ALMA RELEASE MANAGEMENT: A PRACTICAL APPROACH

R. Soto, T. Shen, N. Saez, Joint ALMA Observatory, Santiago, Chile
J. Ibsen, European Southern Observatory, Santiago, Chile

Abstract

The ALMA software is a large collection of modules, which implements the functionality needed for the observatory day-to-day operations. The main ALMA software components include: array/antenna control/correlator, submission/processing of science proposals, telescope calibration and data archiving. The implementation of new features and improvements for every software subsystem must be coordinated by considering developers schedule, observatory milestones and testing resources available to verify new software. This paper describes the software delivery process adopted by ALMA since the construction phase and its evolution until these days. It also presents the acceptance procedure implemented by the observatory for validating the software used for science operations. Main roles of the software delivery and acceptance processes are mentioned on this paper by including their responsibility at the different development and testing phases. Finally, some ideas are presented about how the model should change in the near future by considering the operational reality of ALMA Observatory.

OVERVIEW

The software delivery process of ALMA Observatory has passed for several transformations aligned to observatory's project lifecycle. Thus, it changed from a static model (with few and big releases) to a dynamic schema with emphasis at the testing phases and reducing the integration time required for a new release. This paper will present the evolution of the ALMA Software Delivery Process and Release Management by describing the advantages/disadvantages of the models adopted. Final section will give an overview how the model will change in the near future.

SOFTWARE RELEASES IN THE PAST

The ALMA Observatory started its commissioning phase at the Chilean site during the end of 2009. It considered some antennas installed at the high site (5000m) and the deployment of the first quadrant for the baseline correlator. Additionally, the assembly, integration and verification activities (AIV) related to the new array elements delivered by manufacture vendors [1], continued more intensely at the Operation Support Facilities (3000m). This period was very intensive for the computing group, since simultaneous activities had to be supported in parallel. Commissioning process by using direct observing systems (control and correlator software, front-end archive, etc.) was required at the Observatory and, at the other hand, the preparation, integration and testing of pre and post observing software (Phase 1 and Phase 2 proposal submission, time allocation committee support, data processing software, etc.) was demanded as an Early Science task, planned to be started at the end of 2010. In terms of software releases, a cycle of 6 months for the delivery of new features was established, which included capabilities for the observing systems and the proposal handling process as well. These cycles considered several phases (with a timeline predefined) before declare the software as accepted for AIV activities or Early Science observations as described at [2]: *Code Freeze Period, Initial Integrated Testing, Initial Site Testing, Computing Release, Routine Use, Acceptance Testing and Integrated Testing*. The model worked relatively well during early construction stages when the activities were concentrated at the ALMA Test Facilities [3]. Software was commissioned by using prototypes antennas and there was time available with the operational hardware for testing purposes. However, this approach was deficient when commissioning and AIV activities started at the operational site. There were less access to the hardware for testing and more pressure for having new software capabilities working in order to continue progressing into array commissioning.

Preliminary testing was initially executed in a simulated environment, which was not able to reproduce the exact behavior of the operational hardware. So, even a full battery of tests were executed, which considered new functionality and regression tests, there were not enough to deliver a mature software for the site testing. Many bugs were detected using the operational hardware, which increases the cost of correct them [4]. On the other hand, the big amount of features delivered per release also produced additional problems to distill the software. Thus, the stabilization of a new release took about two months after the integration and testing team delivered it. After that, science group should proceed with the routine use and acceptance testing, but the long integration period introduced big delays in the whole commissioning process. Given the current scenery, the integrated computing team looks for changes in the software delivery process adapting the model to the current reality.

ISBN 978-3-95450-148-9

THE CURRENT MODEL

Incremental Release Process

The model currently adopted differs from the previous one in terms of periodicity of the incremental releases delivered for science commissioning. We moved from 6-month period to bi-monthly schedule, which consider testing and integration as part of the cycle. Thus, a more frequent software delivery also implied that less features and improvements were included per release facilitating the integration, testing and debugging of the detected problems. Of course, these features and improvements are scheduled according to the observatory's milestone such as software needed for proposal submission, projects rating and starting of the observing cycle. This model also included the definition of different phases with formal handover between each one. Three phases were established:

1. **Phase A - Developer Integration & Testing**: Aimed to demonstrate that functionality is implemented as requested but based in unit tests. All tests must pass to move toward verification phase.
2. **Phase B - Verification**: Aimed to tests new functionality using both (largely) simulation and (when required) production environments. This include regression tests suite [5] for detecting bug and provide fixes to them. All tests must pass to proceed with the validation phase; otherwise features are "de-scoped".
3. **Phase C - Validation:** Aimed to validate additional capabilities and scientific data content. Performance and robustness aspects are also analyzed as part of validation tests (new releases should always behave better or at least no worse than the previous ones). Software acceptance requires both correct verification and validation test results.

The introduction of these formal phases also produced an optimization of the development and testing resources at the computing and science areas. A calendar with dates for every phase was prepared and circulated to developers, computing and science testers [6]. Also independent phases were parallelized (as showed in figure 1) which optimize the available resources. Release contents are fully tracked by using Atlassian tool called JIRA [7]. Every feature/improvement is registered in separate tickets, which also contain the testing results of every phase.

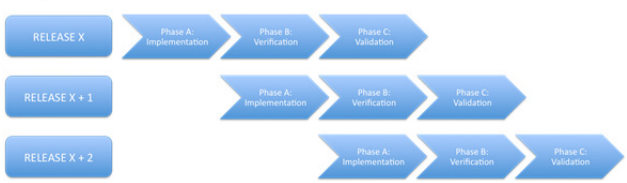

Figure 1: Incremental releases lifecycle.

Formal responsibilities were defined at the computing and science teams related to the planning and delivery of the software. Thus, the *release* and *acceptance* manager roles were introduced. The first one is responsible for the planning and delivery of the incremental releases and the completion of the verification phase. Acceptance manager represents the clients or users point of view. He/she is responsible for reviewing the planning of the releases and make sure they are according to the Observatory's milestones and science needs. Acceptance manager has also the responsibility for the execution of the validation phase and prepare the acceptance plan for the software which has successfully passed verification and validation phases.

Acceptance Process

Once several incremental releases have been successfully verified and validated, they must be accepted in order to be used for official science activities (no commissioning ones) such as: Early Science observations, hardware commissioning, etc. The acceptance manager carries out the acceptance process and it is usually held very close to an observatory milestone. It consider the following steps:

a) **Test Report Review (TRR)**: This meeting has a goal the revision of the verification/validation reports for all incremental releases included at the acceptances. Important issues are also identified and the schedule to solve them is defined. It can be more than a TRR before the acceptance if there are many issues still pending to be solved. Once everything is OK from computing and science point of view, then the candidate branch is created and we move to the acceptance testing period.

b) **Acceptance Testing Period**: During this period the final tests are performed over the candidate branch. The idea is not repeat the verification or validation phases but do a light regression tests of all applications to be deployed on production servers. These tests are performed in an isolated environment, which consider a recent image of the production DB in order to do them as much realistic as possible.

c) **Acceptance Review**: The acceptance meeting considers the revision of the acceptance tests report and also the revision of the pending items. During this meeting the final decision is taken about to deploy the new software in production servers or postpone it until all the critical items had been solved.

d) **Software Deployment in Production Environment**: This step considers the coordination with the different ALMA centers around the world for the deployment of the new software. It also included the changes into the database model needed for the new version of the applications. Usually it involves some downtime at the applications affected in order to deploy new versions. All the work is coordinated by the deployment manager who has to submit a report at the end with the status of the new software.

ISBN 978-3-95450-148-9

The Acceptance Manager is also responsible for preparing the acceptance calendar, which is aligned to the observatory's milestone and consider enough time to proceed with all the steps defined above. The acceptance calendar also contains the information about the incremental releases included into the acceptance. The calendar is distributed between scientists and developers and it represents the official information for any software planning activities.

The Software Change Control Board

Once a software release has been accepted, there is a formal process for introducing changes into the accepted branch. These requests can be associated to a software patch (related to a bug fix), to a change in the functionality requirements (called release change request), to a modification of the database schema (usually requested by software applications) or to a set of minor improvements for a specific application (called a service release). All software requests must be informed to the Release Manager by submitting a JIRA ticket [7].

A *Software Change Control Board* (SCCB) is a committee compound of various project stakeholders that typically fulfills the requirements for such a process. The group is formed for representative from computing, science and engineering sides. The SCCB (or a subset of them constituted mandatorily by the Release Manager and Acceptance Manager) will meet once a week to discuss and decide on outstanding software requests. Additional meetings may be called as required. Emergency requests need to be addressed immediately by a procedure properly established.

The decisions will normally be made by consensus. The nominated Acceptance Manager signs off all decisions. If no mutual agreement can be reached, the Release and Acceptance Managers together the head of the computing section responsible for the implementation must at least agree upon any decision. Otherwise the issue needs to be escalated to ALMA Observatory Management.

EVOLUTION TOWARD AN AGILE APPROACH

The ALMA observatory has concluded its construction phase and it is moving to full operations model. The transition implies more significance at the system robustness and stability in order to implement continuous observation model and a reduction at the time dedicated for commissioning and verification. For this reason, the software delivery process should be adjusted to the current observatory state. Thus, given the hardware restrictions, simulation capabilities will have a relevant role in the verification phase. The number of new features/improvements per release will be reduced but the emphasis at the software robustness becomes essential. System stability turns into a critical point in order to maintain observatory working most of the time, therefore the downtime due to new software releases must be strictly controlled.

The New Approach

Based in the situation mentioned above, an agile approach for the software delivery process should be adopted by the observatory. The proposal is being developed and it expected to be implemented in the coming years. Basically, this approach is based in the existence of a stable branch, which is patched for verification, and validation of new capabilities. Developers should commit functionality in separate branches and verification team should patch stable branch for verification purposes. If verification passed, Science testers should validate same functionality. After successfully validation, the patch can be integrated at the stable branch and considered ready to be used for observatory's activities. This model differs of the previous one, since integration is controlled by verification team instead of developers. Stability should be also granted since less functionality is included per iteration. Features, which do not pass verification or validation phases are rejected and scheduled for another iteration. Observatory's technical times are also optimized since only features, which have passed simulation tests, are considered to be verified with operational hardware. Despite this approach has several advantages in terms of software stability and accelerate the process for having new capabilities in production environment (by eliminating or reducing software acceptance process), it also require some fundamental changes in the paradigm currently adopted. It considers more discipline at the development, verification and validation teams in order to accomplish with the tight schedule, optimizing resources available and delivering new capabilities according to the observatory needs. The figure 2 illustrates the core of the new process where the science branch (accepted) is created after the verification and validation phases have been completed successfully.

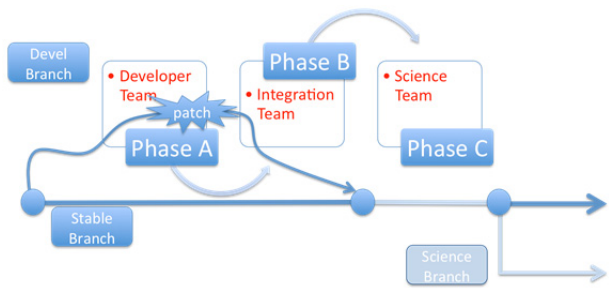

Figure 2: Proposal for agile approach.

CONCLUSIONS

This paper presented the evolution of the release management process in agreement with the life cycle of ALMA Observatory. There was a transition from a traditional and static development model, suitable for early construction phases, toward a dynamical one, which

ISBN 978-3-95450-148-9

considered commissioning restrictions. This new model takes into account the delivery of lite releases in terms of features but more stable as a whole. This also increased the frequency of the development cycles according to the observatory's milestones and decreased the integration/testing time required before the science commissioning phases. Formal phases were introduced as part of the process and responsible for every stage were properly identified and designated. This facilitated the process control, allowing a deterministic schedule for the entire cycle. There was also more emphasis for controlling changes over commissioned releases used for official science activities. The creation of a control board for approving/rejecting changes, evidenced the importance of maintain operational software stable as much as possible. The results showed at the end demonstrated this was the correct path since ALMA commissioning phase has been successfully performed from the software point of view.

However, there is still another important milestone to be completed by the Observatory in the coming years: the full operations model that will demand a new adaptation of the software delivery process in order to fulfill the operational requirements. Thus, an agile approach was proposed that considers the robustness and stability of the system as a mandatory goal over the introduction of new capabilities. It is expected that several improvements at the system simulation and continuous integration environment must be developed as part of the implantation of the model.

The experience reveals that the implementation of a new model is not a straightforward process. It will require several technical improvements but, more important and difficult, is the adaptation of human capital (developers, testers, validators) to new paradigm.

REFERENCES

[1] B. E. Glendenning, and G. Raffi., "The ALMA computing project: initial commissioning", Proc SPIE, Vol. 7019, 701902 (2008).

[2] B. E. Glendenning, J. Ibsen , G. Kosugi , G. Raffi, "ALMA software management and deployment", Proceedings of SPIE Vol. 7740, 77401L (2010).

[3] B. Lopez, R. Araya, N. Barriga, et al., "Software regression testing: practical experience at the ALMA test facility", Proceedings of SPIE Vol. 7019, 70192X (2008).

[4] V. Gonzalez, M. Mora, R. Araya, et al., "First year of ALMA site software deployment: where everything together", Proceedings of SPIE Vol. 7737, 77371Z (2010).

[5] R. Soto, T. Shen, J. Ibsen, et al., "ALMA software regression tests: the evolution under an operational environment", Proceedings of SPIE Vol. 6541, 84511R (2012).

[6] T.C. Shen, J.P.A. Ibsen, R.A. Olguin, R. Soto, "Status of ALMA Software", Proceedings of ICALEPCS 2011, Grenoble, France.

[7] JIRA Atlassian planning and tracking tool, https://jira.atlassian.com

ISBN 978-3-95450-148-9