11 Computing

B.E. Glendenning, G. Raffi

2000-Feb-15

11.1 Introduction

The role of software for the test interferometer is two-fold. First, it is required for testing the prototype antennas. Second, it is a prototype of some aspects of the final array software and use on the test interferometer can be used to prove some aspects of its design. This second use must not jeopardize the primary objective: testing the antennas.

The software group leaders have decided that prudence dictates that there should be a backup strategy to be employed if it appears that the new joint developments will not be ready in time to test the antennas. In this eventuality ESO/VLT common and control software will be modified to handle these tests. This backup approach was successfully tested in November 1999 by modifying ESO/VLT software to perform optical pointing measurements on the NRAO 12m telescope on Kitt Peak. As this plan would entail considerable original work to integrate radio-astronomy specific devices, it will be employed only if the new software development becomes delayed.

There is no intention to test advanced observing systems for the test interferometer. That is, sophisticated archiving, dynamic scheduling, image pipelines, and advanced observer preparation tools are not considered to be necessary for the antenna tests. The elements of the final software that will be prototyped on the test interferometer are those that are related to common and control software.

11.2 Software Engineering Practices

It is important that a distributed software development effort that will last for a decade agree upon software engineering practices. An important product of Phase 1 is a list of agreed upon software engineering practices in the areas of:

- Design Process (almost certain to follow the Rational Unified Process, using UML as a design language).
- Release Strategy (fixed date *vs*. fixed content).
- Documentation standards and formats (likely MS Word).
- Review procedures (based on ESO practice, emphasizing written comments in advance of review meeting).
- Code acceptance and documentation standards.
- Testing (unit and integration).
- Maintain a list of supported platforms.
  - Language: C/C++, Java, and TBD scripting.
  - OS (likely a mixture of VxWorks, Linux, and Windows).
- Configuration management.

While there are no jointly agreed upon standards yet, there are some proposals outlined by Harris *et. al.* (1999).

11.3 Common Software

ALMA will require a considerable software effort of many different types. However, a large part of the software can be shared, with the same API used by many software systems both real-time and non-real-time.

This common software is useful for several reasons:

- It provides common services required by more than one software subsystem. This avoids duplication of effort, and ensures that those common services behave the same in different software subsystems.
- It provides communication and interaction mechanisms between software systems, and provides for a common style of programming (*e.g.,* event driven).

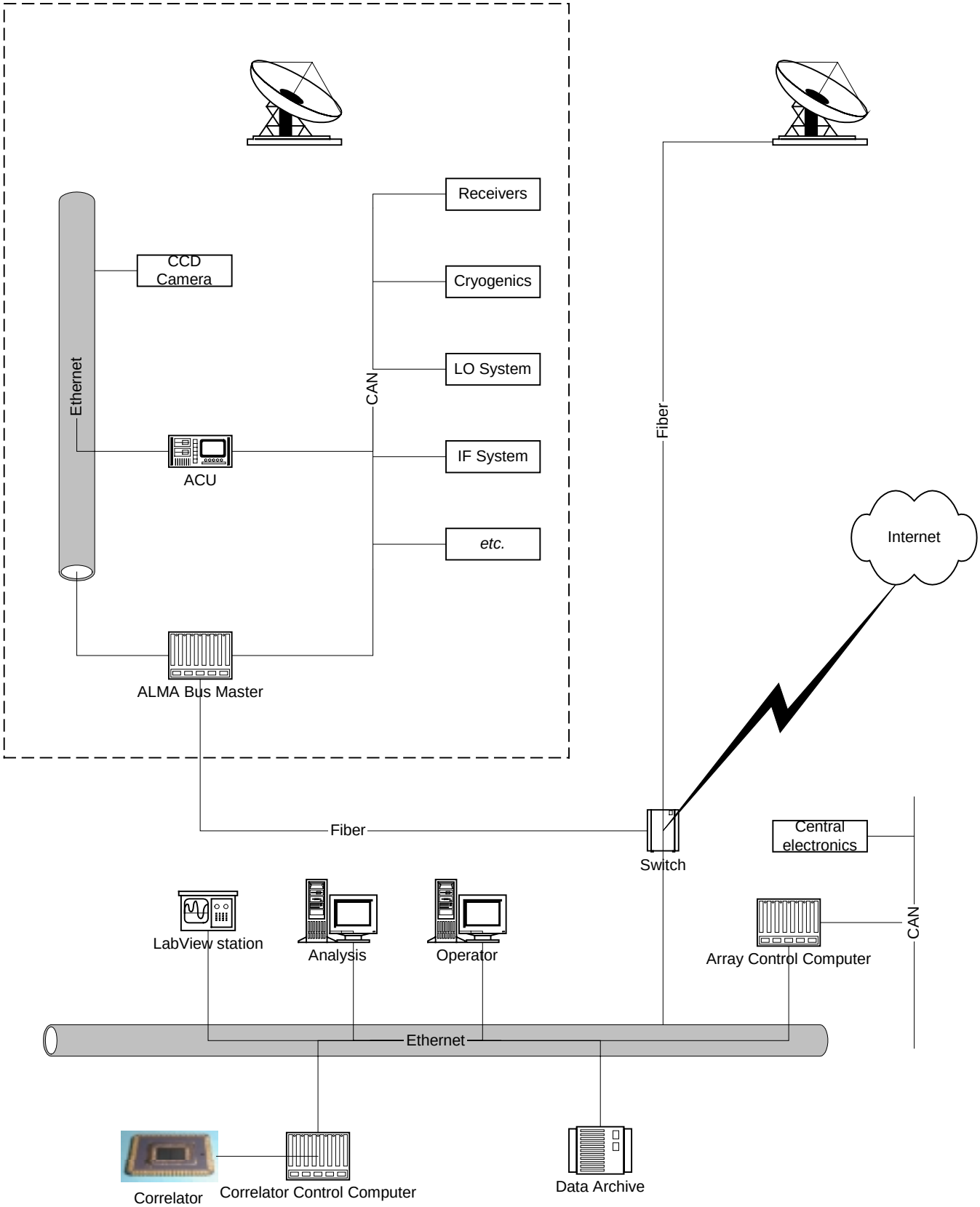
That is, the role of the common software group is to provide tools and libraries that implement common solutions. This enforces standards and similarities in applications, which then become really maintainable by integration and support team.

Where possible, we expect the common software will wrap existing functionality into a consistent interface. For example, rather than writing FITS code from scratch we will almost certainly reuse existing FITS code (for example, the well-known FITSIO library).

Capabilities anticipated for the common software include:

- Communications and remote execution (provided by CORBA).
- Alarms and logging.
- Event handling.
- Scripting support.
- Various utilities (astronomical calculations, FITS, mathematics, *etc*.)

A discussion document describing some desired capabilities and implementation considerations is described by Chiozzi (1999).


Receivers
CCD Camera
Cryogenics
LO System
Ethernet
CAN
Fiber
ACU
IF System
etc.
Internet
ALMA Bus Master
Fiber
Switch
Central electronics
LabView station
Analysis
Operator
Array Control Computer
CAN
Ethernet
Correlator
Correlator Control Computer
Data Archive

11.4 Control Software

For more details on considerations related to Control Software, please see the "Monitor and Control" chapter of this project book.

11.4.1 Device interface

Most devices will be monitored and controlled through a CAN bus connection. CAN is a message-based deterministic fieldbus first used in automotive applications. CAN is inherently a peer-to-peer network; however, we are using it in a master slave fashion using a non-standard protocol to guarantee message delivery time for all message priorities. We have added a non-standard reset line to the connector and cabling to allow hung devices to be reset remotely. Running at 1Mbps we can achieve a throughput of more than 2000 messages per second and more than 600 kbps data transfer after subtracting framing and other overheads. Each CAN bus may contain 64 nodes, and each node is assigned 2^{18} messages. Stauffer (1999) has enumerated the M&C points, with their associated rates, anticipated for ALMA, and Brooks (1999a) describes some further requirements.

Device designers may choose to use either a standard circuit, which provides some A/D, D/A, digital, serial, and I2C I/O, implemented with a Siemens C167CR microcontroller. Device designers may choose to lay the circuit out directly on their own boards, or they can use it via a standard daughtercard. If they do not want to use the standard circuit, they must only conform to the protocol, which is available from the ALMA software group as (relatively) portable "C" source code.

The M&C connector, protocol, and standard circuit are defined by Brooks (1999c). Brooks and Glendenning (2000) provide the reasoning for some of the design choices, which is described in more detail in a memo by Brooks (1999b).

The CAN bus is mastered by a general-purpose computer. For lab testing this mastering can be done with a Windows PC running LabView; in the field we will use a VME PowerPC based system running VxWorks.

At the antenna the CCD camera will not be on the CAN bus. It will be controlled, and data will be transferred, via Ethernet. The ACU supplied by the vendor will be on both CAN and Ethernet, the latter to provide for a path for software upgrades and debugging, and to allow access to "static" parameters (for example, servo loop parameters). The total power detectors might or might not be on CAN as they could take up approximately 50% of its bandwidth (however it is possible to have multiple CAN buses at the antenna).

11.4.2 Network Distribution

All devices requiring monitor and control at an antenna are connected to a bus master, which is a general-purpose computer. Most devices are connected to the master via the CAN bus; however some may be connected via Ethernet or possibly a special-purpose connection in rare instances. The bus masters in turn are connected to a central master computer (ACC – array control computer), probably through a switch, via general-purpose networking. If quality-of-service guarantees are considered to be important this protocol will be based on ATM.

At the center, there will be some electronics on the CAN bus, and there will also be general-purpose computers for operations, telescope calibration, data archiving, and engineering data analysis using LabView. There will also be a general-purpose (but real-time) computer to control the correlator. Additional computers may be used, depending on the details of where

computations take place, for example delay calculations. These computers will all be on a general-purpose (possibly switched) network.

11.4.3 Synchronization

The system will provide a prevalent 20Hz pulse. This pulse will be distributed to all devices (possibly via an unused pin in the M&C connector) that require precise timing. Commands will be delivered to those devices in advance of the next pulse, and will be "strobed" into effect on the next pulse. These commands might take the form of polynomials or table lookups for commands that need an effective rate greater than 20Hz.

Whether the commands from the center to the bus-masters at the antenna consist of real-time commands or time-tagged commands sent in advance of when the commands are required is still TBD. Similarly, whether array time is required to be known by any computer other than the ACC is still being debated.

11.4.4 Correlator

The Test correlator that will be used on the test interferometer is closely based on the design of the GBT spectrometer (hence it is often referred to as "GBT clone correlator") and the MAC spectrometer of the NRAO 12m telescope, modified to provide delays and cross-correlation capabilities. The software that will be used is thus largely based on the existing software developed by J. Hagen of NRAO 12m operations, with modifications made as necessary to allow the correlator to be used in an interferometer, and in a new control system.

Pisano (2000) describes the design of the test correlator software. It is hoped that this design can carry forward to the Prototype and Baseline correlators.

The data rates of the Prototype and Baseline correlators will be very large (Pisano, 1999). Parallel DSP processing system and Beowulf cluster solutions have been proposed as solutions for carrying out the required processing. While not necessary for the Test correlator, we might want to prototype such a solution with it.

11.5 Telescope Calibration

Numerous telescope calibrations will be required for the test interferometer. The approach adopted is to reuse existing software for this purpose whenever possible. This will require that the control system be capable of producing data in the appropriate formats.

We expect the bulk of the telescope calibration (*e.g.*, baseline determination, radio pointing) to come from the existing IRAM software suite. Optical pointing determination software will be based upon existing NRAO 12m software, and the pointing analysis will be done using the TPOINT software of P. Wallace. Single-dish holography might require new development. Holography requirements are outlined in a memo by Glendenning (1999).

11.6 Post-Processing

No post-processing development is anticipated for the test interferometer. Post-processing will take place in existing systems. Data will have to be written in appropriate formats, UVFITS for interferometric data, and possibly SDFITS for single dish data.

While some algorithm development may be occurring during the single dish tests, that development will not be the responsibility of the computing group. Instead it will be undertaken by the science/imaging and calibration groups, or outside of the ALMA project altogether.

11.7 References

The precise URL's for the following documents are subject to change as they become reviewed by the joint project, however they may all be found under the software development pages of the ALMA web site: http://www.alma.nrao.edu/development/computing/index.html.

Mick Brooks, *ALMA Monitor and Control Bus Requirements*, 1999-06-02.

Mick Brooks, *ALMA Monitor and Control System*, 1999-06-07.

Mick Brooks, *ALMA Monitor and Control Bus Draft Interface Specification*, 1999-12-09.

M. Brooks, B.E. Glendenning, *M&C Frequently Asked Questions*, 2000-01-11.

G. Chiozzi, *ALMA Common Software Feature List (prep. 2)*, 2000-01-15.

B.E. Glendenning, *Holography Software Development for the MMA*, 1999-04-15.

G.S. Harris, F. Stauffer, B.E. Glendenning, *Suggested Software Engineering Practices, ALMA Phase 1*, 1999-11-18.

J. Pisano, *ALMA Correlator Output Data and Computer Processing Rates*, 1999-11-12.

J. Pisano, *ALMA GBT Clone Correlator Control Computer Software Design*, 2000-01-04.

F. Stauffer, *Monitor and Control Points for the MMA*, 1999-05-11.