# The ALMA Common Software (ACS) as a basis for a distributed software development

Gianni Raffi, Gianluca Chiozzi

*European Southern Observatory (ESO), 85748 Garching, Germany*

Brian Glendenning

*National Radio Astronomy Observatory (NRAO), Socorro, NM 87801*

**Abstract.**

The Atacama Large Millimeter Array (ALMA) is a joint project involving astronomical organisations in Europe, North America and Japan. ALMA will consist of 64 12-meter antennas operating in the millimetre and sub-millimetre wavelength range, with baselines of more than 10 km. It will be located at an altitude above 5000m in the Chilean Atacama desert. The ALMA Computing group is a joint group with staff scattered on 3 continents and is responsible for all the control and data flow software related to ALMA, including tools ranging from support of proposal preparation to archive access of automatically created images.

Early in the project it was decided that an ALMA Common Software (ACS) would be developed as a way to provide to all partners involved in the development a common software platform. The original assumption was that some key middleware like communication via CORBA and the use of XML and JAVA would be part of the project. It was intended from the beginning to develop this software in an incremental way based on releases, so that it would then evolve into an essential embedded part of all ALMA software applications. In this way we would build a basic unity and coherence into a system that will have been developed in a distributed fashion.

This paper evaluates our progress after 1.5 year of work, following a few tests and preliminary releases. It analyses the advantages and difficulties of such an ambitious approach, which creates an interface across all the various control and data flow applications.

## 1. ALMA Project in Summary

The ALMA project will consist of 64 antennas. Each of them will have a 12 m diameter, with a total surface accuracy better than 25 microns and point within 0.6 arcsec. It will be possible to define ALMA array configurations ranging from having all antennas within 150m to 10Km.

The selected site is over 5000m altitude in the Chilean Andes at the border with Argentina and Bolivia near the village of S.Pedro de Atacama.

## 2. ALMA Computing work

During the ALMA design phase (Phase 1) the ALMA computing work has been tackled in different ways. First a **Top-down** approach was used to define the high level requirements and elaborate on them with Use Cases. Next came the work of the High-level Analysis and Design team aiming to a first global architecture with analysis classes being merged into Packages. This will form the general frame for the development of subsystems:

- Control and Correlator software (input: 96 Gb/s per antenna)
- On-line Pipeline, Off-line Data Reduction, Telescope Calibration
- Archiving (Data rate: 10MB/s - 300 TB/year)
- Observing Preparation, Scheduling (automatic operation support)

At the same time a **Bottom-up** approach was also needed to create the control software for the prototype antennas, test correlator and the resulting test interferometer. It was decided that this would use from the beginning the ALMA Common Software in order to validate it.

The third component of the Phase 1 work is the **Common infrastructure**. While any computing project has to define early its standards, it is equally important that procedures are defined ahead of development. These include configuration control and change request procedures, standard environments for development and common makefile scripts.

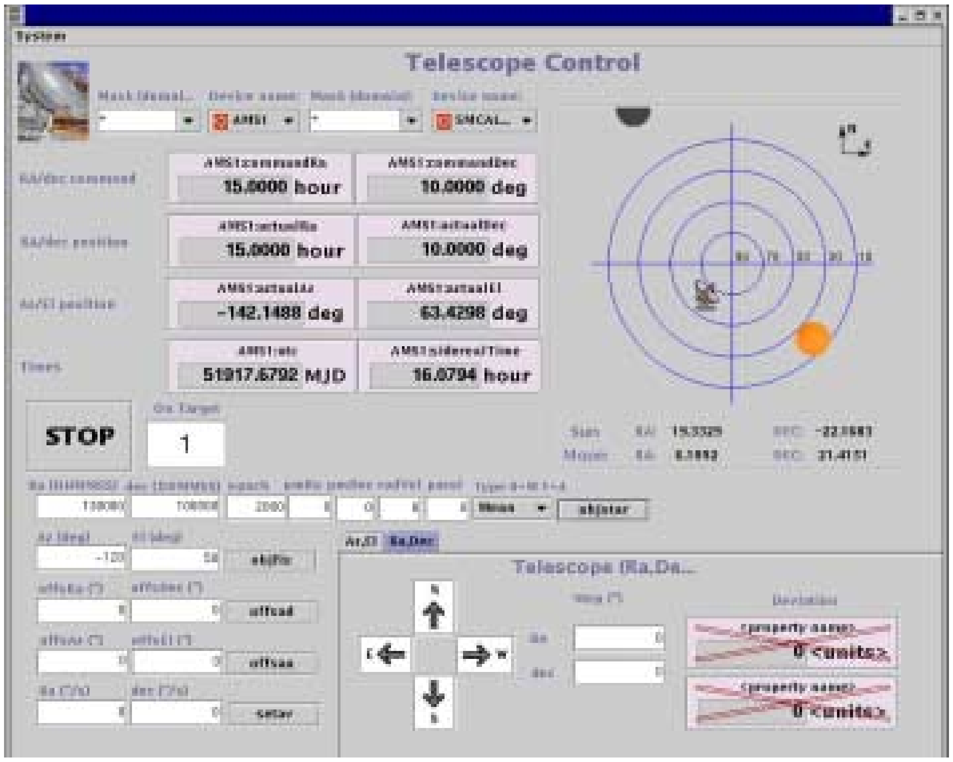
Another important element was the establishment of a common software layer between the operating systems and the specific ALMA software, as explained below.

## 3. ALMA Common Software (ACS)

The ALMA Common Software (ACS) is a comprehensive framework on top of the operating system, offering a complete environment and structures at the base of application software developments. ACS is meant to be a general system, based on available middleware (CORBA) and the easy embedding of languages like JAVA and Python.

We believe that the use of a common software layer in a very geographically distributed development situation will be the best way to enforce the use of common constructs. ACS shall provide a well tested platform that embeds standard design patterns (much better than a set of written rules) and avoids duplication of effort. At the same time this will provide a platform where upgrades can be incorporated and brought to all developers. It will also standardize the underlying architecture of software modules, making maintenance affordable.

ACS will be released every 6 months. Being distributed at fixed dates it will provide the pace for software development iterations in the whole ALMA software effort. The ESO team has developed and released for 8 years now a similar system, which is in successful use at the ESO Very Large Telescope (VLT) Observatory (Raffi, Filippi 2000).



AMS1:commandRa 15.0000 hour
AMS1:commandDec 10.0000 deg
AMS1:actualRa 15.0000 hour
AMS1:actualDec 10.0000 deg
AMS1:actualAz -142.1488 deg
AMS1:actualEl 63.4298 deg
AMS1:utc 51917.6792 MJD
AMS1:siderealTime 16.0794 hour
On Target 1
Sun RA 15.3329 DEC -22.1581
Moon RA 8.1852 DEC 21.4151

Figure 1. User Interface based on JavaBeans

## 4. ACS software status

In order to start effectively with limited resources a project like ACS, it was decided to look for a suitable existing software that could give already a good basis for ACS and bring in from the beginning all the relevant CORBA know-how. A system developed by Jozef Stefan Institute (JSI) in Slovenia was found to fulfill these pre-requisites (Plesko 1996) and therefore a collaborative effort was started by the ESO team contracting further developments to JSI.

This lead to a prototype release of ACS (ACS 0.0) tested at the Kitt-Peak 12m radio antenna in Dec.2000. The first production release of ACS (1.0) was distributed in Sept.2001. ACS is already used with the test interferometer software and is going to be integrated shortly within the test correlator control software.

## 5. ACS Architecture

The ACS architecture is based on the concept of Distributed Object. This is a base class that can be mapped into physical devices in the case of control software or into more abstract kinds of objects for other kinds of software. The ACS architecture uses a 3-tier model: Distributed Object (DO), Property, Characteristics. Examples of DO implementations are physical devices like temperature sensors and motors. Each DO has read-only or Read/Write Properties, for example status values, position values, control and monitor points. DOs and Properties have Characteristics, which are static data like units, ranges, default values. Characteristics are stored in a Configuration Database and so they can also be changed with tools and GIUIs of the database. DOs can be browsed with the ACS Object Explorer.

ACS comes also with integrated JavaBeans to construct user interfaces. These can be created on the client side automatically from the DO interface in IDL (Interface Definition Language). The user interface based on JavaBeans that was used in the Kitt-Peak 12m antenna test can be seen in Figure 1.

The different components of ACS, most of which are at least partly implemented in Release 1.0, consist of layers of increasing complexity: Base tools (CORBA, ACE, drivers, tools), Core components (Distributed Object, Data Channel, Error system, Logging, Time, Astro libraries), Services (Archiving, Commands, Alarms, Sampling, Management and access control), High-level APIs and Tools (GUI libraries, Scripting, Application framework, FITS libraries).

A detailed description of the ACS architecture is given in (Chiozzi, Gustafsson, Jeram 2001), while its requirements can be found in (Raffi, Glendenning 2000).

## 6. ACS underlying technology

ACS runs on Linux and VxWorks and is tested for interoperability with the CORBA ORBs TAO and Orbacus, while for Python work is on-going to integrate Omniorb. Programming in C++, JAVA and Python is supported. DB2 is being evaluated as the final configuration database.

Although ACS is driven by ALMA requirements and based on its standards, it is quite general and can be used by any project, which would decide to go for an object oriented approach in a distributed project based on CORBA and Linux. It is intended to make ACS freely available under the usual GNU free license scheme.

**Acknowledgments.** The ACS software is the result of the work of the ACS team lead by G.Chiozzi, mainly based at ESO but with contributions also from other ALMA centres (IRAM, Bochum, NRAO) in collaboration with JSI. Information on Computing work and Computing Documents can be found at: `www.alma.nrao.edu/development/computing/docs`

## References

Raffi, G. Filippi, G. 2000, SPIE, 4009, 197

Plesko, M. 1996, PPAC Workshop, DESY, Hamburg

Chiozzi, G. Gustafsson, B. Jeram B. 2001 ALMA Computing Document N.16

Raffi, G., Glendenning, B. 2000 ALMA Computing Document N.5