves Jung, Lars K. Lundin, Andrea Modigliani, Ralf M. Palsa

European Southern Observatory

Derek J. McKay [1]

Rutherford Appleton Laboratory

Michael Kiesgen

Michael Bailey Assoc.

Cyrus Sabet

Tekom GmbH

Abstract. The data from instruments of the VLT are automatically processed for standard basic data-reductions and artefact removal. Currently, such pipelines are operational for seven instruments and have been written on the basis of three different software packages. Since 2001, the Data Flow System group of ESO has been working on a Common Pipeline Library (CPL) which is based on code already developed and used at ESO and shall serve as the basis for all future instrument pipelines of the VLT. The main goals for the CPL are to streamline the code for different pipelines, to provide templates for standard algorithms, to support reuse of code, and to ease the maintenance and portability of the code.

We describe the roots of the CPL code, the development process of the CPL and the problems we had to overcome to reach the current stable beta-release of CPL.

1. Introduction

As major components of the Data Flow System (DFS), used to operate the Very Large Telescope (VLT), the pipelines are operation-critical. They must run in an efficient and stable manner to guarantee high-quality data output needed for checking the quality of the observations and monitor the health of the instruments. They are also instrument-specific reduction tools and thus their number keeps growing with each new instrument installed at Paranal. At the time of writing, observational data from seven instruments are automatically processed

[1]and *European Southern Observatory*

by their individual pipelines. The reduction software used in the pipelines was written either at ESO, or outside by the instrument consortia. As a result, the pipelines currently in operation have been written on the basis of three different software packages, use different interfaces to the pipeline infrastructure and, not surprisingly, contain considerable overlap and duplication of functionality. The number of instruments commissioned at the observatory continues to increase, and the operation and maintenance of these pipelines is the responsibility of ESO with essentially the same number of staff and resources.
As a solution to that dilemma, the concept of a library of the reduction software which is needed for typical tasks of pipelines was formed: the Common Pipeline Library (CPL). Such a common software library should greatly help in solving the problems mentioned above - our silver bullet. It would standardize the two main issues of instrument pipelines: writing the data reduction tasks used (the *recipes*), and interfacing these recipes with the pipelines.

2. The Common Pipeline Library

The concept of the CPL was first proposed in 2001 and since then a team inside the DFS group of ESO's Data Management Division (DMD) has been working on the project. The CPL design and development could not start from a clean slate due to

- a lack of resources - the pipelines for all the operational VLT instruments had to be supported in parallel, entailing extensive trips to Paranal (absences) of key members of the CPL group
- a tight deadline because of the rapid arrival of new VLT instruments and political pressure to deliver such a tool to the instrument consortia in time (CPL will be mandatory for pipeline code production in the future)
- lots of data reduction software for pipelines had already been written in the DFS group - we should not reinvent the wheel...

Therefore, we did not follow the usual cycle of requirement analysis, design, review, etc. for the CPL code. Instead, the CPL should be based on code already developed at ESO which was well tested and used for existing, operational pipelines. In particular, the *eclipse* library (used for ISAAC and NACO pipelines) and some of the concepts of the *VIMOS data reduction* software would be the main pillars of the CPL software.

2.1. Requirements

The CPL would provide all the functionality needed for building the data reduction tasks of pipelines for VLT instruments. It would not be oriented towards an interactive data analysis system, but should be optimized for automatic, batch oriented, pipeline processing. The main users of the CPL would be the groups building pipelines for the VLT, i.e. the DFS group of DMD at ESO and the instrument consortia, and not the general user community. Furthermore, we would only implement concepts which had proven to be commonly used in the operational pipelines so far. Also, it was assumed that the CPL user community, experienced application programmers of instrument pipelines, were knowledgable in writing C-code and familiar with object-oriented (OO) concepts.

2.2. Goals

The CPL is intended to serve as the basis for all future instrument pipelines of the VLT, i.e. in particular for the second generation VLT instruments. By using a standard set of components (controlled by ESO) the operation of CPL based pipelines should become more uniform and robust, and, especially, the maintenance of them should be simpler and more efficient, independent of the original authors of the code - whether they be ESO or outside consortia.
The first pipeline to use CPL would be the GIRAFFE pipeline, and thus serve as the testbed for the CPL implementation and its usability. Existing pipelines would be converted on the basis of available resources in parallel with the actual development work. The CPL shall be released in December 2003.

2.3. General Architecture

All the instruments on ESO telescopes (Paranal, La Silla) produce their data in FITS format. Consequently, the internal data format of the CPL is also FITS, and CPL accesses FITS data files via the *QFITS* library which provides all FITS related functionality needed for pipeline processing. The QFITS software library has been developed at ESO, is specifically tailored to the needs of the VLT pipelines, and adapted to the DICB concept of ESO (i. e. standard, ESO defined dictionaries for the FITS keywords of all data files produced by VLT instruments). Chosing QFITS for the CPL gave us full control of the I/O software, facilitating enormously any changes and updates to the FITS I/O code which became necessary due to the CPL development work. The QFITS source code is available independently of CPL as a stand-alone FITS library[1].

CPL software is written in standard ANSI-C, employing object-oriented concepts, with emphasis on platform independency and portability. The code is maintained using CVS and distributed as "tarfiles" at regular intervals, and for the generation of the documentation (on-line help, reference manual) we employ the *doxygen* package.

2.4. CPL Data Objects

CPL is streamlined for smooth execution of standard tasks used routinely in the operational pipelines at the VLT. The software supports all the data types as CPL Objects (in the style of C++) which were commonly needed in the instrument pipelines, e.g.:

- Images, Vectors - 2-dim, 1-dim data arrays with image information
- Tables - structures containing non-homogeneous data (of almost any imaginable type) arranged as rows and columns
- Matrices - matrices in the usual mathematical sense
- Property Lists - containers for ancillary data of a CPL object

Thus, for example, a recipe for applying a filter to an image would first create a *CPL-Image* object and then, using the returned pointer, fill that object with data from the relevant FITS file and finally process (filter) the CPL-Image structure.

[1]`http://www.eso.org/eclipse/qfits`

2.5. CPL accessible Functions (APIs)

The CPL does not just provide interfaces for accessing all data structures (e.g. like ESO-MIDAS) but goes farther and contains already all the functions needed to support the basic operations typically needed in pipeline processing. Available functions of the CPL are e.g.:

- standard arithmetic operations, including basic mathematical functions, *log, exp, ...*
- image processing operations, *rotation, extraction, insertion, ...*
- table operations, *row selection, merging, combining, ...*
- statistical calculations on CPL objects, *images, tables*

Higher-level standard astronomical functions related to pipeline operation, like *standard flat fielding, standard bias subtraction* are planned for future releases of CPL.

2.6. CPL Infrastructure

The CPL also provides a framework for executing recipes in a standardized way via dynamically loadable software modules (*plugins*). The concept of plugin interfaces helps in

- providing a standard way how pipeline recipes are implemented
- hiding the details of the recipe's runtime environment from the recipe
- avoiding re-coding of tasks common to each recipe and pipeline (e.g. command line parsing)

Requiring recipes to be implemented as Pluggable Data Reduction Modules (PDRM) enforces a certain coding standard which guarantees that software developed by external consortia can be faster integrated into an ESO pipeline and validated. It also reduces the need for in depth knowledge of the recipe implementation right from the beginning. Furthermore, all recipes will have the same look and feel across different instrument pipelines.

Once a given recipe complies to the plugin interface it can be executed by an external application which takes over tasks like command-line parameter processing, collecting input data and distributing data reduction products.

3. The Silver Bullet?

Merging two different software packages into a unified library proved to be more difficult and cumbersome than expected. The *eclipse* library and the ESO software written for VIMOS were built with quite different ideas about how such code should be written and were also in different stages of development. A very pragmatic and flexible approach favouring code production which was very user responsive, quickly updated and easily accessed by outside users contrasted with a more formal and static, but thus traceable, way of building up software to be delivered to outside users assuming full responsibility for that code. We had to make many compromises, cutting edges and abandoning high flying objectives in order to get the CPL out in time.

The acceptance of CPL by the instrument consortia will determine if this library will solve most of the problems we're currently facing with the rapid increase of pipelines which have to be supported on Paranal.