Evolution of the Phase 2 Preparation and Observation Tools at ESO

D. Dorigo*[a], B. Amarand[b], T. Bierwirth[a], Y. Jung[a], P. Santos[a], F. Sogni[a], I. Vera[a]
[a]European Southern Observatory, Karl-Schwarzschild-Str. 2, D-85748 Garching, Germany;
[b]top itservices AG, Inselkammerstr. 1, D-82008 Unterhaching, Germany

ABSTRACT

Throughout the course of many years of observations at the VLT, the phase 2 software applications supporting the specification, execution and reporting of observations have been continuously improved and refined. Specifically the introduction of astronomical surveys propelled the creation of new tools to express more sophisticated, longer-term observing strategies often consisting of several hundreds of observations. During the execution phase, such survey programs compete with other service and visitor mode observations and a number of constraints have to be considered. In order to maximize telescope utilization and execute all programs in a fair way, new algorithms have been developed to prioritize observable OBs taking into account both current and future constraints (e.g. OB time constraints, technical telescope time) and suggest the next OB to be executed. As a side effect, a higher degree of observation automation enables operators to run telescopes mostly autonomously with little supervision by a support astronomer. We describe the new tools that have been deployed and the iterative and incremental software development process applied to develop them. We present our key software technologies used so far and discuss potential future evolution both in terms of features as well as software technologies.

Keywords: proposal submission, web, process, scheduling, observing block, Hibernate, Spring, domain modeling

1. INTRODUCTION

At ESO the submission of observing proposals is dealt with in terms of six months intervals called *periods*. These periods are further subdivided into a *phase 1* and a *phase 2*. The first phase of proposal submission starts at the beginning of the period with the call for proposals. During the proposal submission phase the principal investigators apply for observation time. The received proposals are then evaluated, ranked and accepted or rejected by a panel of external expert astronomers, the so-called observing programs committee (OPC). The accepted proposals are then scheduled for observation on the VLT. This ends phase 1 of the observation handling at ESO. Phase 2 deals with the detailed technical specification of the proposal in terms of *observing blocks*. An observing block is a single unit of observation of a specific target with a detailed description of the instrument configuration and the visibility/weather constraints. P2PP is the tool supporting investigators in the creation and editing of observing blocks. P2PP follows a client/server paradigm, allowing investigator to first work offline using a local client and to later connect and submit the created observing blocks to the ESO phase2 database. The created observing block are verified and validated by the support astronomers in Garching using the *observing tool* (OT). Finally, OT is also used by the VLT telescope operators to execute service mode observations. The raw files generated during the observation are then archived back to Garching and a night report is produced. This ends the phase 2 workflow. The phase 2 infrastructure has been thoroughly reviewed in order to support the specific requirements of survey observations at the VLT, primarily on the VISTA and VST telescopes. While P2PP and OT remain our core tools used in the data flow, they have been significantly extended with new concepts and a higher level of automation. Additionally, new tools have been added to simplify the night reporting workflow at the observatory and to minimize error-prone manual intervention throughout the night. Finally, the adoption of modern web technologies and of an iterative, incremental software development process with frequent interaction with Paranal astronomers allowed us to significantly increase our productivity and the quality of our software deliverables.

2. BRIEF HISTORY OF PHASE 1 / 2

The VLT end-to-end data flow model, of which ESO's phase 1 and phase 2 proposal submission is an integral part, was initially designed in late 1995 using Rumbaugh's Object Modeling Technique (OMT). OMT helped a lot in the definition of a clear design that models the observation data flow. A first prototype of the system was verified and validated during

*ddorigo@eso.org; phone +49 89 3200 6721; fax +49 89 320 2362; www.eso.org

the 1997 major update - known as big-bang - of the ESO New Technology Telescope (NTT). This first success was then extended to the VLT in 1999. The core concepts were

- Proposals and Observing Runs
- Observation Blocks

The design choices that have driven the evolution are:

- Thin interfaces with well-defined protocol to the control software
- Clear separation of generic versus instrument-specific concepts and implementation

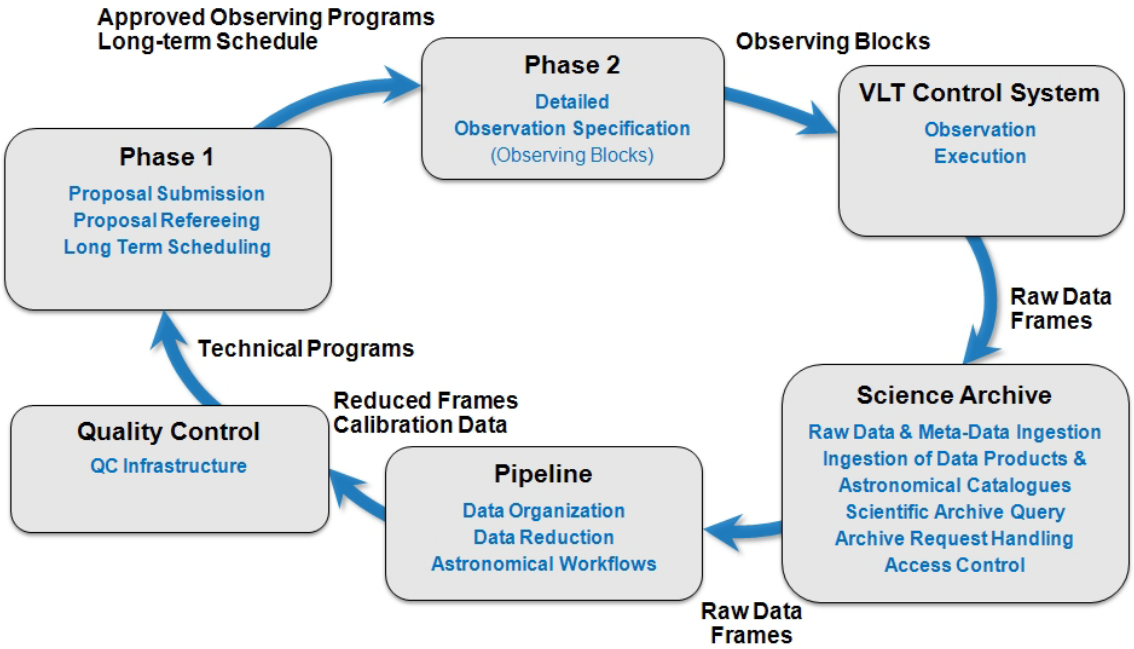

Figure 1: The VLT data flow

Phase 1, i.e. the submission, evaluation and long term scheduling of the observing programs at ESO is under the responsibility of the observing programs office (OPO), while phase 2, i.e. the verification of the submitted observing blocks and the observation execution at the telescope is managed by the user support department (USD) and Paranal Science Operations (PSO). This separation of concerns at the organizational level is reflected in many aspects of the software architecture, such as the creation of dedicated phase1 and phase2 databases and associated software applications. It is crucially important to take these organizational aspects into account when designing and evolving the overall software architecture, when devising workflows and defining interfaces. Experienced software architects often cannot deny that *organization shapes architecture*, wishing it was the other way round. All phase 1 and phase 2 applications are developed by the data flow infrastructure department (DFI).

2.1 Phase 1 Evolution

In the first implementation of phase 1 in 1998, a principal investigator wishing to submit an observing proposal had to download a LaTeX package from an ESO ftp server. A dedicated LaTeX template to be filled in provided structure and was used both for word processing as well as verification and validation of the supplied input. The completed LaTeX form was sent to ESO via email, its content was validated and if all was fine, a confirmation email was sent requesting the submission of supporting material such as pictures. If validation failed, errors were also reported via email. The proposal evaluation process and the notification of the results back to the principal investigators were entirely manual and extremely expensive and stressful.

*ddorigo@eso.org; phone +49 89 3200 6721; fax +49 89 320 2362; www.eso.org

The phase 1 process evolved with the introduction of web technologies. Given our strong Java competence we started using the first generation web framework Struts 1.x, allowing the dynamic creation of web pages by means of Java Server Pages (JSPs). We gradually adopted Struts as the basis for more and more web applications. After almost a decade using this framework, our system of web applications is mature and robust. Several web applications support investigators in their proposal submission, automatic validation and in accessing information about granted observation time. Proposal referees are supported in their review procedure and in reporting of institutional conflicts. These web applications are accessible via a common user portal, an "umbrella" application providing user account management and a single-sign on infrastructure. Finally, several internal Java desktop tools are used to set up, monitor and administrate the entire phase 1 proposal submission cycle. However, from today's point of view, our system is also fairly rigid, making future changes somewhat slow and expensive. Since the beginning in 2003, web technologies have evolved dramatically on both productivity and features and only small changes and bug fixes are still reasonable for the existing applications.

2.2 Phase 2 Evolution

The phase 2 proposal preparation software (P2PP) is used by principal investigators for the detailed specification of their observations in terms of observing blocks. The initial version of P2PP was implemented in TCL/TK. While in version 2 the fundamental concepts remained the same, it was coded in Java to provide significant usability improvements and to benefit from a truly object-oriented language. The driving requirements for P2PP were

- Clear separation of *generic* observing block from *instrument-specific* specification
- The ability for the principal investigator to work offline and submit the work to ESO at a later stage
- Whatever is entered doesn't need a save button

An important, very valuable architectural decision in P2PP was the strong separation between generic observing blocks to specify common observing constraints and instrument-specific templates (acquisition, science, calibration) to specify the details of all instrument-specific parameters. This concept also cleanly separates the development of the core tool from the independent development and evolution of so-called *instrument packages*. The requirement to work offline was realized in terms of a client/server architecture. The client communicates to the server using remote method invocation, which was initially implemented with Java RMI, but later replaced by the HTTP-based HttpInvoker protocol to more easily cross network firewalls. These basic concepts remained the same also with the recent extension of P2PP to supporting scheduling containers.

Once service mode OBs have been submitted to the ESO phase 2 database, the next step in the data flow is their review and validation by the user support department using OT. The validated observations are then ready for execution at the VLT. While VLT service mode observations are carried out using OT, visitor mode observations are executed with P2PP. This is easily possible because both applications share the same domain layer implemented in a common code base and a similar development cycle. Any changes on the domain and data access layer have to be synchronized.

The original design of the phase2 tools already stipulated the necessity of a tool supporting the decision making process about what to observe next called *Short Term Scheduler* (STS). An initial ambitious project applying constraint programming techniques to address the complex scheduling challenges did not succeed. A second, simpler approach to suggest the next service mode OB to be executed was implemented and deployed on the VLT in 2007. While the algorithm provides useful predictions on some instruments, it did not gain full acceptance on Paranal. Important criteria such as instrument-specific constraints or empirical probability distributions of weather parameters such as seeing and sky transparency were not yet taken into account, such that expert night astronomers could still significantly – although manually – optimize the scheduling. The public survey project for VISTA and VST came at the right time for us to start over on OB ranking/scheduling from a broader perspective.

*ddorigo@eso.org; phone +49 89 3200 6721; fax +49 89 320 2362; www.eso.org

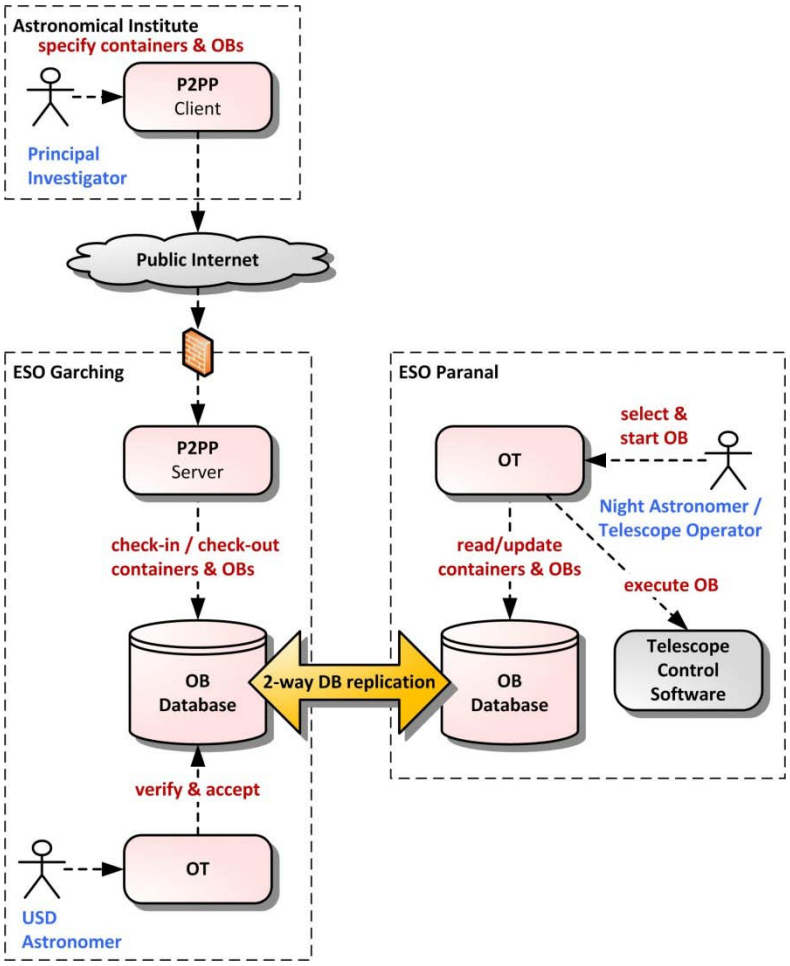

Figure 2: OB specification, review and execution workflow

3. THE SURVEY PROJECT

The execution time of most observation blocks is at most one hour. In a typical VLT night there will be around 10-20 observation blocks observed per UT. This number is still tolerable for manual scheduling by the night astronomer or telescope operator. On the other hand, surveys observations typically consist of thousands of OBs while their execution time is often much shorter. In a good night on VISTA or VST, there can be more than 50 OBs, each with a duration down to a few minutes only. OBs may be logically dependent and there is competition between different scientific programs. The most important requirements were:

- Enable principal investigators to express higher-level, long term observing strategies consisting of several OBs
- Allow the night astronomer to plan the overall observing strategy, but enable the telescope operator to execute observations independently for most of the night
- Maximize telescope usage while guaranteeing fair competition between programs on the same telescope
- Minimize operational overhead and provide comprehensive, highly automated online reports and statistics.

At the VLT facility, VISTA and VST are the first telescopes dedicated to run survey programs strictly in service mode only. It was obvious that running such large survey programs while meeting the above requirements was hardly possible within the available UT concepts and tools. The number of OBs, the complex observing strategies and the short OB execution time would require another level of automation.

*ddorigo@eso.org; phone +49 89 3200 6721; fax +49 89 320 2362; www.eso.org

4. SCHEDULING CONTAINERS

Before the introduction of scheduling containers, the OB was the atomic and only unit of observation on the VLT. Longer-term observing strategies introducing dependencies between several OBs were informally described by the investigator in a text file attached to the observation block. In order to enable investigators to formally express higher-level observing strategies in terms of dependencies between the execution of observing blocks, three types of *scheduling containers* of observing blocks were introduced: Time Links, Concatenations, Groups. Retrospectively, the usage of the term *scheduling* is a bit misleading since, as we discuss later, we are actually not scheduling but only *ranking* OBs.

A *time link* defines an ordered sequence of OBs with minimum and maximum execution time delay between them;

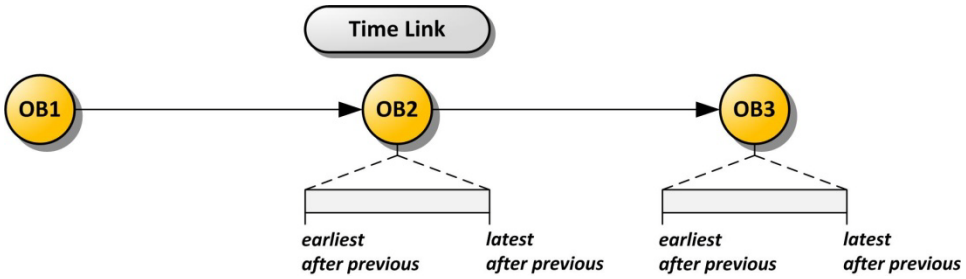

Figure 3: Time Link

A *concatenation* defines an unordered set of OBs to be executed with no break, for example a science OB immediately followed by a calibration OB.

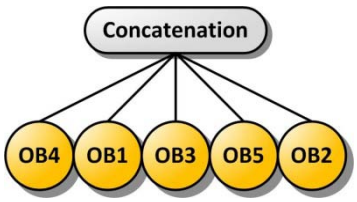

Figure 4: Concatenation

A group allows principal investigators to express their preference to execute several OBs “close to each other”. The constraint on execution preference is only desirable, not mandatory. A group has a group score which is initially 0%. Each group OB has a group contribution that is added to the group’s score after successful OB execution. Group OBs with higher group score are given higher execution priority.

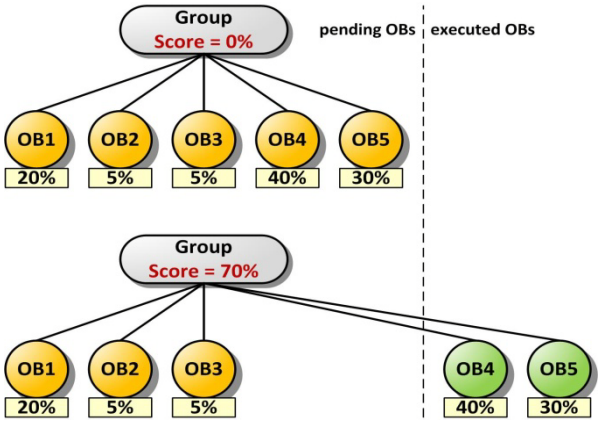

Figure 5: Group

*ddorigo@eso.org; phone +49 89 3200 6721; fax +49 89 320 2362; www.eso.org

5. CURRENT PHASE 2 INFRASTRUCTURE

At the time of this writing, VISTA, VST and UT2 have already been migrated to a completely new phase 2 tool chain supporting the definition and execution of scheduling containers. The deployment to the remaining UTs and the VLTI is to follow later this year. While some of the tools are upgrades of earlier versions, other tools have been newly introduced to optimize the observation and reporting workflow during the night and to significantly reduce manual work. The main tools are as follows

• **New Phase 2 Database & Replication –** Storage of scheduling containers and OBs, two-way replication between Garching Headquarters and Paranal Observatory.

• **P2PP 3 –** Service mode specification tool. Used by principal investigators for the definition of scheduling containers and OBs and for editing changes to ongoing surveys programs.

• **OT 3** – Service mode observing tool. Used by telescope operators and night astronomers to suggest the next service mode OB to be executed, applying a sophisticated OB ranking engine; also used by the user support department for reviewing submitted OBs.

• **vOT** – New visitor mode observing tool replacing P2PP 2 on the mountain. Used by visiting astronomers to execute visitor mode observations. Visitor mode OBs will now also be 'seen' and reported in Garching.

• **Night Log Tool** – Completely new night reporting web application used by telescope operators and night astronomers at the Paranal Observatory.

• **Garching Night Log Tool** – Completely new night reporting web application deployed in Garching to provide full online access to night reports/search/statistics to authorized users and access to observing run progress pages to principal investigators. Additionally, it allows the configuration of report distribution profiles to send specific telescope reports in PDF to a defined list of email recipients.

In the next chapters we describe the current infrastructure and the implications.

6. NEW PHASE 2 DATABASE AND REPLICATION

The new scheduling container concepts were modeled with a generic database schema. During the initial design, not all container types and their properties were known yet. Hence, we decided to design a generic tree-like model.

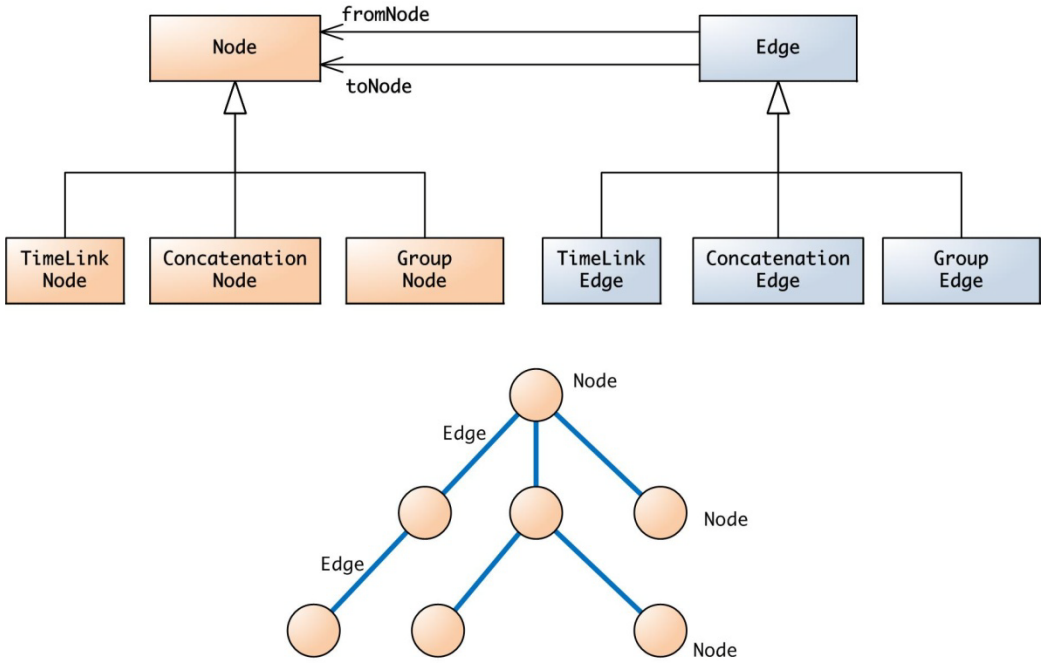

Figure 6: domain model for generic scheduling containers

*ddorigo@eso.org; phone +49 89 3200 6721; fax +49 89 320 2362; www.eso.org

The current schema uses a table for a generic node and a table for the edges connecting nodes. The additional properties of a specific container type (e.g. time link interval) are then stored in other tables in a parent-child relationship with the generic nodes and edges. This design also allows for later extension to nested containers, for instance time links of concatenations, a feature that would for instance be beneficial for the VLTI. It has to be said however that this generic design also comes with a non-negligible level of implementation and maintenance complexity.

7. P2PP, HIBERNATE AND SPRING

P2PP was the first application using the new database schema. In the previous P2PP version, the mapping between the domain model and the relational database was provided by an in-house solution called *object manager*. While P2PP and OT benefitted a lot from this solution, it was also complex to understand and limited in its capabilities. With the increasing popularity and maturity of object-relational mapping (ORM) frameworks, we decided to use Hibernate as our primary mapping layer. Given the complex nested phase 2 domain model, we wanted to benefit from powerful ORM features such as cascaded updates and deletes, eager fetching and automatic *dirty checking* and saving. In order to improve the maintainability of our code and make it more testable we introduced a comprehensive service layer on top of the domain model allowing to call services such as both locally (by the P2PP server) as well as remotely by the P2PP client. Note that for offline storage, the P2PP client uses the local file-based SQL database engine *H2* which is accessed by the same service and ORM layer. While Hibernate solved the challenge of mapping our object-oriented domain model to a relational database, we still had to address the question how to implement many standard features common to many client/server applications in the service layer. We introduced the dependency-injection framework *Spring* into P2PP, and made comprehensive usage of it, with features such as authentication, role-based authorization, remote method invocation, aspect-oriented programming for logging and database transaction management up to even applying the Spring Rich Client framework for the graphical user interface. Using this framework provided us with a major leap in productivity and maintainability, allowing us to replace a lot of infrastructure code by much simpler bean configurations.

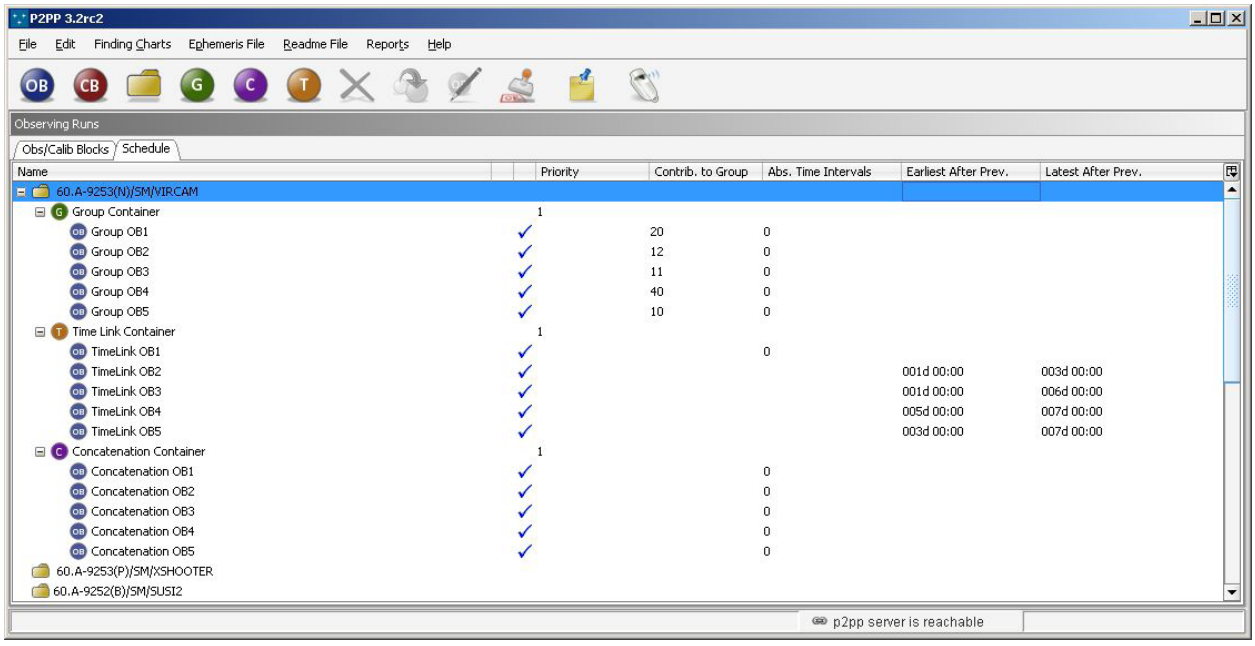

Figure 7: P2PP

As a result P2PP as well as vOT are now user friendly, highly-maintainable applications based and established standard technologies.

*ddorigo@eso.org; phone +49 89 3200 6721; fax +49 89 320 2362; www.eso.org

8. OT RANKING VERSUS SCHEDULING

OT also partially adapted the new P2PP technologies, but while P2PP focuses on user-friendly specification and representation of survey observations, OT's focus was to support the automatic ranking of service mode observations at the telescopes. With the lessons learned from the past *short term scheduler* project, we decided rather than actually *scheduling* OBs, we instead take the somewhat simpler approach to filter and prioritize (i.e. rank) OBs for a given moment in time.

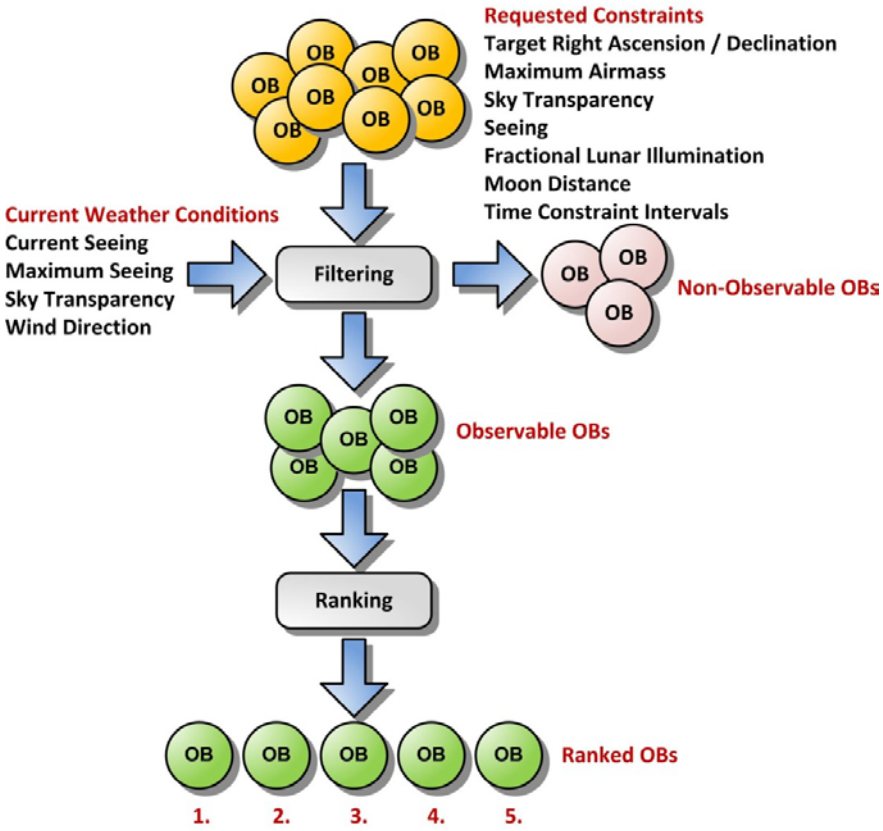

Figure 8: OB Filtering and Ranking in OT

While this approach sounds somewhat simpler, the applied algorithms are highly sophisticated and take into account the vast operational knowledge of highly experienced night astronomers, empirical probability distributions measured over several years and the outcome of dedicated simulations. For VISTA and VST we can already conclude that these are the first telescopes at the VLT observatory than can be run fairly autonomously by the telescope operators. Yet, there are still major improvements ahead of us which are already being implemented, specifically a more comprehensive consideration of instrument-specific constraints.

Since the OB filtering and ranking requirements on OT are of highly astronomical nature, it was obvious to us that it was of utmost importance to establish an intensive feedback loop to the Paranal experts for the ongoing consolidation and refinement of requirements. While detailed formal specification was written and continuously kept up-to-date by the software development team, the content was repeatedly reviewed and discussed with all stakeholders, i.e. project scientists, operators, developers, testers and served as the basis for functional testing by the software engineering department (SED). We introduced an incremental, iterative development approach, in which we incrementally implemented new features, tested them, deployed them in Paranal and gathered and incorporated feedback from operational usage. We reacted quickly to change requests and additional requirements to gain the project scientist's confidence that we are responsive to his feedback and we are actually delivering a tool that adds value. OT for VISTA had 24 beta releases and most of them were used by stakeholders either in Garching or at the VLT. The fast cycle of requirement → implementation → deployment → feedback was a major improvement in exploring and understanding what was really necessary. In order to make sure that the SW quality can still be guaranteed in such a frequent release

*ddorigo@eso.org; phone +49 89 3200 6721; fax +49 89 320 2362; www.eso.org

cycle with a high change rate, a comprehensive automated functional test suite was developed by a test engineer. The tester was able to carry out both regression testing as well as new features within two days using automatic test procedures.

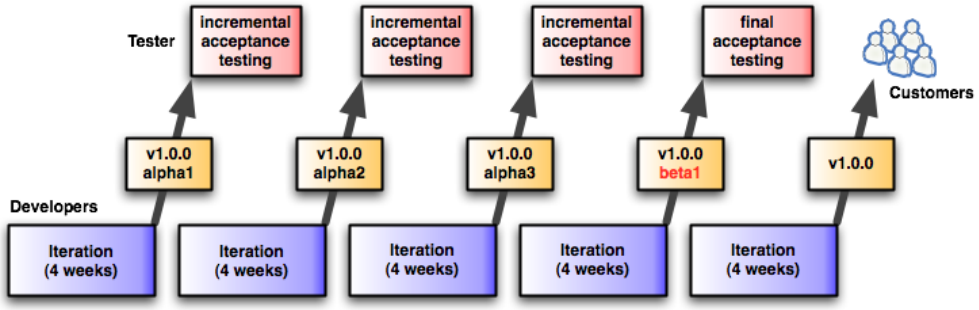

Figure 9: Iterative, incremental development and testing of OT

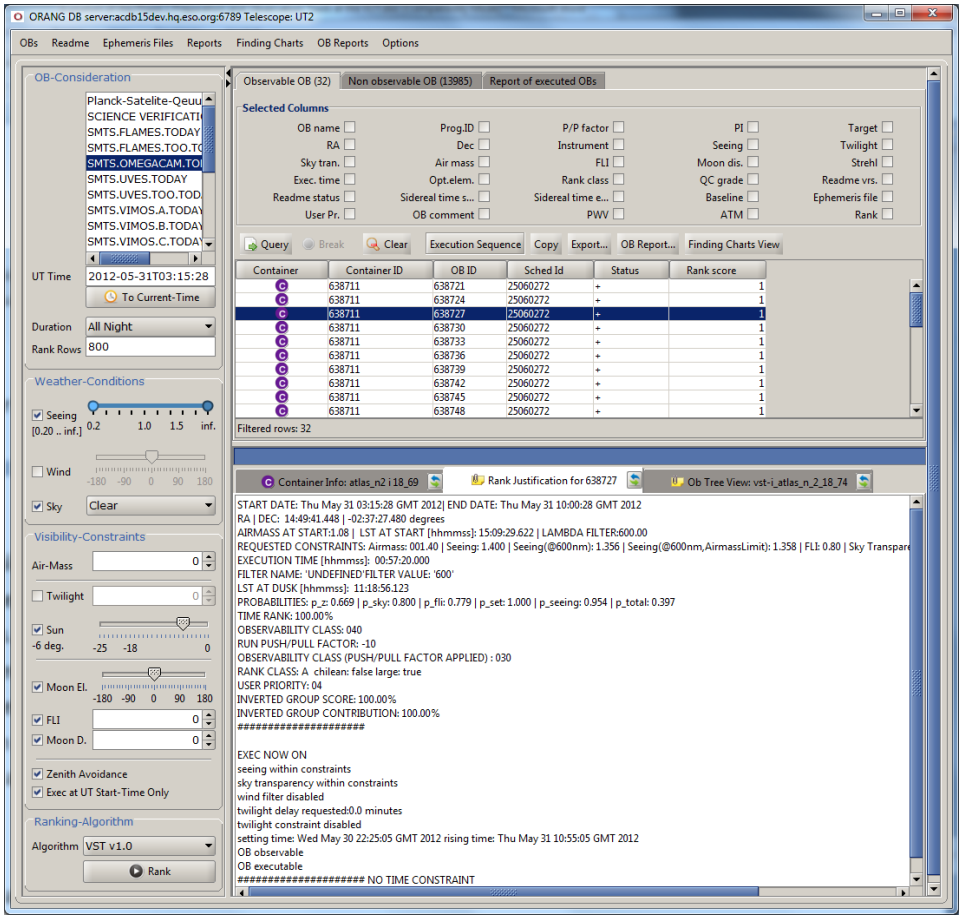

Figure 10: OT filter/ranking panel

The current filtering and ranking algorithm is already proven to be very successful on VISTA and VST. It works as expected but also sometimes produces interesting surprises that the astronomers did not expect but that are well justified. In almost 95% of the time the telescope operator relies on the OB rank as provided by OT. Only minimal additional supervision by the night astronomer is still given in the first half of the night. The scientists at the VLT can now concentrate on the science produced by the telescopes instead of the technical complications of the scheduling. The

*ddorigo@eso.org; phone +49 89 3200 6721; fax +49 89 320 2362; www.eso.org

success with VISTA/VST propelled the deployment of the new tools also to the other VLT telescopes. While UT2 has started using the new tool chain as of April 1st 2012, we are currently preparing the deployment to the remaining UT telescopes and the VLTI as of October 1st.

9. NIGHT LOG TOOL AND GRAILS

The Night Log Tool (NLT) consists of the actual NLT in Paranal for *content creation* by telescope operators and night astronomers and the Garching Night Log Tool (gNLT) for content consumption by everybody interested in night reports, i.e. visitor and service mode investigators and various ESO departments. Until recently, reporting at the VLT was somewhat simplistic and required a lot of error-prone manual labor, especially towards the end of the night. The survey project helped also in pushing for a change in the reporting area, since the previous reporting facility was not able to generate reports for VISTA and VST since it was not able to interact with the new database schema. We took the opportunity to design a new comprehensive night reporting infrastructure from scratch, with a strong focus on minimizing the information that still needs to be entered manually and with the goal to seamlessly integrate into the existing workflows. Within the data flow infrastructure department (DFI), we are continuously promoting the usage of domain modeling as an approach to structural problem analysis (as opposed to only thinking in terms of the solution space / DB schemas). As already discussed, we also had to change technology to a modern and productive web framework respecting however our acquired competencies and investment in the Java space. After careful evaluation and prototyping, we established Grails as our new core web framework for the following reasons

- **Less Code** - Grails is implemented in Groovy, an agile and dynamic scripting language with a syntax that is very intuitive for Java developers. It is significantly less verbose than Java such that we have to write, understand and maintain much less infrastructure code.
- **Shallow Learning Curve** - Groovy fully leverages the existing Java APIs. Hence, the learning curve is very shallow for a Java developer. In fact, Java code is valid Groovy code and Java classes can be called from Groovy and vice versa.
- **Focus on Domain Modeling** – Grails forces the software engineer to develop a clean domain model and to place this model in the center of problem analysis activities
- **Maintainability** – Grails establishes a very clear architectural separation of model, controller, view and services such that code is well structured, readable, and maintainable
- **Rich Ecosystem** – Grails is a mature web framework with a big and active community supplying many quality plugins that solve recurring problems in the web space, such that we do not have to re-implement commodity, i.e. non-astronomy specific problems over and over again
- **Huge Productivity Gain** – Retrospectively, we can confirm that Grails delivers on the productivity promise. We are incredibly more productive compared to our previous framework without sacrificing software quality.

With this new technology it was extremely easy to provide prototypes of the application and react to changing requirements. Based on an always running demo installation, agreed requirements were implemented and made available in the demo almost on a daily basis. Since the primary stakeholders to the NLT are located in Paranal, this was the perfect way to make sure that we are not only *building the software right*, but - much more important - that we are *building the right software*, i.e. we are truly adding value. Software development with this framework and the very intensive interaction with committed project scientists in Paranal providing frequent and proactive feedback were a truly motivating experience and the outcome is a highly usable, feature-rich tool developed with comparably little effort.

*ddorigo@eso.org; phone +49 89 3200 6721; fax +49 89 320 2362; www.eso.org

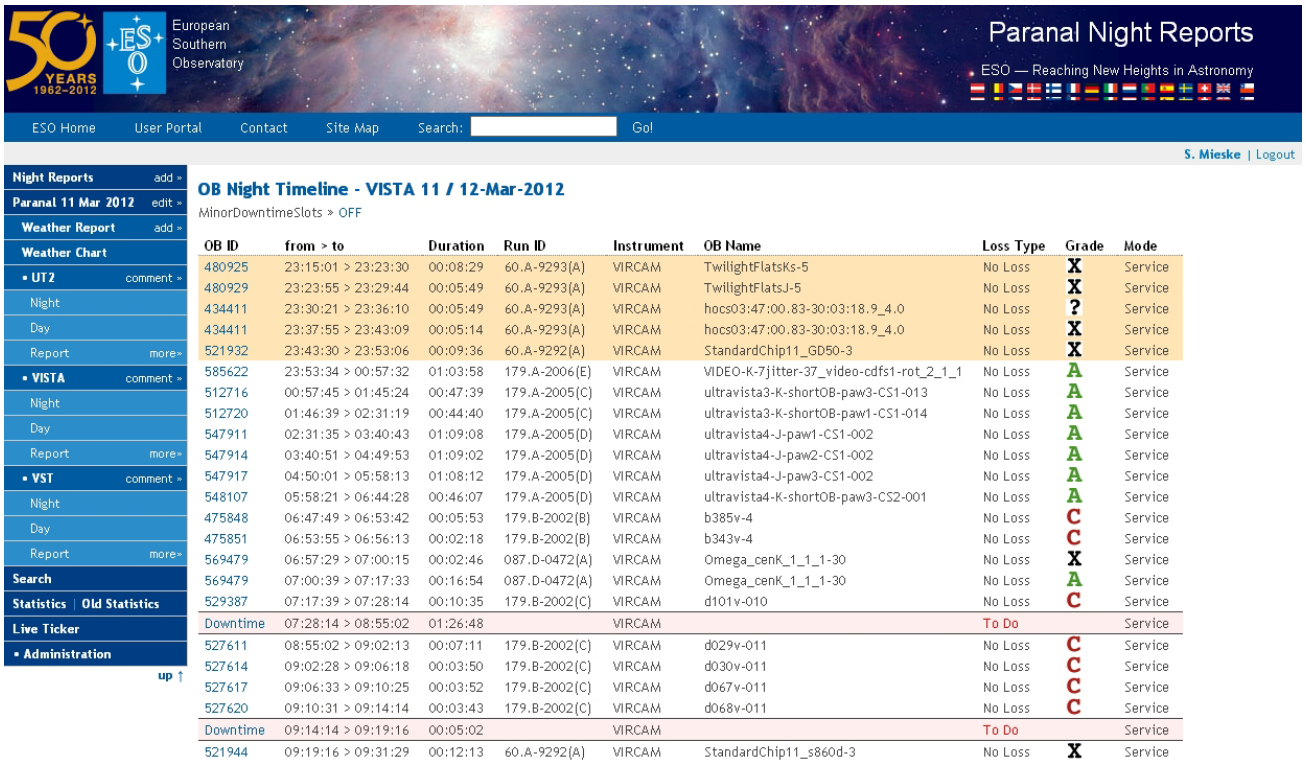

Figure 11: Night Report

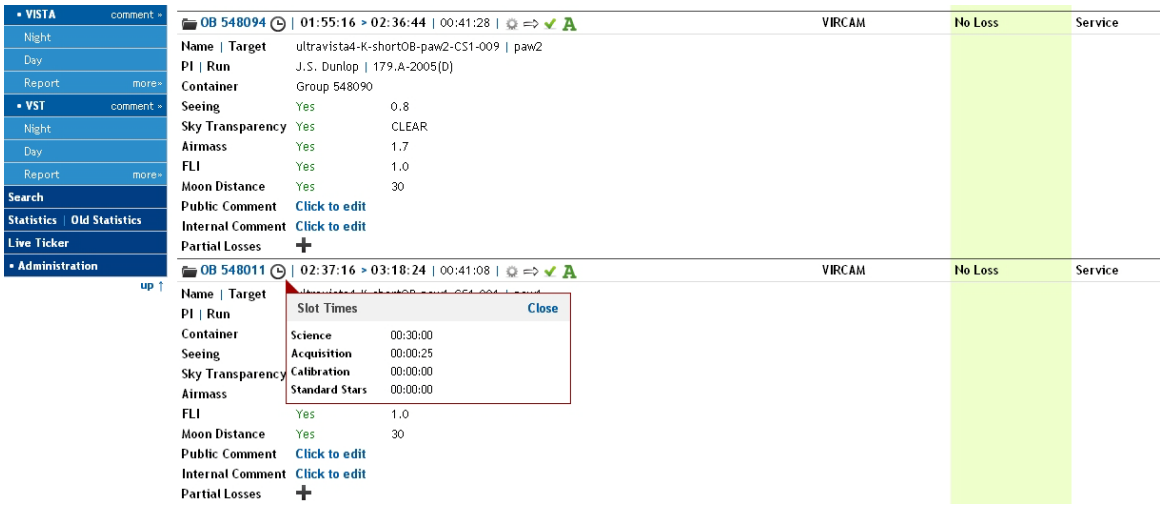

Figure 12: Detail of the report

The main features of NLT / gNLT are as follows

- Auto-generated night and day timelines of OB execution slots and downtime slots. Most of the time the telescope operator only classifies losses, add comments for 'negative classifications' and opens tickets
- Instrument-specific Time Accounting (acquisition / science / calibration / standard stars)
- Instrument/Mode-specific File Listings
- Weather Reports

*ddorigo@eso.org; phone +49 89 3200 6721; fax +49 89 320 2362; www.eso.org

- Comprehensive Search and Statistics Facilities
- Both Online and PDF Reports
- Display of Observing Run Progress Pages to principal investigators
- Configuration of report distribution profiles the send *daily* telescope reports in PDF to a defined list of email recipients
- Filtered Visitor Mode Reports sent automatically whenever a principal investigator had visitor mode observations in the previous night

10. CONCLUSIONS

Thanks to the major survey project at ESO we have acquired comprehensive knowledge of the problem domain, technologies and exploration techniques that makes us confident to also face new challenges in the coming years. We learned a lot about process, domain modeling new techniques and we built a strong and very fruitful relationship with highly committed astronomers. Applying an iterative, incremental development process with frequent feedback from testers and customers, and a high degree of test automation have proven to be highly successful. All project participants enjoy that their feedback is taken into account much quicker and we made sure that we are actually building the right software and are thus adding value to ESO and the observatory. We do not expect perfect, stable requirements and consider joint requirements analysis with customers an integral part of our job and we encourage controlled change. Looking at the past some choices would have been different. The somewhat over-generic phase 2 database design has shown some challenges. In particular the mapping of new concepts into the database is possible but very complicated to manage at the mapping level. In NLT we used domain modeling techniques that have reduced the complexity of the final design of the domain. We are now investigating if domain modeling should also be adopted more extensively for future projects. Another lesson that we learned is to avoid a big bang approach. We have reduced the risks of failures with the incremental deployment of tools starting with VISTA, then VST, then UT2 and finally all telescopes. Another area in which we are analyzing future improvements is P2PP. New requirements like nested scheduling containers - e.g. time links of contatenations - are very difficult to implement within the current client/server architecture. With the ubiquity of internet, the now obsolete P2PP requirement of having to work offline introduced enormous *accidental complexity* of custom transaction management and failure/resume handling when checking in/out OBs and containers. From today's point of view, we would build a P2PP 4 with support for nested containers as a modern web application where our framework of choice is definitely Grails.

REFERENCES

[1] Chavan A. M., "A Front-end System for the VLT's Data-Flow System" Proc. SPIE **4010**, 81 (2000)

[2] Bierwirth T., New Observing Concepts for ESO Survey Telescopes Proc. SPIE **7737**, 77370W (2010)

[3] VISTA – Visible and Infrared Survey Telescope for Astronomy (ESO Website) http://www.eso.info/sci/facilities/paranal/instruments/vista/

[4] Rejkuba, M., "Phase 2 Proposal Preparation Tool for Surveys – User Manual", ESO, http://www.eso.info/sci/observing/phase2/P2PP/P2PP3Documentation.html

*ddorigo@eso.org; phone +49 89 3200 6721; fax +49 89 320 2362; www.eso.org