The Common Pipeline Library — Standardising Pipeline Processing

Derek J. McKay[a,b], Pascal Ballester[a], Klaus Banse[a], Carlo Izzo[a], Yves Jung[a], Michael Kiesgen[a,c], Nick Kornweibel[a], Lars K. Lundin[a], Andrea Modigliani[a], Ralf M. Palsa[a] and Cyrus Sabet[a,d]

[a] European Southern Observatory, Karl-Schwarzschild-Str. 2, 85748 Garching, Germany
[b] Rutherford Appleton Laboratory, CCLRC, Chilton, Didcot, OX11 0QX, United Kingdom
[c] MBA Michael Bailey Associates, Schackstr. 1, 80539 München, Germany
[d] Tekom GmbH, Gautinger Str. 29, 82152 Krailing, Germany

ABSTRACT

The European Southern Observatory (ESO) develops and maintains a large number of instrument-specific data processing pipelines. These pipelines must produce standard-format output and meet the need for data archiving and the computation and logging of quality assurance parameters. As the number, complexity and data-output-rate of instrument increases, so does the challenge to develop and maintain the associated processing software. ESO has developed the Common Pipeline Library (CPL) in order to unify the pipeline production effort and to minimise code duplication. The CPL is a self-contained ISO-C library, designed for use in a C/C++ environment. It is designed to work with FITS data, extensions and meta-data, and provides a template for standard algorithms, thus unifying the look-and-feel of pipelines. It has been written in such a way to make it extremely robust, fast and generic, in order to cope with the operation-critical online data reduction requirements of modern observatories. The CPL has now been successfully incorporated into several new and existing instrument systems. In order to achieve such success, it is essential to go beyond simply making the code publicly available, but also engage in training, support and promotion. There must be a commitment to maintenance, development, standards-compliance, optimisation, consistency and testing. This paper describes in detail the experiences of the CPL in all these areas. It covers the general principles applicable to any such software project and the specific challenges and solutions, that make the CPL unique.

Keywords: Common Pipeline Library, Very Large Telescope, Data Flow System, software

1. INTRODUCTION

Pipelines, in the context of this paper, are those collections of software used to perform automatically a large range of data reduction tasks. Effectively, they provide an initial preprocessing of telescope data, before it is even perused by the recipient astronomer.

As major components of the Data Flow System (DFS) used to operate the Very Large Telescope (VLT) and VLT Interferometer (VLTI), the pipelines are operation-critical. They must run in an efficient and stable manner to guarantee the high-quality data output needed for checking the quality of the observations and to monitor the health of the instruments. They are also instrument-specific reduction tools and thus grow with the number of new instruments installed at ESO's observatory at Paranal.

Further author information: (Send correspondence to D.J.M.)
D.J.M.: E-mail: dmckay@eso.org, Telephone: +49 89 3200 6548

2. MOTIVATIONS

Traditionally, each telescope instrument has seen the development of new pipeline software. However, with the rapid increase in the number of supported instruments, there has been the curse of duplicated functionality; the same algorithms and standard operations being repeated time and time again. At the time of writing there are, for example, 7 VLT and 3 VLTI instrument pipelines, all of which perform similar calibration and data reduction tasks. These tasks include, for example, flat-field, bias and dark-frame corrections, artefact removal, wavelength calibration, and so on.

Ideally, there would be a single set of generic library routines to accomplish all of these tasks, and to do so in an efficient, optimised and extensible manner. It would also address the issue of "robustness" and, with wide-spread and varied use, would receive rapid and comprehensive testing under varied conditions and loads. That would make the resultant software reach stability much more rapidly, as well as allowing it to adapt quickly to the changing astronomical computing environment. Ultimately, the CPL would provide a standard template for the development of algorithms, and it would be designed to work with FITS data, extensions and meta-data, by using the QFITS library.

Furthermore, it was envisaged that it would be ideal if all pipelines could be run within a common environment, perhaps even as part of a single executable "master process". However, the challenge was that instrument data reduction software is written by a plethora of different consortia, and to provide uniform interface control would be problematic.

Additionally, the provision of a unified code-base for such tasks would meet the growing challenge of the evolving VLT. It would cope with the increasing burden of exponentially growing data output and would accommodate the need to reprocess data, based on new telescope and instrument calibration models.[7] It would also be capable of expanding to support the growing number and complexity of VLT and VLTI instrumentation, while at the same time easing the liability of rising software maintenance and development costs.

3. HISTORY OF THE PROJECT

The concept of the CPL was first proposed in 2001 and a team within the DFS group of ESO's Data Management Division (DMD) undertook the project.

Despite the limited resources, mostly due to the ongoing support requirements of the operational VLT instruments, the project was brought to a successful conclusion, with the public release of the CPL in December 2003. As such, the CPL was released on-time and within budget.

While the code itself was developed anew, the algorithms were taken from the successful VIMOS and Eclipse (used for ISAAC and NACO pipelines) projects, and were thus already proven. Of course, members of the CPL team had worked on both the VIMOS and Eclipse in the past, and it was possible to draw on the considerable experience gained through these projects.

The timely completion of the primary release of the CPL has allowed ESO to keep apace with the tight deadlines imposed by the onslaught of new VLT instrument commissioning. Additionally, it meets the political pressure to deliver pipeline development tools and resources to the instrument consortia (it is anticipated that the use of CPL will become a requirement in future pipeline code production).

The first official release of CPL (version 1.0) was announced in December 2003. A link on the ESO-DMD home page guides users to the official CPL website of ESO (`http://www.eso.org/cpl/`) with further information about the CPL and from where the CPL software can be downloaded.

A first patch (1.0.1) of CPL has been released in May 2004, containing some bug fixes and one updated function. CPL 1.0.1 is now in use at the VLT for the operational VIMOS pipeline as well as for the GIRAFFE pipeline.

For the VISIR recipes (and for NACO), additional CPL-functionality was needed. Together with enhancement of the original design, these modifications led to a major CPL update, justifying a new release, 2.0.

Final preparations and tests of CPL version 2.0 are under way with the aim of releasing CPL 2.0 in July 2004, when it wil be needed for the commissioning of VISIR and its pipeline at the VLT on Paranal.

4. THE DEVELOPMENT PROCESS

The CPL was developed by the DFS with the manner in which the DFS supports the science operation model of the VLT firmly in mind. One of the primary development goals was to maximise the efficiency of the software being written at a global level and to ensure a high and predicatable data quality level. Algorithm integrity was of paramount importance.

The CPL has been written using an object-oriented approach wherever applicable, however still using the ISO/IEC 9899:1999(E) C coding standard (with POSIX extensions). Basic data types (images, tables, matrices, etc.) are provided, with composite data types built upon these so-called primitives. Although a C library, the CPL can and does operate as calls from the C++ environment as well, making it suitable for use in applications such as GUIs or databases, rather than the structured programming environment of linear processing.

Standard code management techniques (CVS version control, Doxygen code documentation, etc.) have also been used. Internal coding standards and naming conventions have made the library easy to maintain and easy to anticipate (from the point of view of an external user). The system has been used under multiple platforms (x86 Linux, Alpha-64 Linux, Solaris, HP-UX, Mac OS X, etc.) to ensure portability, not only between platforms, but also to guard against compiler upgrades as well.

The CPL serves as the basis for all future instrument pipelines of the VLT. By using a standard set of components controlled by ESO, the operation of CPL based pipelines has become more uniform and robust. In particular, the maintenance of them is now much simpler and more efficient, independent of the original authors of the code — whether they be ESO or outside consortia.

Through the CPL, all the functionality needed for building the data reduction tasks of pipelines for VLT instruments has been provided. It is not oriented towards an interactive data analysis system, but has been optimised for automatic, batch oriented, pipeline processing. The main users of the CPL are the groups building pipelines for the VLT, namely the DFS group of DMD at ESO and the instrument consortia. Note that this excludes the general user community. Furthermore, only concepts which are common to the pipelines are actually implemented, thus curbing code bloat. It was also assumed that the CPL user community — experienced application programmers of instrument pipelines — were knowledgeable in writing C-code and familiar with object-oriented concepts.

In terms of testing, DFS has embarked upon an ambitious testing programme, which involves writing test software to automatically check functions within the CPL. The CPL Validation Software (CPLVS), which provides rigorous testing of all aspects of the CPL, is part of the quality assurance programme. The CPLVS provides four levels of automated testing: General Testing, In-Depth Testing, Load Testing and Performance Testing.

The *General Testing* (function coverage) aims to ensure that all functions that are documented by the provided CPL documentation are called at least once during normal testing conditions. This ensures that the provided library is "complete" (that is, all advertised functions are provided as stated) and hence validate the API.

The *In-Depth Testing* (full code-path) extends this, by calling CPL functions with known success and failure cases, thus testing all possible execution branches. Expected errors are handled. This is the most comprehensive level of testing, but it is also the most time consuming to design, implement and test itself.

The *Load Testing* (maximum resource) refers to either calling the CPL functions an arbitrarily large number of times, or by calling the functions with extremely large data units. This tests the ability of the CPL library to cope under heavy operating conditions. Most of this has been provided for within the infrastructure of the CPLVS, and so can be implemented later, making use of the existing General and In-Depth test functions.

The *Performance Testing* (benchmarking) provides a means by which the execution times for CPL testing functions may be assessed. This functionality is a part of the CPLVS testing structure. This aspect of the CPLVS has already been used to test, determine and demonstrate performance increments resultant from subsequent modifications to the library.

5. FEATURES OF CPL

The CPL itself is split into three major components, reflecting the algorithmic level of the software contained therein. These components are:

- The CPLCORE library provides the fundamental CPL data types (such as `cpl_image`, `cpl_table`, `cpl_vector`, `cpl_matrix` etc.), the operations defined on these data types, and elementary utility functions. There should not be any physical units associated with any of the data types or functions of that library.
- The CPLDRS (DRS for Data Reduction System) uses the CPLCORE data types and functions to implement higher level data processing algorithms (such as wavelength calibration, image recombination, dark-frame correction and so on). The physical approach of this library in in contrast to the mathematical approach of the CPLCORE library.
- The CPLUI (UI for User Interfce) provides services defining the standard interface for recipes and provides more complex data reduction related utilities and services. In particular, it contains the necessary types to handle the plugin interface (see Section 5.2), the parameters used to handle command-line options and the (set of) frames used to store the input list of files.

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

- Images — not only the image structures, but also load/save/copy functions, logical and morphological operations, bad-pixel handling, processing operations and a plethora of other functions
- Vectors, Bi-Vectors — arrays of data or data pairs, with arithmetic, manipulation and statistical operations
- Tables — structures containing non-homogeneous data arranged as rows and columns with statistics, sorting, and access routines
- Matrices — in the usual mathematical sense, complete with a large selection of accompanying operations
- Property Lists — containers for ancillary data of a CPL object

5.1. CPL accessible Functions (APIs)

The CPL not only provides interfaces for accessing all data structures but also all the functions needed to support the basic operations typically used in pipeline processing. Functions of the CPL include:

- 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*, etc., are planned for future releases of CPL.

5.2. CPL Plugins

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

- providing a standard pipeline recipe implementation interface,
- hiding the details of the runtime environment from the recipe itself and,
- avoiding re-coding tasks common to each recipe and pipeline (e.g. command-line parsing).

Requiring a recipe to be implemented as a plugin — or Pluggable Data Reduction Module (PDRM) — enforces a uniform environment that ultimately results in code that is not only easier to integrate into the ESO Data Flow System, but which is also faster to develop in the first place. 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 — this uniformity was a large motivation in developing this system.

Once a given recipe complies with 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.

This plugin concept is a key point in the CPL design. The fact that the recipe algorithm can be detached from the executable environment means that additional algorithms may be added, in real-time, to the VLT data reduction system, without the need to terminate and restart existing data processing operations.

It is quite clear, from the experience that the DFS group has gained using this system, that this approach provides major advantages over previous software. These include the ability to create a unified recipe-testing environment, which dramatically reduces the load on testing algorithms, thus permitting even more testing — in terms of both quality and quantity — to be performed.

It also gives a simpler development environment, as the software engineers who are responsible for writing recipes can now concentrate on the algorithms of the data reduction itself, with all the infrastructure and data handling being provided. Figure 1 shows the relationship of components within a pipeline recipe system, indicating the well-defined area that concerns the actual recipe developer. That is not to say that such things are not accessible, and it is very likely that additional host software will be written to accommodate the different working environments. The benefit is, however, that when these applications are written, they will be able to execute the existing recipes without modification.

6. INTEGRATION

Like most real projects, there is usually a need to continue using existing systems before bringing the new development work into operation. One of the pleasing results from the CPL project is the ease with which this is being accomplished. Therefore, in addition to the inclusion of CPL into new projects, there is also the conversion of existing pipeline recipe software to incorporate the CPL. ESO believes that this will be a worthwhile investment, providing increased reliability to existing pipelines, and permitting easier long-term maintenance.

6.1. Promoting CPL in the pipeline community

In addition to retrofitting the CPL to existing pipelines and incorporating it into new developments within ESO, it has also been important to promote its use within the pipeline development community. This community comprises the consortia that build and deliver instrumentation to ESO as part of ongoing development contracts.

While ESO can certainly mandate that CPL be used, we wish to go beyond that, and provide the necessary support to make consortia not only appreciate the benefits offered by the new software development tools, but also to actively wish to incorporate them into existing and subsequent programmes, regardless of any contractual requirements.

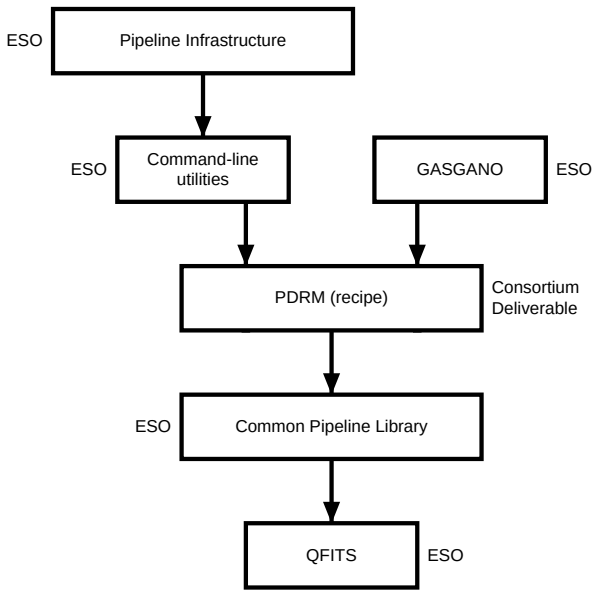

Figure 1. Instrument Pipeline Software Structure. This figure shows the execution dependency of the major components within the recipe operating environment. Note that the consortium that actually provides the recipe has no need to contribute any of the supporting infrastructure, and can thus concentrate on the algorithms themselves.

To this end, ESO provides extensive support for CPL, over and above the normal web-based and paper documentation that accompanies most software projects. Additionally, a recipe template is available, which allows developers to avoid concerning themselves with the infrastructure and have an operational hull within minutes, and be implementing algorithms within hours.

Additionally, ESO provides tools to run recipes in either a GUI or scripting environment (see Section 7).

Careful management for portability and multi-platform support has ensured that the CPL will be adaptable to the wider community, and will ease the transition to newer compiler and operating system regimes. Such techniques also permit the possibility for strict static and run-time checking using external tools, thus maximising testing and code-integrity — a goal that ultimately results in ease of use for end users.

ESO is committed to maintaining a development programme for CPL, and a carefully managed progress plan has been devised to incorporate common algorithms from developed recipes, as well as to address new calibration technologies as they emerge from the developments within the astronomical community, whether they be adaptive optics, interferometric analysis or something else.

7. CPL TOOLS — THE NEXT LEVEL UP

As already mentioned, one of the key concepts of the CPL, in terms of developing recipes, is that of PDRMs. These abstract the algorithmic engine from the execution environment. To illustrate the variety that is already available in the latter of these, two such host applications are described forthwith.

7.1. Gasgano

Gasgano is a GUI software tool for organising and viewing data files produced by VLT Control System (VCS) and the Data Flow System (DFS). The main purpose for developing it was to provide a user-friendly tool capable of dealing with the large amount of data generated by the VLT and the other ESO astronomical facilities.

The functionalities offered by Gasgano can be summarised as follows:

Data Grouping/Sorting: Gasgano groups files located on a specified set of directories automatically into a tree structure, based on the observing run and observation blocks corresponding to the files. Users have the option to change the "virtual view" by first grouping their data around the directory in which they belong and/or by defining instrument specific groups. Note that Gasgano is capable of recognising raw files compared to pipeline products, and displaying them differently using labelling and colour. Furthermore, instrument pipeline products are grouped together with the raw and master calibration frames that were used to generate them, making the raw-processed data association very intuitive.

Data Classification: Gasgano overcomes the well-known difficulty of understanding what the type of a data file is (e.g. BIAS, DARK, SCIENCE, etc.). The archive filenames used for ESO data products are certainly unique, but somewhat user-unfriendly. Gasgano automatically assigns a classification tag to all FITS files (tables and images) which have been loaded by the tool, by applying a set of instrument-specific and keyword-based logical rules.

Data Browsing/Filtering: Gasgano allows the user to select a list of keywords to be displayed in the tree view for all FITS files loaded by the tool. Users might browse these directories by filtering their contents based on logical keyword-based expressions that can be specified at run-time. Filtering and searching functions permit the searching of directories for specific keyword values.

Data Viewing/Searching: Gasgano provides means for viewing and searching all the relevant information of a data file. Viewing options include displaying FITS images and header information (as a whole or based on a selected set of keywords only, e.g. by using the Filter option). The Search option allows searching keywords/keyword values within a displayed header.

Data Handling: Gasgano allows the user to make mouse selections on the files displayed on the tree. The headers of selected files are visualised in detail (or just in part, depending on the filtering option used). Additionally, selected files may be moved, copied, archived or piped into an external display tool. The Report option provides a useful tool to generate a report on the selected files. These options (Move, Copy, Archive, and Report) are Gasgano pre-defined commands. Gasgano may also be used as a front-end application to other data reduction schemes.

Recipe Execution: Gasgano will run any requested recipe on the data, using the CPL as the basis for activating the PDRMs.

A screen-shot of the Gasgano program is shown in Figure 2.

7.2. Command-line utilities

ESO also provides command-line utilities that may be used to run pipeline recipes. The aim is to provide a tool that may be embedded into data reduction scripts for the automation of processing tasks.

The basic operation of such a tool is to take various arguments specified on the command-line, and then use them to display the recipe environment, to query an individual recipe or run the recipe itself. These arguments fall into four categories:

- Command-line options for the tool itself (which control the behaviour of the utility: search paths, message level control, etc.)
- The recipe (the actual name of the recipe)
- Recipe options (if a recipe was specified)
- Set-of-Frames file (if a recipe was specified; this provides a list and designation of input data)

ESO plans to make a command-line utility available in mid-2004 as part of the next public release of its processing software.

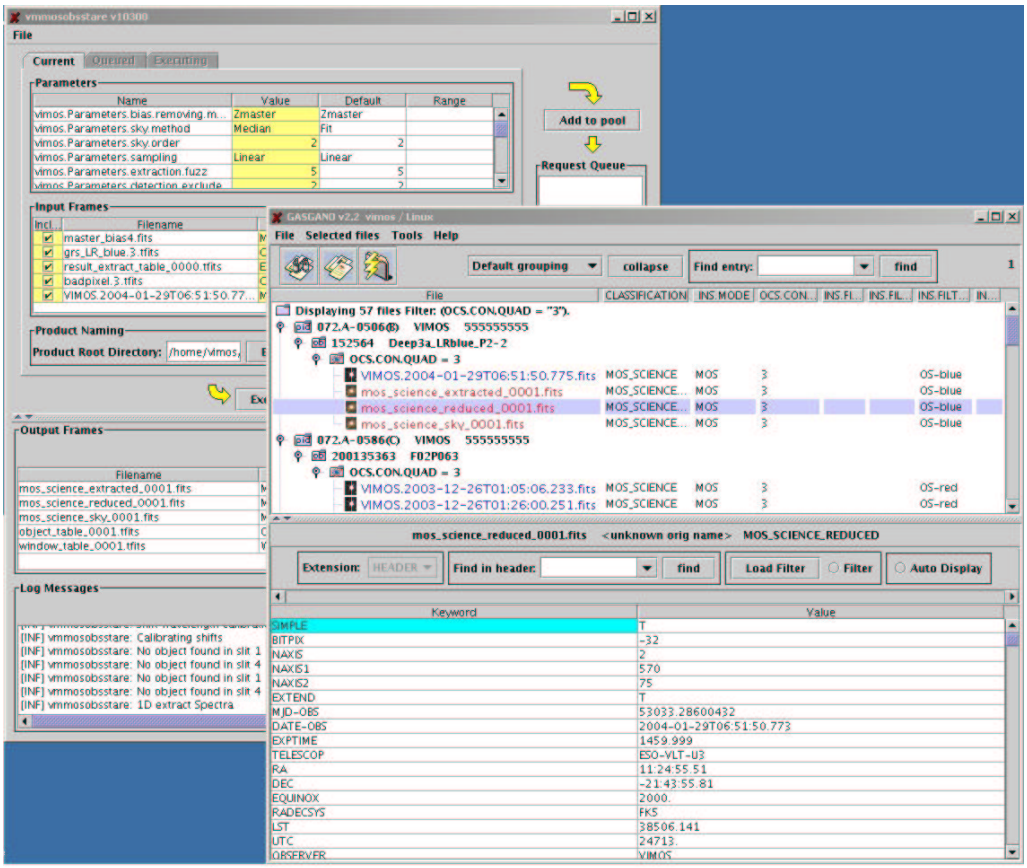

Figure 2. A screen-shot of the Gasgano program, which can activate and run CPL-based PDRMs, as well as handling and organising astronomical data.

7.3. The ESO Recipe Execution Infrastructure

The DFS infrastructure provides an automatic on-line environment for creating/executing pipeline recipes at Paranal. Frames are classified, associated to relevant calibration frames and finally reduced by applying appropriate recipes. The DFS infrastructure is being upgraded to take advantage of the CPL plugins and to enhance and simplify the operations of the ESO automatic reduction system. Of course, subsequent generations of this software will be based entirely on CPL.

8. CONCLUSION

ESO's DFS group has designed and produced the Common Pipeline Library (CPL) to meet the rapid growth in data rate and processing complexity of VLT and VLTI instruments. The CPL provides a large number of standard data processing functions and operates on FITS data using the optimised QFITS library. A pragmatic and flexible approach to code production has ensured rapid and timely completion of the project. Furthermore, ESO is committed to supporting the package, as it is incorporated into pipeline recipe software by external consortia.

The CPL has been successfully completed, and is publicly available via the website `http://www.eso.org/cpl/`. It is now being incorporated into existing instrument data-reduction pipelines as well as being used as the basis for new pipeline projects.

ABBREVIATIONS AND ACRONYMS

Table 1 lists all of the abbreviations and acronyms that are used in this paper.

Table 1. A list of abbreviations and acronyms used in this paper.

API	Application Programming Interface
CPL	Common Pipeline Library
CPLVS	CPL Validation Software
CVS	Concurrent Version System
DFS	Data Flow System
DMD	Data Management Division
DO	Data Organiser
DRS	Data Reduction System
ESO	European Southern Observatory
FITS	Flexible Image Transport System
FLAMES	Fibre Large Array Multi Element Spectrograph
FORS	FOcal Reducer/low dispersion Spectrograph
GUI	Graphical User Interface
Gasgano	A host application for running CPL-based recipes
IEC	International Electrotechnical Commission
ISAAC	Infrared Spectrometer And Array Camera
ISO	International Organisation for Standardisation
NACO	**NA**OS-**CO**NICA
PDRM	Pluggable Data Reduction Module
POSIX	Portable Operating System Interface (standard)
QFITS	(A FITS access library used by CPL)
SINFONI	SINgle Far Object Near-ir Investigation
UVES	UV-Visual Echele Spectograph
VIMOS	VIsible Multi-Object Spectrograph
VLT	Very Large Telescope
VLTI	Very Large Telescope Interferometer
X-shooter	allows simulatneous U+V+J+H band observations

REFERENCES

1. Ballester, P., *et al.*, *DFS Pipeline & Quality Control – User Manual*, ESO document VLT-MAN-ESO-19500-1619, 2002.
2. Banse, K., *et al.*, *The Common Pipeline Library — a silver bullet for standardising pipelines?*, in Astronomical Data Analysis Software and Systems XIII, 2003, in press.
3. Banse, K., *et al.*, *VLT Common Pipeline Library User Manual*, ESO document VLT-MAN-ESO-19500-272, 2002.
4. Banse, K., *et al.*, *The Common Pipeline Library Reference Manual*, ESO document VLT-MAN-ESO-19500-272, 2003.
5. Knudstrup, J., *et al.*, *Evolution and Adaptation of the VLT Data Flow System*, in Observatory Operations to Optimize Scientific Return III, Peter J, Quinn, Editor, Proceedings of SPIE **4844**, 213, 2002.
6. Kornweibel, N., *The Gasgano User's Manual*, ESO document VLT-PRO-ESO-19000-1932, 2004.
7. Zampieri, S., *et al.*, *Adapting the VLT Data Flow System for handling high data rates*, this volume.