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

The VLT Data Flow System, a flexible science support operations engine

Karim Haggouchi, Michele Peron, Pascal Ballester, Klaus Banse, Tim Canavan, Maurizio Chavan, Dario Dorigo, Carlos Guirao, Carlo Izzo, Yves Jung, Michael Kiesgen, Jens Knudstrup, Nick Kornweibel, Tom Licha, Lars Lundin, Derek J. McKay, Gerhard Mekiffer, Andrea Modigliani, Ralf Palsa, Francesco Ricciardi, Cyrus Sabet, Fabio Sogni, Arthur Stansfield, Jakob Vinther, Stefano Zampieri

European Southern Observatory

Abstract. The Data Flow System (DFS) supports the science operation model which is in place for the ESO Very Large Telescope in order to globally maximise its efficiency and ensure a high and predictable data quality level. It is a system which handles all necessary components involved in the observation life-cycle. Although after some four years of operation, the basic framework of DFS can be considered as 'complete', the implementation of new improvements and enhancements is a never-ending process due to the appearance of new observing facilities, changing hardware and software standards, and also due to feedback coming from our various users, namely the operations teams at Paranal, La Silla, at the ESO headquarters as well as external astronomers from their home institutes. This article describes how new requirements are addressed to maintain a good level of performance, together with keeping maintenance costs low. In particular, it explains how the organisation and the Software Engineering process put in place for that project makes it possible to have a robust, efficient and flexible 'science support engine', without which it would be difficult, if not impossible, to operate the ESO telescopes.

1. Introduction

The VLT Data Flow System is a distributed system which provides all necessary components to support the operations of the Very Large Telescope :

- Phase 1: assist external users in the preparation and submission of their proposals, and the Observing Programme Committee to assess and select the proposals.
- Phase 2: Observation handling to prepare and schedule the Observation Blocks, the smallest observational unit defining a sequence of high-level operations that need to be performed sequentially and without interruption in order to ensure the scientific usefulness of an observation. The latter are submitted to the VLT Control System for execution.
- An archive system to store the raw data acquired by the instruments.

- Pipelines to reduce the raw data in order to check the quality of the observations and monitor the health of instruments.

2. Why continuously enhance the DFS ?

Many new requirements imposed on DFS undoubtedly origin from its fairly long life duration — a minimum of 20 years — which implies that the system cannot be frozen for such period. There are 4 VLT telescopes at Paranal operational for 4 years, but still the overall system is not 'complete' since several new instruments are being developed and will be installed on-site in the next coming years: 7 out of the 13 VLT/VLTI instruments are currently operational, and a second generation of VLT instruments will start to be operated around 2007.

Thus, we expect on average one new instrument every year, each coming with a new set of operational concepts and requirements to be addressed by the DFS, such as higher data rates: VIMOS can produce up to 20 GB/night, VST/OmegaCam up to 75 GB/night, and VISTA up to 300-400 GB/night. For instruments producing more than 40 GB/night, the limits of capacity of the current DFS Archive System — based on DVDs — are reached. Consequently, a 'Next Generation Archive System' (NGAS) based on magnetic disks was developed, and is actually already operational.

Furthermore, the operational environment of DFS is quite complex: about 40 components (GUIs, web applications, processes, or daemons), and some 500 users located on different sites sometimes having different views, specific needs, and a wide spectrum of interests. The astronomical community submits new change requests after using the ESO facilities. Paranal, La Silla and Garching operations teams develop new schemes from their experience operating existing telescopes, together with continuously optimising existing procedures.

Finally, as for any project, we try to keep up with new technologies, new software and hardware standards in order to be able to meet new requirements, streamline the system, and on top of everything, bring down the maintenance costs as much as possible. This context is not specific to DFS, but is a real challenge to harmonise requirements and avoid introducing changes rendering existing features backwards incompatible. It requires a lot of caution, investigation, and formal software engineering processes and mechanisms to extend the system without disturbing the current operations.

3. How to smoothly handle numerous changes ?

Only significant examples of approaches initiated to adapt the DFS to new scenarios and make it flexible are described in this section.

3.1. A centralised project organisation

The DFS Group is composed by several different development teams all coordinating with a specific entity: the System Engineering Team (SEG). An important responsibility of the latter is to carry out Integration Tests, in particular making sure that all DFS components can interact together. Consequently, a significant effort (about 10 per cent of the overall DFS resources) has been invested

early in the project to thoroughly carry out automatic and manual Integration Tests. Also, experienced and professional testers act independently from the development teams in order to check the software with a particular perspective in mind (probably close to end-user's one). The idea of having independent testers is actually simple but has helped us a lot to deliver a robust software which meet users requirements.

Another basic but extremely useful concept is to always keep a test environment which - as much as possible - mirrors the main target configurations. This rule has some cost, but our experience showed it was always worth while to make such investment: many problems have been found before delivering the software, saving a lot of time for installation and avoiding interruptions in the daily operations.

Furthermore, SEG provides standard services, like configuration control management, standards for Operating Systems, preparation and configuration of delivery packages, as well as the distribution to external users.

3.2. A strict control of software changes

With a large amount of requests for new features, a well-defined scheme must be in place to handle the changes in an efficient manner. For this purpose, a Software Problem Report (SPR) system is used ESO wide for submitting change requests and bug reports. It is supported by dedicated tools, and a formal and controlled process.

3.3. An open centralised repository of the project knowledge

A major issue when having to deal with a large set of components continuously evolving is to have efficient communication means so that the project knowledge is properly saved, in other words try to catch the huge volume of 'flying or floating' pieces of information which usually circulate within the Group members over so many years.

For that aim, we have put in place a central repository which is actually a quite important internal web site (currently made of several hundreds of pages), together with some mechanisms to enable any member of the Group to add his own contribution. In that way, all usual project documents are easily accessible on-line.

3.4. A standard release cycle scheme

A release procedure of DFS has been established, and must be followed by every development team. It encompasses a number of steps and rules, for instance concerning how new DFS components should be delivered to the SEG team for Integration Testing, i.e., when they should be frozen, so that the SEG team has enough time to implement new test cases as needed before doing the testing. It also defines the content and calendar constraints of Release Notes - which should precisely describe the new features, giving direct links to all related problem reports fixed since last release. Thanks to the fact that this information is systematically required from developers, stored and easily accessible on the DFS intranet, we are then able to keep a quite precise record of the history of all DFS deliveries, and trace when and to whom they have been distributed.

3.5. A conservative approach with new technologies ?

Most software engineers desire to use new technologies. While keeping abreast of new products and trends and adopting new technologies remains critical to meeting changing requirements of projects having a long life duration, progressive modification as well as continuous improvement are more effective in an operational environment than technology revolution. For instance, we try to limit the number of programming languages used in order to bring down the maintenance costs and promote re-use of source code. Even upgrades of compilers, operating systems, and Java releases, are planned carefully in advance. It may take up to 6–8 months to test and adopt DFS software components on a given baseline (operating system + compilers). This baseline will then remain the standard one for a minimum of 1 year in the operational environment.

3.6. An iterative approach with new concepts

When significant functional or design changes are requested by our users, we usually prefer to work in an iterative manner, to avoid going too far in the implementation while the user's requirements are not clear enough. We therefore implement prototypes, like with the Next Generation Archive System, for validating new concepts, operate them for a while to gather feedback and refine the requirements. This might appear as more costly. However, in the end it saves resources as the final product often will be more robust, consistent and better fit users' requirements.

4. Conclusion

This paper presents the main Software Engineering practices applied in the context of DFS, pointing out how we gain benefit from, e.g., having a dedicated integration tests team, how effort invested on integration tests pay off in the long-term, and how important a strict control of changes or communication within the development group are.

Each of these points may not be so sophisticated individually, but the combination of all of them in an overall established and formal process had proved to be efficient in our case and a good solution to obtain a flexible and robust data flow system which is something crucial for operating a modern observatory facility.

Delivering the first version of such a system is probably not the most difficult, but defining a strategy for keeping it evolving during many years is also a real challenge. Our experience so far when it comes to developing, maintaining, and upgrading the DFS, has been good. Many problems have been faced, which have led to changes in architecture and strategies but in general none of these problems have become insurmountable obstacles leading to major delays or interruptions of operation.

References

Ballester, P., 2003, 'Data Flow for the Very Large Telescope Interferometer', in Observatory Operations to Optimize Scientific Return, SPIE Proc., 4477

Chavan, A.M., et al., 2000, 'A Front-End System for the VLT's Data Flow System', in Observatory Operations to Optimize Scientific Return, SPIE Proc., 4010

Quinn, P.J., et al., 1998, 'VLT Data Flow System: From Concepts to Operations', in Observatory Operations to Optimize Scientific Return, SPIE Proc., 3349

Quinn, P.J., et al., 2000, 'The ESO Data Flow System in Operations: Closing the Data Loop', in Observatory Operations to Optimize Scientific Return, SPIE Proc., 4010

Links

Links to the Phase II Proposal Preparation tool and VLT instruments pipelines: `http://www.eso.org/org/dmd/`
Next Generation Archive Systems Web page, : `http://archive.eso.org/NGAST`