MMA COMPUTING

Brian Glendenning
Last changed October 28, 1998

Revision History:
October 28 1998: Summary, correlator delays
August 20 1998: Minor update (GNATS and readback for write-only devices)

Summary

This chapter describes the operational software required for the MMA. This includes real-time and near-real-time software to monitor and control hardware devices, software to schedule the array, software to format the data suitably for post-processing, software to archive and restore the data, software to perform fundamental calibrations (*e.g.*, pointing) required to operate the array, commissioning software (*e.g.*, holography), and software to implement a near-real-time imagine pipeline. It does not generally include post-processing software, firmware which is "inside" the device (possibly excepting the correlator), or engineering test software which is not needed during operations.

Table 12.1 MMA Computing, principal requirements

Sustained data rate, science data	10 MB/s (first light)
Image pipeline	First-look images produced automatically for common observing modes for UV-data rates up to 1MB/s
Dynamic scheduling	Nearly automatic scheduling of the MMA, accounting for current weather and other conditions, to optimize the scientific throughput of the array.
Archiving	Networked archive of all MMA raw science data and associated calibration data and derived data products.

Table 22.2 Principal computing milestones, D&D phase

M&C draft interface specifications	1999/06/01
Preliminary design review	1999/06/30
Critical design review (test correlator)	1999/09/01
Critical design review (single dish antenna test system)	2000/03/01
Critical design review (M&C)	2000/03/31
Deliver single dish antenna test system	2001/03/01

12.1 Introduction

This document describes the computing environment required for correct operations of the MMA. It includes the real-time software to control and monitor the array, as well as other software required for the array to be operated as an astronomical instrument (for example, scheduling). Post-processing software is described in another chapter of this project book, as are monitor and control (M&C) hardware considerations and holography requirements.

The MMA will undergo two stages of development, a Design and Development (D&D) stage, to be followed by a construction stage, possibly with another partner. The brief of the computing group during the D&D phase is as follows:

1. Write all software that is necessary to operate and test the hardware created for the D&D phase of the MMA. In particular, software required to measure and test the antenna is of the highest priority. If necessary this software can be thrown away after the D&D phase, although it would be preferable if this was not required.
2. A paper design and implementation plan for all software that will be required during construction and initial operation of the MMA.
3. Prototyping of software elements required for MMA construction or operation whose design or implementation technology is problematical.

An important principle is that we should use other people's software whenever possible, and if we can't reused their software directly, we should attempt to reuse their design ideas. As a rule of thumb, for software $effort \sim size^3$, so there is a tremendous incentive to fight the "not invented here" (NIH) syndrome unless adopting that software truly does do violence to the integrity or maintainability of the system.

We intend to write software that emulates critical hardware (for example, the antenna) before that hardware is ready. We have not yet decided whether to create a software-only emulation, or whether it should be plug-compatible with the hardware it is emulating.

This document is not a design document. No detailed analysis has been undertaken yet. In many instances the detailed requirements are not yet known. In general, this chapter contains some requirements at the highest levels, and some discussion of implementation choices that might be taken to fulfill those requirements.

The most important thing to identify for MMA computing is a mechanism to obtain detailed specifications for the software and hardware. While these will likely be self-generated from within the project for the real-time control aspects of the MMA, it is vital that a mechanism be found for obtaining astronomy-level specifications from the community of potential users of the array.

12.2 Computing Standards

Some project-wide standards for MMA computing will be required. It is important that these standards be useful, not merely bureaucratic.

We must decide if a formal software design methodology is useful; and if so whether a tool to support it is worth purchasing. While there are many choices for a design methodology, the most promising is probably the Unified Modeling Language (UML) developed by a number of software methodology luminaries. If we decide not to adopt a formal methodology, we need to define how we want to document the design (this alone might be reason enough to adopt a methodology).

We need to choose which operating systems the MMA software will be built upon. For the real-time operating system we shall almost certainly choose one of VxWorks or real-time Linux. The former has been used widely within the observatory (GBT, VLBA, Tucson) and is a mature, well-supported commercial product. It is also relatively expensive. RT Linux is currently being investigated in Tucson for two 12m experiments (focus and tracker computers). It is a much simpler RT environment (no debugger, for example), but offers the ability to tightly couple a full

Linux programming environment on the other "side" of the RT interface. For the non-real-time systems, the obvious choice is between a version of Unix and Windows NT. I believe we will face fewer problems if we adopt Unix for the D&D phase. We might well want to revisit this choice before construction.

We need to choose computer languages for the software we write ourselves. A mixture of C and C++ for the control software (C for real-time, C++ for non-real-time) might be appropriate for most of the MMA backbone software. We might well use Java and/or a scripting language such as Glish or Tcl/Tk for GUI's. We should be prudent and try to keep the number of languages we require to be as small as possible for maintenance reasons. Software that we adopt might require that languages be available that we do not ourselves use, for example, Spike (Miller, 1996), is written in Lisp.

Whatever languages we choose, it is vital that all software and associated documentation that we write which is required for operating the MMA be under the control of a version control system. This includes software which is required to test the MMA at the site (*i.e.*, not test software that is only run in the lab). Concurrent Versions System (CVS) is widely used and appears to be suitable. It supports distributed development. The GNATS system appears to be suitable for handling bug reports and enhancement requests.

It is desirable that the programmer-level documentation be extracted from the source code – if the documentation is near the source code it reduces the likelihood that the documentation will become stale. Many tools are available to do this.

We have to define what, if any, code acceptance procedures and automatic test procedures we want to adopt for the MMA software code base. At the minimum we should probably require that new code be reviewed for completeness of documentation, and that optional automatic unit-test procedures are available.

12.3 Real Time Components

The MMA computing design should minimize the software that runs in real-time systems. Experience shows that the real-time systems are replaced more slowly than general-purpose computers. In particular, as much of the data-handling system as possible should be in general purpose computers to allow the throughput of the MMA to be increased by replacing general computer infrastructure.

12.3.1 Monitor & Control

As described in the M&C chapter of this project book there is a fundamental design choice which has to be taken: does every antenna have a computer or not? This fundamental architectural question which must be decided early in the D&D phase since many decisions will flow from it.

There are a number of general features that the M&C system should support however which are independent of the above choice.

- The software is less brittle if there is a consistent interface to devices. For example, changes to the hardware interface should not necessitate software changes.
- Operator interfaces should not be considered a last minute detail. Unsuitable interfaces can limit the operational efficiency of the telescope, and are hard to retrofit. A system

that can avoid displaying error cascades (where a faulty component induces many other errors), like the system implemented at OVRO is helpful.

- The ability to remotely monitor and control is extremely valuable (*e.g.,* Scott, 1998). Security is clearly a concern in such systems.
- The MMA will have to take a decision about whether monitor data is ephemeral, or whether it should to be archived to allow for long-term monitoring of the health of the instrument, and to allow astronomers to make post-observation modifications to their data based on what they find in the monitor data. Whatever the lifetime of the monitor data, a database to contain it will either have to be defined or bought. Although relational database systems appear to be natural for this application, there are efficiency concerns that should be investigated before taking this decision.

All of the above argue for a middleware level that isolate the above items from some of the hardware and communications details. One possibility is to use an Object Request Broker (ORB), probably The Ace Orb (TAO) which is a CORBA standard ORB which offers Quality of Service (QoS) guarantees to allow it to be used within a real-time environment. Although there is much to recommend this technology, it may be premature to adopt it.

Another possibility is the Experimental Physics and Industrial Control System (EPICS) which has been under development in the accelerator physics community for about 10 years, and has been used to good effect in various astronomy telescope control systems (*e.g.,* Gemini). It offers us a framework for the entire M&C software subsystem (including things like alarms, plotting, and monitor screens). It does however require that we use VxWorks for the RTOS.

Another possibility would be to use the device manager/RPC++ subsystems designed in for the GBT.

Whichever middleware solution is adopted or implemented, it is important that the hardware and software protocols are not so elaborate that they preclude engineers from "plugging" devices into something like LabView for testing in the lab. It is unlikely that engineers will want to deal with the details inherent in the full M&C software system.

It should be possible to read any value which can be set. If the underlying device is inherently write-only, a software readback should be implemented.

12.3.2 Correlator (Data Rate)

Perhaps the single most important number for the MMA software is the maximum data rate the MMA must support initially. The data rate of the MMA will be approximately:

$$10\ (N_{antenna}/40)^2 \times (N_{channel}/1024)\times(N_{pol})\times(1s/t_{dump})\ \text{MBytes/s}$$

Rupen (1997) makes the case that on-the-fly (OTF) synthesis imaging of astronomical sources could quite plausibly use 25ms correlator dump times (for a 10m antenna), leading to data rates of 100's of Mbytes/s.

It does not appear to me to be prudent to commit the MMA to be able to sustain such data rates. While they might be achievable through the use of enough parallel I/O channels, the entire software system (and supporting hardware) would have to contort itself to support such data rates, and the resultant system might well end up being brittle.

Clearly the correlator hardware should be designed in such a way that data can arrive at interface computers at these sorts of data rates, so that the data rate can be increased by changing interface computers rather than correlator internals.

It appears that data rates of a few 10's of Mbytes/s should be achievable with available bus technology and disk arrays. If rates higher than this are required at first light, then during the D&D phase high-data rate solutions should be investigated at high priority (note that 10 Mbytes/s was the recommended maximum rate of the MMA Software Working Group (MMASWG) (Scott *et. al.*, 1996)).

For the test (2-antenna) correlator, delays may be applied in software rather than hardware.

12.3.3 Antenna

Sophisticated antenna control is critical for the D&D phase. For example, holography will impose stringent requirements on the supported single-dish modes.

Antenna control is another area which is strongly affected by how computers are distributed in the array. For example, with a computer in each antenna one would be more inclined to update simple antenna positions frequently, whereas with a central computer you would likely send commands less often, so the command would probably be more complex (*e.g.*, contain polynomials).

In any event, basic antenna control (point, track) appears to be an area where we can reuse code or designs, from other instruments, both from within the observatory and from without.

Since the single-dish modes for the D&D phase are in service of holography of the prototype antenna, and since that effort will be building upon the experience of NRAO/Tucson personnel, the single-dish modes should probably be built by replicating the "experience" built into the 12m observing scripts.

12.4 Data Production

A thorny question is whether the first data "emitted" (that is, available to other programs) by the MMA should be in FITS format or a local format.

There are good reasons to choose a local format. FITS is a fairly rigid data format. It can be awkward to fit data with different "shapes" together, and there is no agreed upon standard way to represent hierarchies in FITS. Quick debugging or display programs are arguably harder to write for FITS files than with other formats.

However there is a compelling reason to choose FITS. It keeps the system honest – it ensures that data which is read from the archive will not be second class data – it will contain all required information. (I assume there is no dispute that archived data will be in FITS format).

The fundamental atom of data production will be a multi-HDU FITS file containing the data for some time range. The data, synthesis and total power, should be written in the format defined Diamond *et. al.* (1997) augmented as necessary for the MMA. If total power data cannot be represented readily in this format, they should be written in the emerging SDFITS format.

These FITS data files will be created by combining data blocks from the correlator with monitor data. In keeping with the principle that the real-time systems be kept as simple as possible,

creating the FITS files should be the responsibility of a general-purpose computer. If the monitor data volume is small compared to the volume of science data, we should consider embedding all monitor data for a time range in its FITS file.

The MMA should maintain a dictionary of all FITS keyword names and column names to ensure that they are used uniformly.

12.5 Offline Processing

It is important to ensure that MMA data can be reduced by astronomers at their home institutions by at least one software package. The MMA should obtain a commitment from the AIPS++ project that it will meet those needs. The MMA should provide to AIPS++ a set of requirements for the offline system, particularly those that involve new algorithm development or high data-rate or parallel processing. While the AIPS++ project will be responsible for implementing most processing (*e.g.*, mosaicing), very instrument-specific operations may be implemented by MMA staff – at a minimum, the filler.

Astronomers are of course not precluded from using other systems, or writing their own programs to do specialized processing.

12.6 Service Data Processing

A fundamental decision for MMA operations is how much data processing will be performed for the observer. At one extreme the observatory merely provides raw data and instrument calibrations, at the other extreme the observatory undertakes to provide the observer fully processed images which have been carefully scrutinized for flaws by humans.

The recommendation of the MMASWG is that final, or near-final, quality images be produced for the user by MMA software and staff.

This decision implies that a larger number of data analysts may be required for the MMA than for other NRAO instruments. This need may be alleviated if adequate automatic tools for data editing (*e.g.*, interference excision), data calibration, and image formation can be created. Automatic imaging of this sort is going to be investigated at NCSA during the D&D phase as an MDC project.

Automatic processing of data at the average data rate of the MMA will probably require parallel processing during the initial operations of the MMA. The MMA should obtain a commitment from the AIPS++ project that it will be able to utilize such hardware.

12.7 Scheduling

To maximize the scientific throughput of the MMA, dynamic scheduling will be required. In brief, projects will be observed only when conditions are favorable for that project (*e.g.*, phase stability and airmass).

As Holdaway (1997) describes, this implies that the schedule will be created very nearly in real-time using many criteria, including:

- Scientific priority of the project.
- LST.

- Environmental measurements:
  - Current opacity of the atmosphere (as a function of frequency).
  - Current phase stability.
  - Weather measures, especially wind statistics.

  A history should be kept for all these measurements so that the scheduler can make decisions based on, *e.g.*, how often these conditions are likely to recur before the next array configuration change.
- Required UV coverage (while the snapshot coverage of the MMA is sufficient for compact configurations, larger configurations require more hour-angle coverage (Holdaway, 1998)).
- Calibration will impose additional requirements, for example parallactic angle coverage for polarization calibration (Cotton, 1998). Calibration observations should be shared between projects when possible. However projects which require extra calibrations should be charged by the scheduler for them.
- Array efficiency (slew time, projects that share calibration, linked observations that must be executed at about the same time).

This implies that the raw data for a single project will consist of independently calibratable chunks, possibly spread over many days.

Fortunately for the MMA there is experience with implementing dynamic scheduling. Wright (1997), has implemented a scheduling system at BIMA which incorporates some measurements of the atmosphere into the scheduling process. The ESO VLT telescope is implementing a "Data Flow System" which appears to implement many of our scheduling requirements (Silva and Quinn, 1997). The STScI has released Spike, a portable scheduling engine which we may want to adopt (Miller and Bose, 1996).

We shall have to provide the user with proposal and scheduling tools. We will have to decide whether the user should merely specify science objectives (*e.g.*, point source S/N) and rely on the MMA system (software and personnel) to schedule the observation (including calibration), or whether the user should create a more explicit schedule of observations. If the user specifies scientific objectives the system has to be able to measure them, which is in general a difficult problem. Measuring the number of "good" hours spent on project targets and calibrators would be much more straightforward.

Although dynamic scheduling will probably be the most common scheduling mode, standard queue observing must be supported (*e..g*, VLBI, transient sources). In addition, there may be circumstances where real-time control of the array is required.

There could be feedback between the scheduling system and array configuration changes. In principle, configuration changes could be driven by the scheduler.

12.8 Archiving

It is assumed that all data products from the MMA shall be archived in perpetuity, and made available to researchers over the network. If the average data rate of the MMA is 1Mbyte/s, and if the data product volume is dominated by the raw data, then a few 10's of TB per year will have to be archived. While this is considerable, it is also manageable (a few thousand double-sided, double-density DVD's, for example). The choice of archival media and hierarchical storage system should be deferred until MMA construction nears completion.

The main archive repository will probably not be at the MMA site in Chile (the demands it will place on general networking infrastructure will be considerable). As noted by the MMASWG (Scott *et. al.*, 1996), if we cannot achieve sufficient network bandwidth to Chile we may need to ship physical media from the MMA site in Chile.

We might need to support replication of the archive at one or more locations to relieve, for example, transcontinental network congestion. Replication should be built into the architecture of the archive. Even if there is only one active data center, the data should be physically replicated and stored in an offsite location for data recovery and safety reasons.

Access to data is not sufficient of course. MMA observations are apt to be complex, with observations and calibrations spread out widely in time. Also, some or most observations will have images of varying sophistication associated with them. This implies that fairly sophisticated meta-information, largely gleaned from the observers proposal and the actual schedule, will have to be maintained to allow the desired data to be found in a straightforward way.

The archive will be used both for post-observation data retrieval, and to inspect data while observations are proceeding. In this latter case, email notification to the proposal authors should be implemented so they can inspect their data as it becomes available.

The MMA archive system appears to share many features with other astronomy archives. For example, the Sloan Digital Sky Survey (SDSS) archive has an interesting three-tiered archive structure, and a decomposition into an operational and science archive (Brunner et. al., 1996). Another example is the STScI archive re-engineering effort (Hanisch et. al., 1997). We should take advantage of such expertise.

The archive must protect proprietary data rights and not release data to the general community until time limits have expired. Mechanisms to enforce these rights must not be too burdensome.

Archiving is going to be investigated during the D&D phase at NCSA as an MDC project.

Acknowledgements

Although the above is still woefully incomplete, it would be even more so without a number of discussions I have had with various people. I would like to thank: Tony Beasley, Barry Clark, Tim Cornwell, Darrel Emerson, Jeff Hagen, Ron Heald, Mark Holdaway, Frazer Owen, Michael Rupen, Steve Scott , and Al Wootten for helpful comments and advice.

References

R.J. Brunner, I. Csabai, A. Szalay, A.J. Connolly, G.P. Szokoly, “The Science Archive for the Sloan Digital Sky Survey”, ADASS V, 1996.

W.D. Cotton, “Polarization Calibration of the MMA: Circular versus Linear Feeds”, MMA Memo #208, 1998.

P.J. Diamond, J. Benson, W.D. Cotton, D.C. Wells, J.D. Romney, G. Hunt, “FITS Format for Interferometry Data Exchange”, VLBA Correlator Memo #108, 1997.

R.J. Hanisch, F. Abney, M. Donahue, L. Gardner, E. Hopkins, H. Kennedy, M. Jyprianou, J. Pollizzi, M. Postman, J. Richon, D. Swade, J. Travisano, R. White, “HARP – The Hubble Archive Re-Engineering Project”, ADASS VI, 1997.

M.A. Holdaway, “High Level Computing Information Flow for the MMA”, MMA Memo #167, 1997.

M.A. Holdaway, “Hour Angle Ranges for Configuration Optimization”, MMA Memo #201, 1998.

Glenn E. Miller, Ashim Bose, “Portable Astronomical Scheduling Tools”, ADASS V, 1996.

Michael P. Rupen, “The Astronomical Case for Short Integration Times on the Millimeter Array”, MMA Memo #192, 1997.

Steve Scott, Ray Finch, “The New User Interface for the OVRO Millimeter Array”, ADASS VII, 1998.

S. Scott, D. Emerson, R. Fisher, M. Holdaway, J. Knapp, L. Mundy, R. Tilanus, M. Wright, “MMA Computing Working Group Report”, MMA Memo #164, 1996.

D. Silva and P. Quinn, “VLT Data Flow Operations News”, ESO Messenger #90, 1997.

M.C.H. Wright, “DYNAMIC SCHEDULING: Implementation at Hat Creek”, BIMA Memo 60, 1997.