New direction in the development of the observation software framework (BOSS)

Eszter Pozna*[a], Alain Smette [b], Ricardo Schmutzer [b],
Roberto Abuter [a], Thanh Phan Duc [a], Paolo Santin[c]
[a]ESO, European Organisation for Astronomical Research in the Southern Hemisphere,Karl-Schwarzschild-Strasse 2, D-85748 Garching bei München,Germany; [b] ESO, Alonso de Córdova 3107, Vitacura – Santiago, Chile [c] Osservatorio Astronomico di Trieste, Succursale di Basovizza, loc. Basovizza 302, 34012 Trieste, Italy;

ABSTRACT

The Observation Software (OS) of astronomical instruments, which lie directly beneath the instructions of astronomers, carrying out exposures and calibrations is the supervisor of the multi-process and multi-layer instrument software package. The main responsibility of the OS is the synchronization of the subsystems (detectors and groups of mechanical devices) and the telescope during exposures. At ESO a software framework Base Observation Software Stub (BOSS) takes care of the common functionalities of all OS of various instruments at the various sites VLT, VLTI, La Silla and Vista. This paper discusses the latest applications and how their new generic requirements contributes to the BOSS framework. The paper discusses the resolution of problems of event queues, interdependent functionalities, parallel commands and asynchronous messages in the OS using OO technologies.
Keywords: BOSS, observation software, instrument software, control software, exposure control, optimized exposures, CRIRES, PACMAN, XSHOOTER

1. INTRODUCTION TO OBSERVATION SOFTWARE

The Observation Software (OS) of an astronomical instrument is the top level control software that carries out the instructions of astronomers (given as sequential command series) in order to record astronomical images.
Receiving a command the OS distributes the necessary actions to the subsystems (detectors and groups of mechanical devices) and to the telescope. The main responsibility of OS is to take care of series of exposures and meanwhile monitor the system giving up-to-date information to the operator. These common requirements of OS are well supported by the BOSS framework[1]. BOSS has 10 years history and up to now more than 20 applications in operation at the VLT, VLTI, La Silla and VISTA sites. The 25000 lines of C++ code of BOSS gives such level of support that it fulfils all requirements of the OS of simpler instruments. BOSS supports instruments with general structure including multi level instruments (controlled by Super OS)[1] . The exposure control is done via sequential command series –SETUP, START, WAIT. The basics of the exposure control mechanism are shown in Fig. 1. For more details and functionalities of BOSS (e.g. optimized exposure control, repeated exposure functionality, error and interaction handling) please see SPIE'2008 paper [1] or the BOSS User Manual.

----
*epozna@eso.org; phone 49 89 32006441.

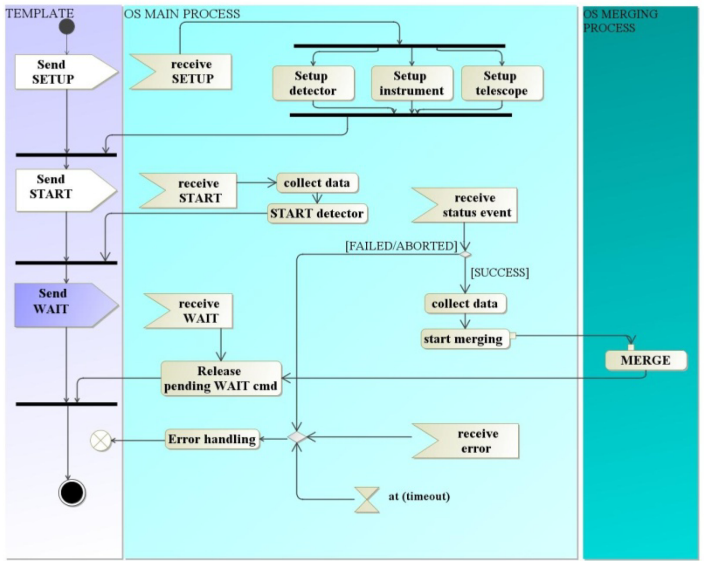
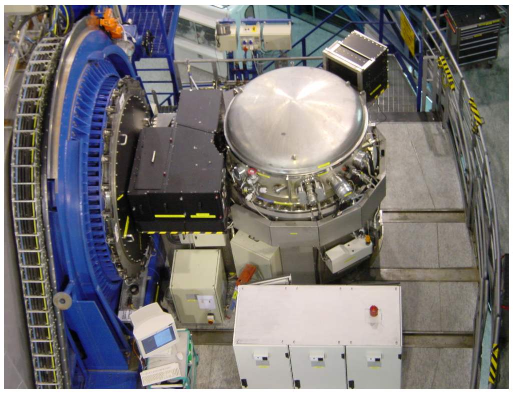


Figure 1. The basics of exposure control mechanism of the BOSS framework. The astronomical template (script containing a series of commands) appears on left side; the main process of OS is in the middle; the second process of OS responsible for fits file handling is shown on the right.

This paper introduces the last three applications of BOSS and tries to predict the update needed on BOSS for future instruments. The latest instruments delivered are: CRIRES a VLT instrument with one of the most complex OS requiring an event scheduler; PACMAN a VLTI instrument control software with a very special OS controlling only one subsystem; XSHOOTER with minimum size standard OS but with new requirements on optimized parallel exposures.

2. CRIRES OS

The VLT cryogenic high-resolution infrared echelle spectrograph CRIRES[2] is located at the Nasmyth focus A of UT1 (Antu). It provides a resolving power of up to 10^5 in the spectral range from 1 to 5µm when used with a 0."2 arcsec slit. CRIRES can boost all scientific applications aiming at fainter objects, higher spatial (extended sources), spectral and temporal resolution. Simultaneous spectral coverage is maximized through a mosaic of four Aladdin III InSb arrays providing an effective 4096 x 512 focal plane detector array in the focal plane. Adaptive Optics (MACAO - Multi-Applications Curvature Adaptive optics) is used to optimize the signal-to-noise ratio and the spatial resolution.



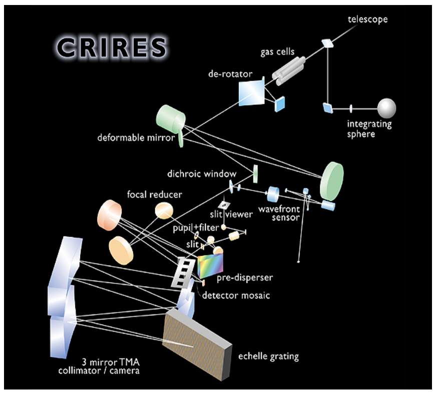


Figure 2. On the left Instrument CRIRES at the VLT platform. Light path and the mechanical components on the right.

2.1 Requirements and analysis of CRIRES OS

The ever increasing pressure for both high spectral and high angular resolution spectrograph imposes an increasing complexity on astronomical instrument control software

To achieve the accuracy required to maintain the image of the target within its 0.2 arcsec entrance slit, the Observation Control Software (OS) for the ESO-VLT CRIRES instrument must take into account a number of optical phenomena (differential atmospheric refraction, distortion, etc.), some of them time dependent. These phenomena should be accounted for even during the observation of an object moving at a rate different from the object used for auto-guiding. In order to achieve the required precision four internal software control loops needs to adjust the position of mechanical devices and/or the telescope in addition to the OS standard functionalities (e.g. monitoring, exposure handling). Besides internal activities, the OS must promptly response to sequential commands as well as simultaneous interruptions/adjustments from operator via GUI interface.

The CRIRES OS has to be ready to deal with the following problems:

- The system triggered procedures i.e. the internal loops (all with different frequencies) should not collide with each other. Theses system triggered events are:
  - internal timer events : executing adjustment for refraction and tracking
  - events from subsystems : adaptive optics system and guiding detector
- Interdependent functionalities:
  - some procedures contains others , or initiate the execution of other functionalities
  - some internal activities should be suspended while another is running

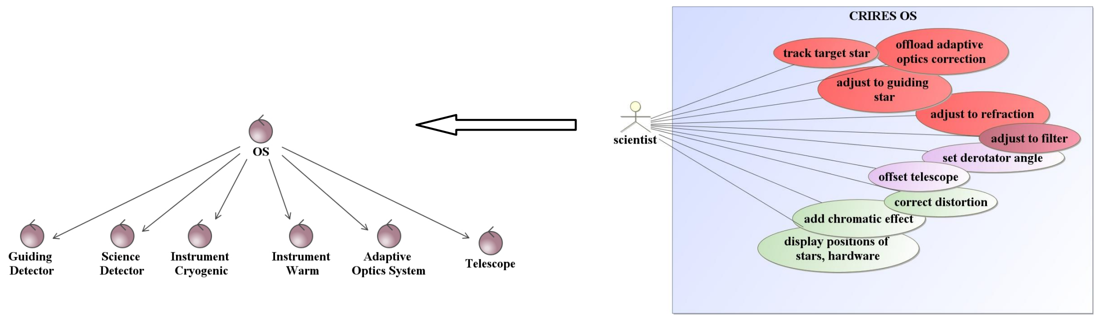


Figure 3. CRIRES software structure and the special functionalities given by the scientist

- User triggered actions should not be rejected because of system triggered actions.
- Accept simultaneous commands coming from template and GUI. I.e. user interaction from GUI should not interrupt the execution of templates. Note that BOSS primarily designed for sequential command execution and simultaneous SETUP commands by default is not permitted.
- Ensure a responsive system (using asynchronous message handling).
- Event queues raise further matters to consider in case of CRIRES:
  - The actions waiting on the queue may expire and their belated executions would result in the target jumping around its desired location, making the system unstable.
  - The event queue must be limited (i.e. stop the event queue from growing). A multiple occurrence of an event on the list should not result in an unnecessary frequency of its execution once the resources are freed.

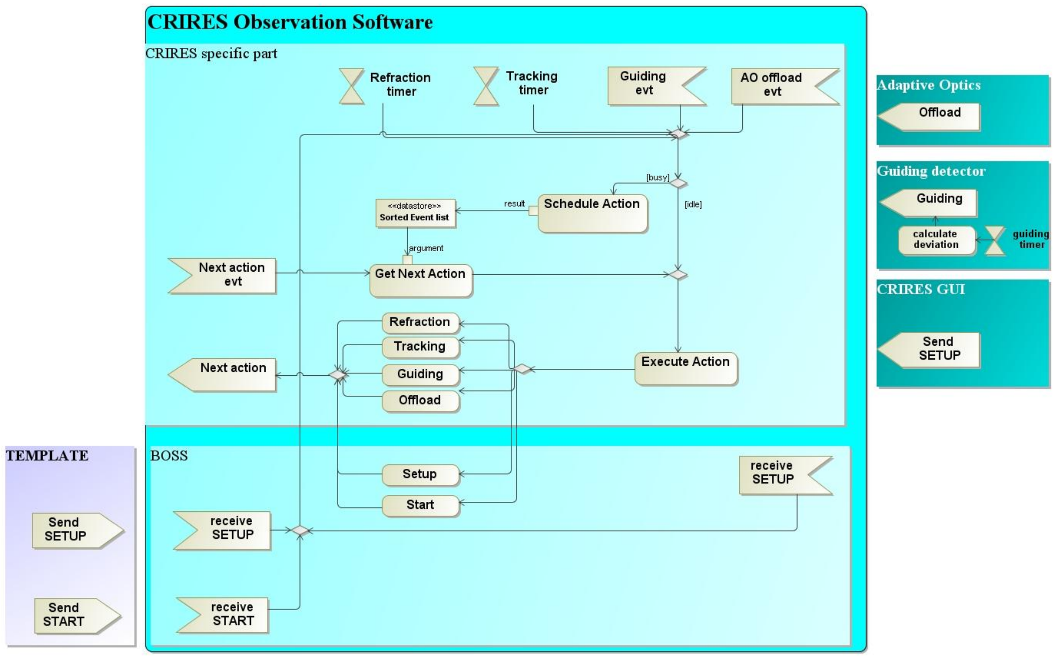


Figure 4. The scheduler synchronizes the special CRIRES actions and the basic BOSS functionalities.

- Priority tasks should be executed first.
- During the event queue handling a responsive system should be ensured as well.

Robustness, maintainability, reusability are key features of a cost-effective software. Object oriented techniques offer robust, maintainable and reusable solution that is required by both the operator and the developer. However the key to achieving this is the lack of interdependency!

2.2 Implementation

The control software of CRIRES (with its 12000 lines of code) is far from the simple cases; its many extra (and complex) functionalities rose problem with synchronisation and event queues at the top level control software. Since internal loops are not (yet) supported by BOSS, the required advanced synchronization mechanisms are implemented as an extension to the BOSS framework.

In order to resolve the seemingly contradictory problems the following was done:

- Interdependencies are removed by breaking down the individual functionalities into a list of reusable components (referred as actions).
- The handling of the events (i.e. series of actions) is then managed by the scheduler as shown in Figure 4. When an event is caught, its belonging actions are added to the ‘action-list’ while the current list is manipulated accordingly. Should there be no waiting list the procedure (or its first action) is executed immediately.
- Set a maximum occurrence for each action to protect event queue from growing indefinitely.

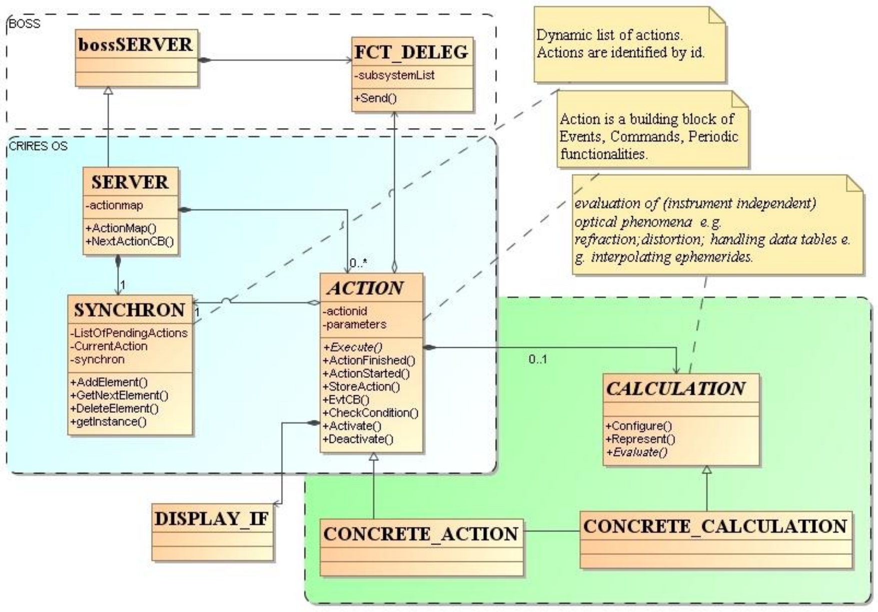


Figure 5. Class design of the scheduler mechanism and the actions in the CRIRES software

The implementation shown on Figure 5 agrees with maintainability requirements and also allows future generalization. The heart of this design is the singleton class SYNCHRON which stores the information (id and parameters) about the pending actions, yet independent from the ACTION classes. The SYNCHRON class implements an iterator on the list of pending actions. This list is allowed to be dynamically modified via its functions. The implementation of the individual actions is based on a common abstract class 'ACTION' to ensure the capability of their general handling. For more details please see ICALEPCS'2009 paper on CRIRES[3].

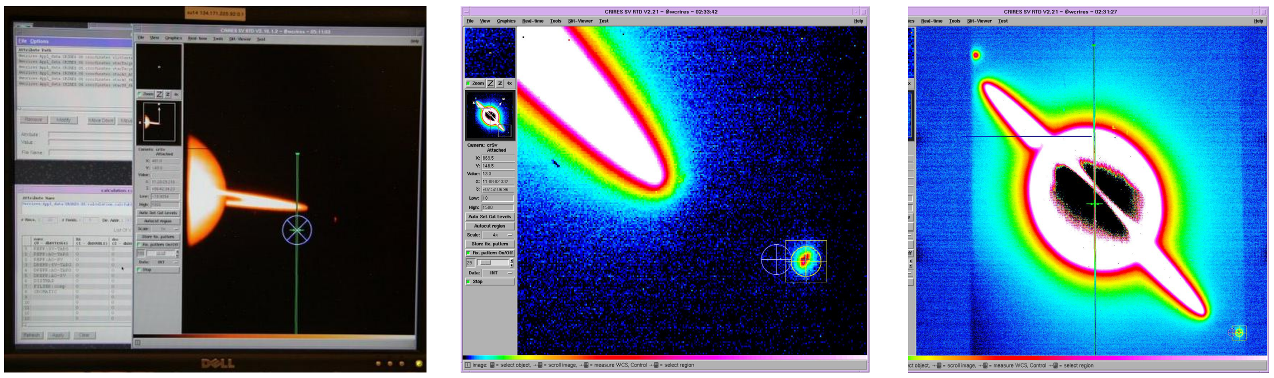


Figure 6. In order to observe Saturn, all internal functionalities must work together. Guiding takes place via a moon of Saturn, since a big blurry object is not applicable for the positioning measurement. To correct the relative motions two tables of ephemerides are used during the differential tracking. First picture shows the action list in the background during the offset of the telescope. On the second picture Saturn's moon is centered for guiding. The light that goes through the slit of the positioned object (on the third image) is captured by the Science detector and supplies the spectra.

The CRIRES system described above has been in operation (Figure 6) since approximately one year without any software break down, and offers an easy way for future updates (e.g. adding additional loops). The software design created can be also easily turned into part of the BOSS framework (although this would need some further consideration). One of the most challenging parts of the project was to identify the possible source of problems, that even if unhandled may remain hidden during tests, but can cause disturbance during operations.

3. PACMAN OS

The PRIMA facility is a system designed to enable simultaneous interferometric observations of two objects - each with size of at most 2 arcsec - that are separated by up to 1 arcmin, without requiring a large continuous field of view. The Observation software of the VLTI instrument PACMAN[4] operating the PRIMA fringe sensor units (FSU) is simple but special.

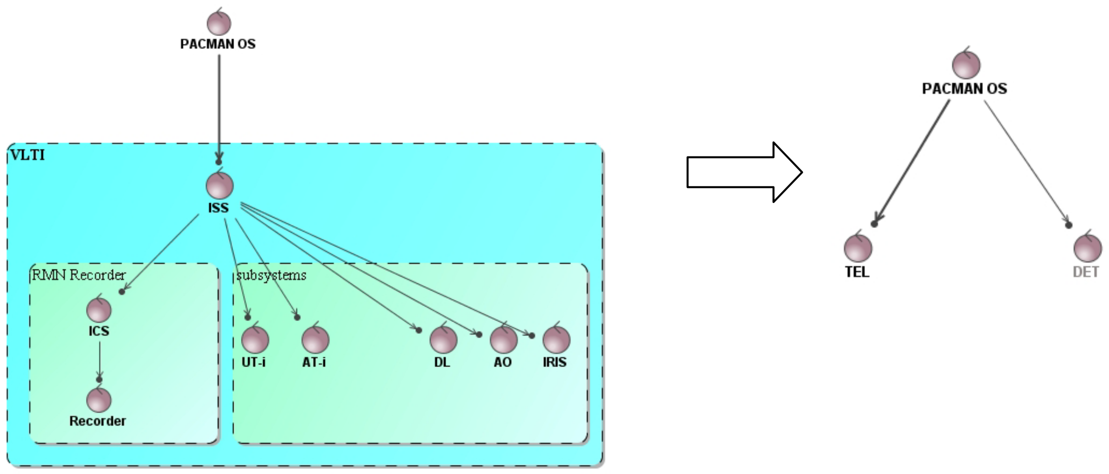


Figure 7. The real structure of PACMAN and as PACMAN OS has been configured to see it to be able to reuse the exposure handling functionalities of BOSS[1] (see also Fig 1).

The PACMAN architecture includes a special subsystem for recording RMN (Reflective Memory Network) data[5,6] that acts as the instrument detector producing binary tables. Typically for VLTI instruments the **I**nterferometer **S**upervisor **S**oftware (ISS) and its subsystems are considered as a special telescope system since the interface is similar to the telescope. (Note that BOSS supplies an interface for ISS[1] which is based on the telescope interface). However in this case the RMN 'detector' is accessible only via ISS therefore ISS itself is now also acts as a detector from the user point of view. The user starts and ends exposures (i.e. the collection of data) sending commands to ISS , however at the end of the exposure the RMN recorder data has to be merged with the data of ISS subsystems. In order to make use of the reliable exposure handling mechanism of BOSS and its standard user interface a somewhat tricky solution been applied: The observation software has been configured to have a telescope and a 'fictitious' detector subsystem ; both accessing the ISS. The only matter one had to be careful in this case is to avoid to send the commands twice to ISS. This way the costly reimplementation was avoided. For the recorder subsystem PACMAN OS now supplies an interface which converts the standard BOSS commands into ISS commands.

Most of the 800 lines of PACMAN are supporting additional fits file handling, merging the multiple binary files arriving from the recorder with extended header information and VLTI information into one FITS file.

The interface supplied in PACMAN OS serves as an example for other VLTI instruments which intends to use the RMN recorder. In the future this interface could be made available in a generic way.

4. XSHOOTER OS

XSHOOTER is the first of the second generation instruments at VLT, a wide-band (UV to near infrared) spectrometer designed to explore the properties of rare, unusual or unidentified sources. The X-shooter is a high-efficiency spectrograph with a spectral resolution in the range 4000-10000, depending on wavelength and on slit width. X-shooter consists of a central structure (the backbone) which supports three prism-cross-dispersed ´echelle spectrographs optimized for the UV-Blue, Visible and Near-IR wavelength ranges respectively. The backbone contains the calibration and acquisition units, an IFU which can be inserted in the light path and reformats a field of 1.8'' ×4'' into a long slit of 0.6'' × 12'', two dichroics to split the beam amongst the three arms and relay optics to feed the entrance slits of the spectrographs.

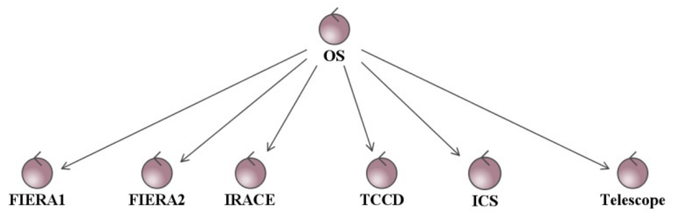


Figure 8.The subsystems controlled by the XSHOOTER OS

The OS of the XSHOOTER software has the same number of subsystems to control (Figure 8) as the CRIRES OS yet XSHOOTER OS is one of the simplest OS with its ~150 lines code; and hence it's a perfect BOSS application in terms of minimum implementation and maintenance efforts. Its only specialty regards the looping technical CCD the status of which is excluded from the global status.

XSHOOTER however raised the problem of optimizing the execution of parallel exposures, i.e. starting an exposure on a detector as soon as it finished independently from the ongoing exposures on the other detectors. Although BOSS supports the execution of simultaneous WAIT commands which makes this optimization possible for more user friendly approach (and to support the standard sequential command execution) BOSS recently been updated with a WAIT parameter option that returns the names of the idle detectors.

Should in the future exposures needed to be controlled independently a scheduler mechanism as shown in CRIRES would be necessary to avoid the collision of SETUP and START commands.

5. SUMMARY AND FUTURE CONSIDERATIONS

The curve of progress of BOSS (Figure 9) fits the expectation. Initially BOSS was based on an existing instrument software[1,7,8] , nevertheless during the generalization process and adjustment to new requirements its size has tripled.
The intensive progressing period (an average 3-5000 lines of code per year has passed in the first 4-5 years,however many instruments still contribute with a smaller bigger requirement to the progress of BOSS. E.g. for CRIRES a delegation class (Figure 5) was supplied to give access to some of its useful functionalities (and hence simplify the application). XSHOOTER contributed with the optimized parallel exposure functionality.

Figure 10 shows the applications in chronological order indicating that even up to date there are applications for which BOSS is perfectly suitable and ensures their quick development. The currently ongoing project (SPHERE, MUSE, ESPRESSO) seem to fall into the simpler category of Observation Software too.

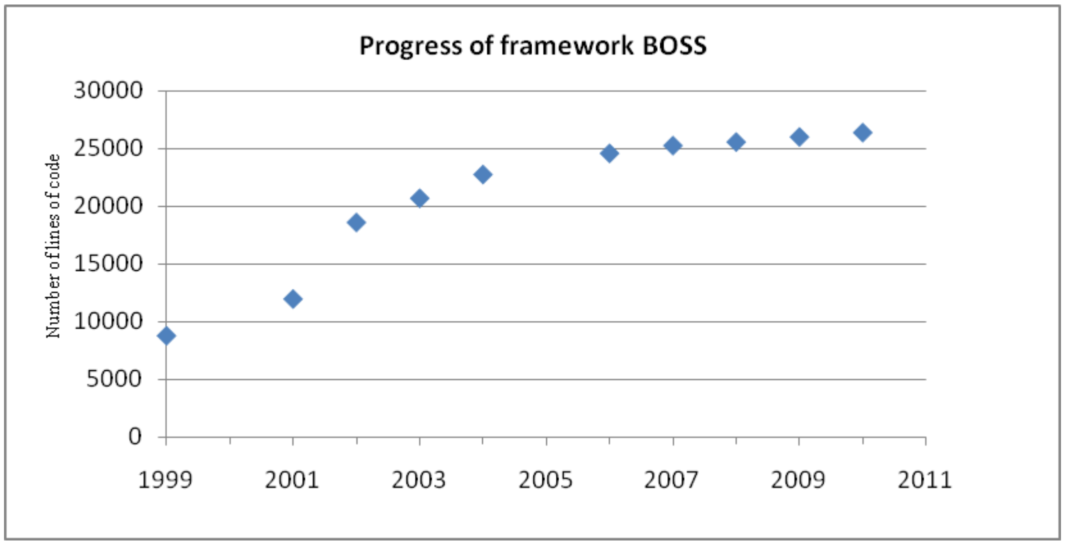


Figure 9. Development of BOSS framework over the years in terms of lines of code.

Looking at the size of applications we see a tendency that from time to time a bigger application stands out, the last one CRIRES is approximately half a size as BOSS. The experience with CRIRES pointed out a missing support for internal loops. Also several control procedures for adjusting to optical phenomena have been developed within CRIRES OS which could be needed more widely. CRIRES OS software might be just the first of its kind at ESO. The increasing resolution of the detectors imposes higher demand on the control aiming to achieve (and/or not to lose) the level of precision that the new detectors are now allowing. As time goes the once specific features become generic (such as interacting with adaptive optics facilities) and ask for generic software support.

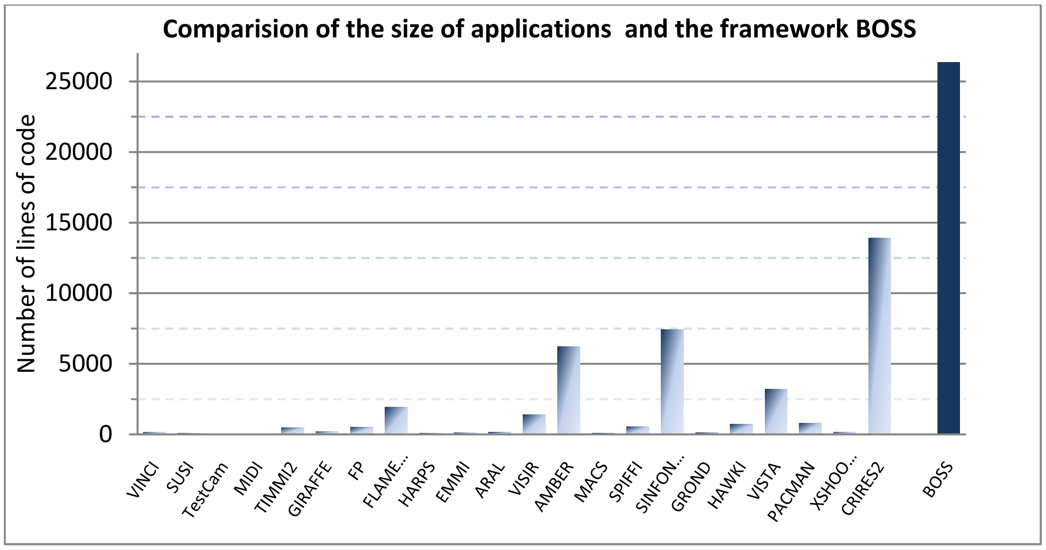


Figure 10. Comparison of the size of the observation software application of various atronomical instruments to the size of BOSS.

Extrapolating the image on Fig 10 predicts that there are instruments coming which will have more complex requirements then CRIRES while simple applications will still appear (perhaps) more frequently. Therefore the general features of complex instrument could be supplied either as extension to BOSS, or as plug-in classes in separate modules or library. It is important that the generic behaviors of complex instruments are identified in advance so that the functionalities can be supported in a generic way saving time and cost of reimplementation. Whether or not this will be done as part of BOSS or perhaps a second generation of BOSS is yet to be decided.

REFERENCES

[1] E. Pozna, G.Zins, P.Santin, S.Beard, "A common framework for the observation software of astronomical instruments at ESO", Proc SPIE Vol. 7019, 70190Q (2008)

[2] Käufl, H. U., Amico, P., Ballester, P., Bendek Selman, E. A., Bristow, P., Casali, M., Delabre, B., Dobrzycka, D., Dorn, R. J., Esteves, R., Finger, G., Gillet, G., Gojak, D., Hilker, M., Jolley, P., Jung, Y., Kerber, F., Klein, B., Lizon, J., Paufique, J., Pirard, J., Pozna, E., Sana, H., Sanzana, L., Schmutzer, R., Seifahrt, A., Siebenmorgen, R., Smette, A., Stegmeier, J., Tacconi-Garman, L. E., Uttenthaler, S., Valenti, E., Weilenmann, U., Wolff, B., "CRIRES: commissioning and first science results" in Ground-based and Airborne Instrumentation for Astronomy II, edited by Ian S. McLean, Mark M. Casali, Proceedings of SPIE Vol. 7014 (SPIE, Bellingham, WA 2008) 70140W

[3] E.Pozna, A.Smette, R. Schmutzer: Task shyncronisation in the Observation Control Software for the ESO- VLT CRIRES instrument. ICALEPCS2009 conference (Kobe/Japan) http://icalepcs2009.spring8.or.jp/abstract/pdf/THP087_POSTER.PDF

[4] Abuter R., Sahlmann J., Pozna E., PACMAN: PRIMA Astrometric Instrument Software (SPIE, San Diego, WA 2010)

[5] Abuter, R., Popovic, D., Pozna, E., Sahlmann, J., Eisenhauer, F., "The VLTI real-time reflective memory data streaming and recording system" in Optical and Infrared Interferometry, edited by Markus Schöller, William C. Danchi, Françoise Delplancke, Proceedings of SPIE Vol. 7013 (SPIE, Bellingham, WA 2008) 70134A.

[6] Le Bouquin, J.-B.; Abuter, R.; Haguenauer, P.; Bauvir, B.; Popovic, D.; Pozna, E. : Post-processing the VLTI fringe-tracking data: first measurements of stars Astronomy and Astrophysics, Volume 493, Issue 2, 2009, pp.747-752 http://adsabs.harvard.edu/abs/2009A&A...493..747L

[7] Zins, G., Lacombe, F., Knudstrup, J., Mouillet, D., Rabaud, D., Charton, J., Marteau, S., Rondeaux, O., Lefort, B., "NAOS Computer Aided Control: an Optimized and Astronomer-Oriented Way of Controlling Large Adaptive Optics Systems", ADASS, Vol. 216, (2000)

[8] Zins, G.; Lacombe, F.; Knudstrup, J.; Mouillet, D.; Rabaud, D.; Marteau, S.; Rousset-Riviere, L.; Rondeaux, O.; Charton, J.; Lefort, B.; Rousset, G.; Hubin, N. N, "NAOS computer-aided control: an optimized and astronomer-oriented way of controlling large adaptive optics systems" Proc. SPIE Vol. 4009, p. 395-401 (2000)