EVLA Monitor and Control

Software Design

Version 1.0.0

Revision	Date	Authors, Contributors	Description of Changes
VerCDR 1.0.0	22Nov2006	Bill Sahr, Barry Clark, Hichem Ben Frej, Walter Brisken, Rich Moeser, Bryan Butler	Original Version

VerCDR 1.0.0
Section 1 is chiefly the work of the author/editor, Bill Sahr. My thanks to Barry Clark and Dave Harland of NRAO and to Brent Carlson of DRAO for reading and commenting on section 1.

The material in section 2 is drawn from multiple sources. The text on the Interim Observation Executor was reviewed and corrected by Barry Clark, with the text on Calc directly contributed by Barry. The material on the CMP is an excerpt from an as yet unpublished document by Hichem Ben Frej. The material on IDCAF is from a discussion with and a diagram supplied by Walter Brisken, the author of the software. The section on alerts is an excerpt from a soon to be published document written by Rich Moeser and Bryan Butler.

Of course, the attributions made above are only those that stand out most strongly or can easily be assigned to a single source or a particular occasion. Numerous conversations, discussions, and other exchanges with many individuals over long periods of time have contributed to the content of this document.

Master document:
/users/bsahr/EVLA/architecture_design/MandC_design/MandC_design_doc/
/VerCDR/MC_Design_VerCDR.doc

Subdocuments:
\\filehost\bsahr\EVLA\architecture_design\MandC_design\MandC_design_doc\VerCDR\

Overview.doc

SelectedComponents.doc

Last saved: 11/22/2006 20:16:00

Table of Contents

Table of Figures

1 Overview

1.1 *Scope of the EVLA Monitor and Control System*

The EVLA high level software design is presented in the document "EVLA High Level Software Design", by Tom Morgan, Kevin Ryan, Ken Sowinski, and Boyd Waters, dated June 7, 2004. This document is available on the EVLA Computing Working Documents web page:

http://www.aoc.nrao.edu/evla/techdocs/computer/workdocs/index.shtml

It is document # 33. The design presented in that document is meant to be a high level design for all EVLA software. It is not limited to EVLA monitor and control.

The High Level Software Design document presents a diagram of the data flow within the system. That diagram, with a few later modifications, is reproduced in Figure 1, given on page 2. The diagram shows two domains that are relevant to the monitor and control system - the Online Domain and the Real-Time Domain. The elements of those two domains are as follows:

- Online Domain
  - Observation Scheduler
  - DCAF (Data Capture and Format, creates archive records)
  - Telcal (Telescope calibration, both old & new)
  - Quick Look Pipeline
  - User Interfaces (not shown in the diagram)
- Real-Time Domain
  - Observation Executor
  - Antenna Monitor and Control (AMCS)
  - Correlator Monitor and Control (CMCS), including the Correlator Backend (CBE) and a recently added element known as the Fast Formatter

The scope of the EVLA Monitor and Control System is taken to be all of the elements in the Online and Real-Time Domains, with the exception of the Observation Scheduler and the Quick Look Pipeline.

Note that no mention is made of a visibility data archive. The visibility data archive is not considered to be a component of the EVLA Monitor and Control System. The EVLA Monitor and Control System is the data source for the visibility data archive, and it must include an interface to the archive. That interface is considered to reside in the DCAF component.

1.2 *Components of the EVLA Monitor and Control System*

The high level design and its accompanying data flow diagram provide a very large granularity view the EVLA Monitor and Control System. While that view is useful for placing the monitor and control system in context, it provides no detail and no insight into the structure or components of the monitor and control system.

EVLA Top Level Data Flow Diagram, with Post Processing

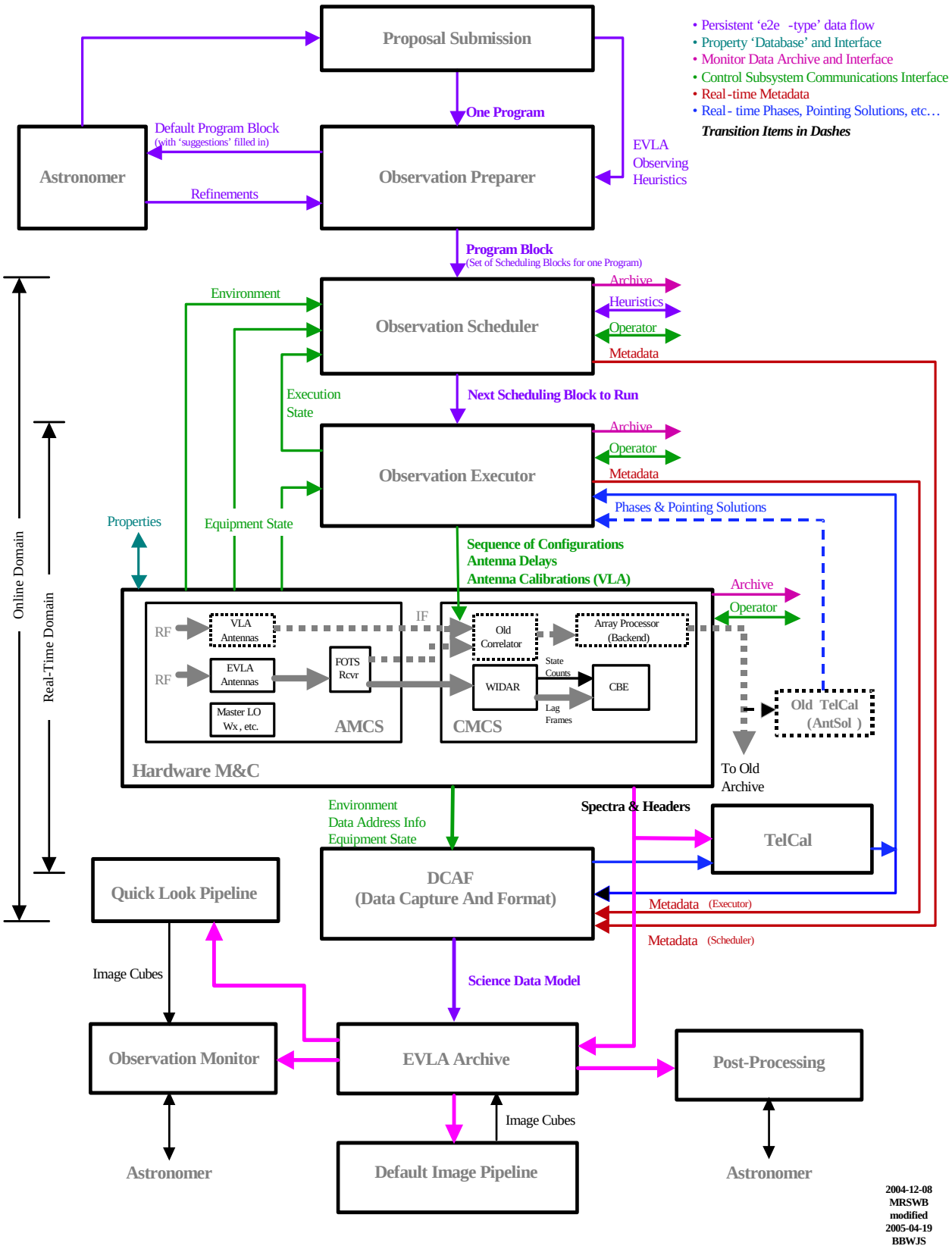


Figure 1 EVLA Top Level Data Flow

The components of the EVLA Monitor and Control System are:

- The Observation Executor, Interim & Final versions
- Obs2script
- Antenna Monitor & Control (AMCS), consisting of
    - Monitor & Control of EVLA Antennas
    - Monitor & Control of VLA Antennas
- Correlator Monitor & Control (CMCS), consisting of
    - Monitor & Control of the VLA Correlator
    - Monitor & Control of the WIDAR Prototype Correlator
    - Monitor & Control of the WIDAR Correlator
- Telcal (Telescope Calibration) - both VLA Telcal and EVLA Telcal
- Flagging
- DCAF - both VLA DCAF and EVLA DCAF, including the interfaces to the VLA Archive & the EVLA Archive
- User Interfaces, including
    - The Array Operator Screen
    - EVLA Antenna Screens (module/subsystem screens)
    - VLA Antenna Screens
    - VLA Correlator Screens
    - WIDAR Correlator Screens
    - Data Quality Displays for data coming from the VLA Correlator and the WIDAR Correlator
    - Data Quality Displays for Telcal, both VLA Telcal and EVLA Telcal
    - A Device Browser
- Parameters Database (System Parameters)
- Monitor Data Archive
- The Alerts Subsystem
- Operations Software

1.3 *Transition System vs. Final System*

If one wishes to understand the architecture, design, and development of the EVLA Monitor and Control System, it is important to understand that there will be two major versions of the EVLA Monitor and Control System – a Transition System and a Final System. The purpose of the Transition System is to bridge the gap between the Modcomp-based VLA Control System, and the final version of the EVLA Monitor and Control System, while maintaining the operational capabilities of the array. The Transition System began with the development of software sufficient to the task of supporting the design and development of the hardware subsystems that comprise an EVLA antenna. It will end only when all VLA antennas have been converted to EVLA antennas, the WIDAR correlator has been commissioned, and the VLA correlator has been retired.

The current state of the Transition System (as of September 2006) is that it supports the use of EVLA antennas in scientific observations. To do so, it must interoperate with the VLA Control System. The next major milestone for the Transition System is to reach a state of development that will permit it to completely replace the VLA Control System, allowing the Modcomp-based VLA

Control System to be retired. The current schedule calls for retirement of the Modcomps and the VLA Control System sometime in Q2 2007.

The Transition System will be required to monitor and control a variety of old and new hardware, including EVLA antennas, VLA antennas, the current VLA correlator, and the WIDAR prototype correlator. It will need a means by which the output of the VLA correlator can be distributed to those software components that require this data, and it must be able to form and write VLA format archive records. Additionally, it must serve as a platform for the development of the final version of the EVLA Monitor and Control System - all without loss of operational capability. To achieve these goals the Transition System must implement the full complement of capabilities now contained in the VLA Control System, and extend those capabilities to include the configuration, monitor and control of hardware never seen by the VLA Control System.

As currently conceived, the dividing line between the Transition System and the Final System is the introduction of the WIDAR correlator, and its integration into the EVLA Monitor and Control System. The WIDAR correlator will present new and major challenges to all areas of the EVLA Monitor and Control System – user interfaces, configuration, monitor and control, data products, data volume, and data rates. It will require the development of software components that either have no counterpart in the Transition System, or that are so different in the requirements for capabilities and performance that there will be little or no carryover from the Transition System.

While some of the components of the Final System will be quite different from their analogs in the Transition System, the underlying architecture of the Transition System will, for the most part, be sufficient as an overall structure for the Final System. In the Final System, subsystems will become more elaborate, complex, and capable, but the relationships among the subsystems will remain substantially the same. The change from the Transition System to the Final system will be incremental and is best viewed as a continuum rather than as a distinct and abrupt transformation. While it is true that the WIDAR correlator will never accept data from VLA antennas and the output of the WIDAR correlator will never be written in the VLA archive format, there may be a period of time when VLA antennas, the VLA correlator, EVLA antennas, and the WIDAR correlator are all supported simultaneously by an EVLA Monitor and Control System that could be viewed, at that point in its development, as a mature Transition System or as a nascent Final System.

1.4 *Description of the EVLA M&C Components*

1.4.1 The Observation Executor

The Observation Executor is the central high level component of the system. It accepts a control script and executes it. Execution of a control script is equivalent to conducting an observation. Execution of a control script implies control of hardware. The Interim Observation Executor (a component of the Transition System) sends commands directly to the hardware. The final version of the Observation Executor may send commands to device servers, and the device servers will, in turn, control the hardware.

The Observation Executor is the locus for a great deal of the higher level functionality required by the monitor and control system. High level configuration and control of EVLA antennas, VLA antennas, the VLA correlator, and the prototype and full production versions of the WIDAR correlator is all done by the Observation Executor. Reference pointing, round trip phase

corrections, phased array operation , first and second level subarraying and many other capabilities are all implemented in whole or in part by the Observation Executor.

1.4.2 Obs2script

Obs2script converts VLA Observe files to EVLA control scripts. Currently, the control scripts are written in jython. Generally, complementary changes/additions are required for obs2script and the Observation Executor. The Executor must implement methods to support a particular capability, and obs2script must be modified to recognize a request for that capability and to then invoke the appropriate methods in the Executor.

Obs2script is a Transition System element. In the Final System, a Scheduler component will send EVLA control scripts, whose content will have been specified by an Observation Preparation Tool, directly to the Observation Executor. JObserve, the program now used to produce VLA Observe files, will not be upgraded to support EVLA antennas or the WIDAR correlator. Eventually, VLA style Observe files will no longer be relevant and there will be no need for obs2script.

1.4.3 Antenna Monitor and Control (AMCS)

1.4.3.1 EVLA Antennas: MIB-based Devices

A MIB is a module interface board. MIBs are central to the monitor and control of EVLA antenna subsystems. Basically a MIB is a custom, in-house designed single board computer (SBC) that mates to antenna module hardware, providing both an interface to the outside world and computing resources that are used for monitor and control of the module. The interface to the outside world is via Ethernet and standard IP protocols – TCP/IP and UDP, both point to point unicast of packets and multicast. The interface to module hardware is chiefly via SPI (Serial Peripheral Interface) and GPIO lines (General Purpose Input/Output). The processor is a 96 MHZ TC11IB chip with 1.5 Mbytes of on-chip RAM and timer capabilities. Flash memory is available to store software images and configuration files.

A common software interface has been implemented for all MIBs. Commands use a simple <set @time device commandpoint.attribute=value> syntax. Requests for data use a simple <get device monitorpoint.attribute> syntax. A UDP port (also known as the service port) is used to receive commands. A telnet port is available for human interactions with the MIB, and a data port is used to multicast monitor data and alerts (alarms).

It is expected that a full, production version of an EVLA antenna will contain approximately 34 MIBs.

1.4.3.2 VLA Antennas: The CMP

There is a hardware element in the current VLA Control System known as the Serial Line Controller (SLC). The SLC is a point of concentration for all VLA antenna monitor data and commands. Fortunately, the SLC was designed as a two port device. For historical reasons, the second port was never used, until now. The CMP (Control and Monitor Processor) is a combination of hardware and software that interfaces to the second port of the SLC. It provides a path by which the EVLA Monitor and Control System may send commands to VLA antennas and tap into the monitor data streams coming from VLA antennas.

Figure 2 presents a diagram of the Transition System that shows several control paths, including the role of the CMP.

1.4.4 Correlator Monitor and Control (CMCS)

1.4.4.1 Monitor and Control of the VLA Correlator: The New VLA Correlator Controller

A replacement for the current correlator controller has been developed. It replaces both the current correlator controller and the (old and very obsolete) array processor that is used to perform the FFT of the raw correlated data. The new VLA correlator controller implements an Ethernet-based control path for control of the VLA correlator. The Ethernet-based control path is crucial. Since both the VLA Control System and the EVLA Monitor and Control System have access to Ethernet, the new VLA correlator controller can be controlled by either system, allowing for rollover from one to the other. Without the new VLA correlator controller, the EVLA Monitor and Control System has no path by which it can access and control the VLA Correlator.

The diagram given below (Figure 2) shows the new VLA correlator controller (bottom, right of center) and the paths by which it may be controlled by both the VLA Control System and the EVLA Monitor and Control System. The correlator control paths are shown in blue. For the VLA Control path, control originates in the Modcomp computer named Boss (upper right hand portion of the diagram). For the EVLA Monitor and Control System, control will originate in the Interim Observation Executor that will run in the system named mchost (center of diagram).

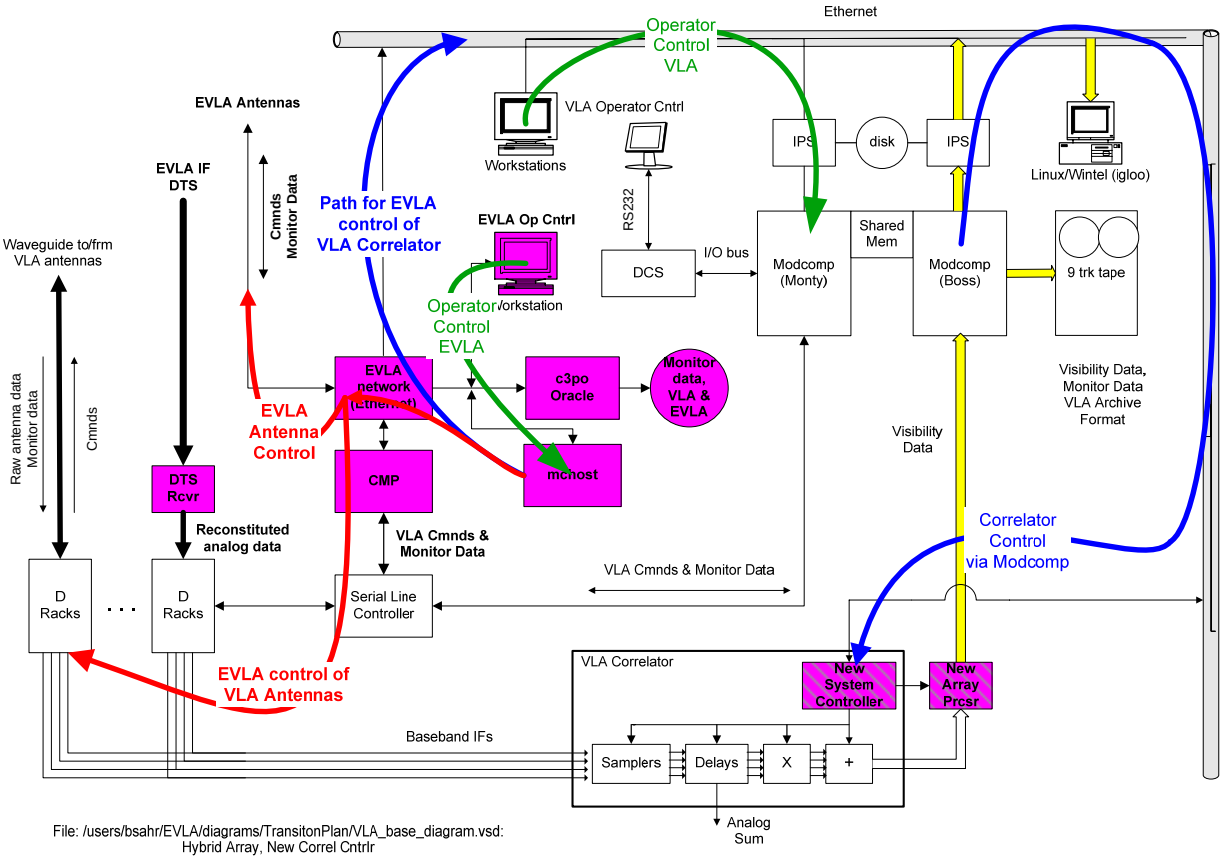


Figure 2 Transition System w CMP & Correlator Controller

1.4.4.2 VLA Correlator Data Output Path: The Visibility Pipe

The phrase "visibility pipe" is the term used to designate the means by which data will flow from the new array processor (that is part of the new VLA correlator controller) into the EVLA Monitor and Control System. The approach decided upon was to interface the output of the array processor to the ISA bus of a PC104 module that includes an Ethernet interface. After the visibility data has been loaded onto the PC104 module via the ISA bus interface it will be multicast on the Ethernet interface, making it available to any client that cares to subscribe to the multicast. The Visibility Pipe is not shown in Figure 2.

1.4.4.3 Monitor and Control of the WIDAR Prototype Correlator

Monitor and control of the WIDAR Prototype Correlator (PTC) will be responsibility of the Transition System. For on-the-sky (OTS) testing, the critical responsibilities of the EVLA Monitor and Control System will be to supply delay tracking models, and integration control parameters. The monitor and control system must also supply the equivalent of a prototype Correlator Backend (CBE) and Fast Formatter (FF), and an archiving capability for visibility data and ancillary data products (principally Station Board data products). The CBE performs long-term integration of correlator lags and the Fast Fourier Transform (FFT) of the integrated data. The CBE is a cluster system with multiple nodes. The job of the FF will be to combine the data coming from the multiple CBE nodes into a single binary data stream.

PTC configuration for initial on-the-sky (OTS) testing will be accomplished via standalone screens. As the interface to the PTC matures, it is expected that a prototype Virtual Correlator Interface (or, more accurately, the VCI and the software associated with it) will be used for PTC configuration, the transmission of models, and the reception of status information.

The software involved in monitor and control of the PTC is also taken to include the correlator board device drivers and module access handlers (MAHs) that run on embedded processors (CMIBs - correlator MIBs) located on each and every correlator board, board level GUIs that can be used to configure, test, and troubleshoot the correlator boards, system level GUIs that will be used for higher level, manual correlator configuration, and a wide variety of currently unspecified utility software that will, no doubt, be developed.

1.4.4.4 Monitor and Control of the WIDAR Correlator

Monitor and control of the WIDAR correlator will be accomplished via a suite of software executing on the Master Correlator Control Computer (MCCC). This software will include the Virtual Correlator Interface (VCI), the configuration mapper, software for the forwarding of models, and software for a number of other functions such as alert handling. Configuration data, models, and commands will flow to the MCCC from the Observation Executor. Software on the MCCC will transform these higher level specifications as needed and forward the data to the correlator Station Boards (SBs), Baseline Boards (BBs), and the Correlator Backend (CBE). Correlator status will flow from the correlator boards and CBE nodes to the MCCC and thence into the EVLA Monitor and Control System.

1.4.5 Telcal

Telcal is an acronym for telescope calibration. It refers to the software component in the EVLA Monitor and Control System that intercepts the visibility data and performs the calculations needed to derive telescope calibration quantities. The quantities derived include raw self-cal (complex

antenna gains, i.e. Antsol in the VLA Control System) reference pointing offsets, quantities needed for the calculation of global pointing model coefficients, focus settings, and delay settings.

Telcal for the Transition System (a.k.a. VLA Telcal or Interim Telcal – ITelcal) will be quite different from Telcal for the Final System (EVLA Telcal). The requirements for EVLA Telcal are more extensive, and EVLA Telcal will deal with much higher data volumes and data rates.

1.4.6 Flagging

Flagging is the process whereby the data for one or more antennas over one or more correlation integration periods is (or is not) flagged as bad based upon the detection of events by the monitor and control system. Flagging information is written as a component of the archive records for use by the post-processing software.

1.4.7 Data Capture and Format (DCAF)

DCAF is the software responsible for gathering together the relevant meta-data and visibility data (or pointers to the visibility data) and formatting this information into archive records that are written to disk. DCAF can be considered to contain the interface to the visibility data archive, even if the actual, physical deployment of the system may differ somewhat from this conceptualization.

Data from the VLA correlator will always be written in the current VLA archive format, while data produced by the WIDAR correlator will be written in some version of a Science Data Model (SDM).

1.4.8 User Interfaces

By User Interfaces is meant the means by which users will interact with the hardware and software components of the EVLA Monitor and Control System. Users include scientists, engineers, technicians, and other personnel. The purpose of the interactions will include the conduct of observations, obtaining the status of the system or its subsystems, and the diagnosis of problems. The means of interaction will include screens and displays of various sorts and command line interfaces, implemented as standalone applications, servlets, web applications, and in whatever other form proves to be appropriate.

The user interfaces of greatest interest were listed in section 1.2, Components of the EVLA Monitor and Control System. For convenience, that list is reproduced here:

- The Array Operator Screen
- EVLA Antenna Screens (module/subsystem screens)
- VLA Antenna Screens
- VLA Correlator Screens
- WIDAR Correlator Screens
- Data Quality Displays for data coming from the VLA Correlator and the WIDAR Correlator
- Data Quality Displays for Telcal, both VLA Telcal and EVLA Telcal
- Device Browser

1.4.9 Parameters Database (System Parameters)

The parameters database is to the EVLA Monitor and Control System as the system files are to the VLA Control System. It contains unvarying or slowly varying elements such as pad locations and pointing model coefficients that are necessary to the conduct of an observation. The Observation Executor reads the needed information from the parameters database. Most of the information is read when a control script is started. Some of it is read during control script execution.

1.4.10 Monitor Data Archive

The monitor data archive is the repository for data on status and state produced by various components (mostly hardware) of the EVLA Monitor and Control System. In practice, its chief use is to serve as an archive for all monitor data produced by antenna subsystems. It may also eventually serve as the repository for ancillary data products produced by the WIDAR correlator.

The monitor data archive can be conceptualized as consisting of three major components – a filler program that accepts data for transmission to a database, the actual database that accepts the monitor data, and an interface that can be used to retrieve data from the database.

1.4.11 The Alerts Subsystem

Alerts are messages that describe abnormal conditions, i.e., alarms. These conditions may be specific to a device or they may be generated by a software entity, such as an antenna object, on the basis of monitor data &/or other alerts.

The alert subsystem refers to the means by which alerts are generated, transmitted, augmented, collected, distributed, logged, and displayed.

1.4.12 Operations Software

Operations software refers to the multitudinous bits and pieces of software used to assist the array operators in the conduct of their duties. It includes, but is not limited to, software that is used for antenna acceptance after an antenna undergoing maintenance/overhaul is returned to service, antenna checkout after an antenna has been moved, startup after maintenance days, first level problem diagnosis, and many other items.

1.5 *Relationships Among the Components: Architecture & Dataflows*

1.5.1 The Transition System

Figure 3 is a diagram that gives some of the major, high level data flows in the Transition System version of the EVLA Monitor and Control System as it will exist at the time the Modcomp-based VLA control System is retired. It also gives, via the color coding and legend, some information on the status of the Transition System components as of Q4 2006.

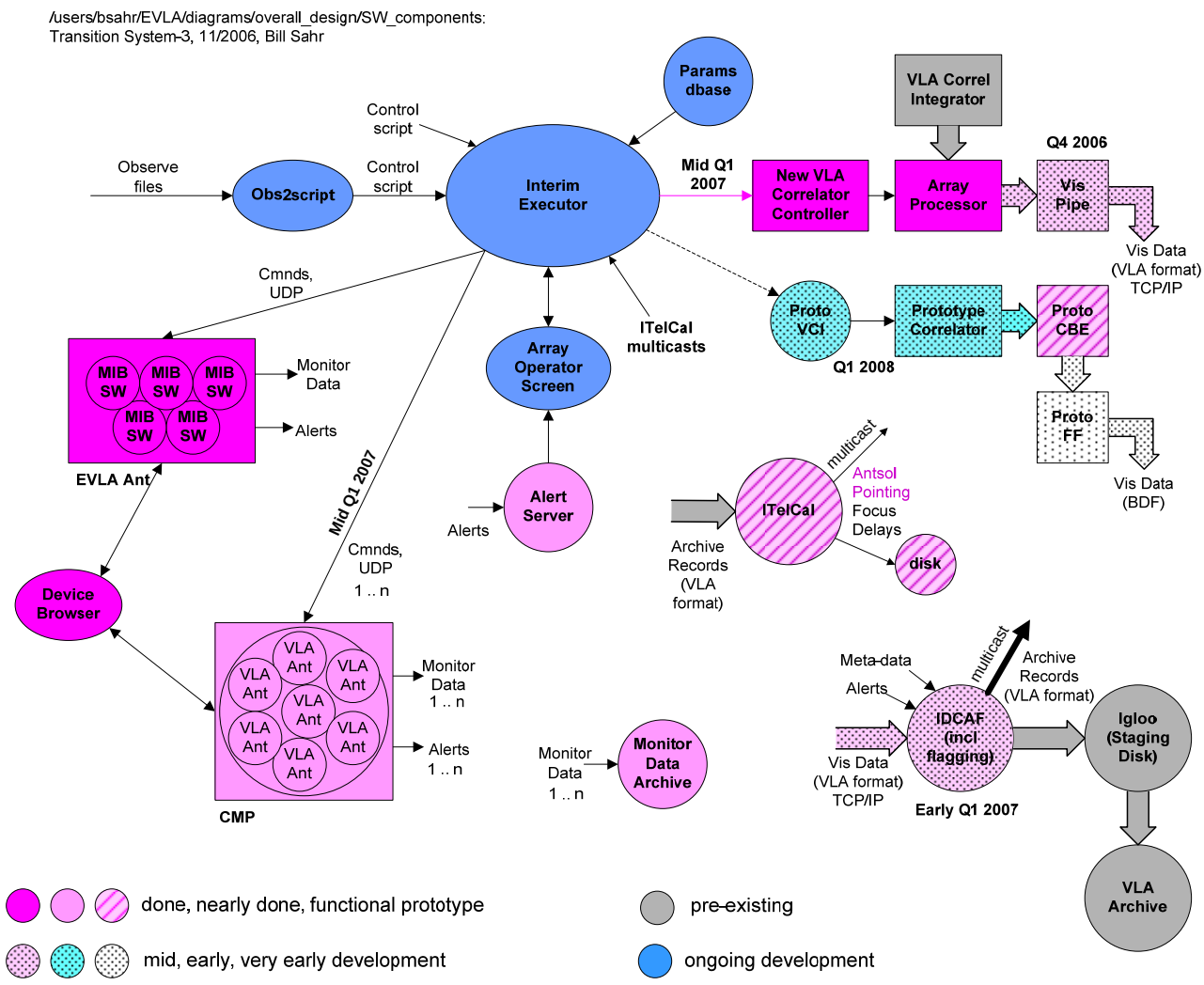


Figure 3 Transition System DataFlows

1.5.1.1 Job Submission

The process of conducting an observation begins with the submission of a control script or a VLA-style Observe file to the Observation Executor (upper left hand portion of the diagram). Job submission (and a number of other tasks) is typically done via the Array Operator Screen.

For software debugging, an Observation Executor can be started from the Unix command line. Executor components are written with a debug facility, which can be used to suppress the transmission of commands to the hardware. For selected devices, the commands can be directed to the screen. This facility can be invoked via command line execution, in effect creating a telescope simulator.

A control script can be created directly in jython, or it can be created by the conversion of a VLA-style observation file into a jython control script. A software component named obs2script is used to perform the conversion. In practice, the conversion is performed automatically by the Executor – it distinguishes between jython scripts versus Observe files, and, if the latter, automatically invokes

obs2script. Control scripts are interpreted within the Observation Executor, with data added from the parameters database as needed to complete the specification of an observation.

1.5.1.2 The Interim Observation Executor

The Interim Observation Executor (top center of the diagram) directly configures, updates, and otherwise controls the hardware. Currently, the Interim Observation Executor has the ability to directly control EVLA antennas, and is in the testing stage w.r.t. its abilities to control VLA antennas. Software development for control of the VLA correlator is expected to begin in either October or November 2006, depending upon the date at which the new correlator controller comes into regular use. The ability to generate and pass models to the Prototype Correlator (PTC) is needed for on-the-sky (OTS) testing, which is currently scheduled to begin in late January or early February 2008. The ability to configure the PTC will not be needed until that time or even later, and is dependent upon delivery of prototype MCCC software. (Please recall that initial configuration of the PTC for OTS testing is to be done via standalone screens.)

1.5.1.3 Antenna Control

Of course, the ability of the Executor to control a device assumes the existence of complementary software in the device to be controlled. For EVLA antennas, the software in question is the module interface board (MIB) software. (An EVLA Antenna containing MIBs is shown on the left hand side, center of Figure 2.) The MIB software is now in an advanced state of development. A critical design review (CDR) of the MIB hardware and software was held on October 20, 2004, and advanced prototype or production versions of the software exist for every module of any significance in the system.

(For further information on the MIB CDR, please see EVLA Monitor & Control Hardware Critical Design Review, http://www.aoc.nrao.edu/evla/admin/reviews/MIB/MIB.html.)

The MIB software provides a simple, capable, generic interface for monitor and control of EVLA antenna subsystems. Commands are sent from the Interim Observation Executor (and other sources) as UDP packets, and received at the MIB over what is termed the MIB service port. Commands to change the state of the system ("set" commands) may be time-stamped and the MIB provides a queue that can hold upto 50 time-ordered commands. Get commands may also be sent to the MIB service port. Get commands are, essentially, requests for monitor data. The requested data is returned as XML formatted messages contained in UDP packets. These messages are transmitted via the service port. There are "get commands" to request all of the devices associated with a particular MIB and all of the monitor points associated with a device, giving the MIB interface a self-describing capability.

In addition to the service port, the MIB interface also provides a data port. Monitor data intended for the monitor data archive is multicast via the data port. A typical MIB can support just under 300 monitor points. Each monitor point has an individually specifiable archive rate. To support the need of the monitor and control system for certain monitor data quantities in real-time, at update rates higher than the archive data rate, the MIB data port supports a second type of multicast monitor data stream, termed the "ostream" for "observing data stream". Which monitor point values are placed in the ostream, and at what rates, are specifiable on a monitor point by monitor point basis, and this specification is separate and distinct from the specification of that monitor point's archive rate. The ostream will be used to support functionality such as the multicast of round trip phase and total power data.

Last, but not least, alerts are also multicast by the MIB. Alerts are simply alarms. As multicast from a MIB, they are messages that alert any listeners to abnormal conditions within a subsystem that may affect data quality. As such they are of concern to operators and to the flagging process.

The MIB interface also implements a telnet port intended for human interaction with the MIB. The Telnet port presents an interactive shell that accepts all of the set and get commands that are recognized by the service port and provides responses in a format identical to that used by the service port. This port is useful for expert users who need a minimum of interaction.

Control of VLA antennas is accomplished via the CMP (Control and Monitor Processor). While the MIBs present antenna subsystems to the outside world, the CMP, on the other hand, presents a collection of VLA virtual antennas. The CMP interface is a full function port of the MIB interface, adapted to the fact that it is a virtual antenna rather than a subsystem that is being presented. Each virtual VLA antenna accepts UDP based commands via a service port that is identical to the MIB service port, and multicasts archive and ostream monitor data and alerts via a data port identical to the MIB data port. Again, as with MIB-controlled subsystems, each VLA virtual antenna will accept Telnet connections that allow commands to be submitted and the state of devices within the antenna to be queried. The CMP software handles the conversion of VLA to/from EVLA formats for monitor data, alerts, and commands, and hides many of the details of VLA hardware from EVLA software.

The multicast of EVLA format monitor data for VLA antennas by the CMP has been functioning for many months. Its ability to correctly organize, format, and time the delivery of commands to VLA antennas has already undergone initial testing , and is scheduled for intensive testing starting sometime in October-November 2006.

The diagram given in Figure 3 shows a component labeled "Device Browser" (lower left hand side). The device browser is a low level user interface that provides a GUI-based interaction with any device implementing the MIB interface. Its capabilities include the display of a tree of all devices with all monitor points and command points for a single or multiple MIBs, the display of all attributes of a selected monitor point or command point, the ability to set the values of monitor and command point attributes (which means that it can be used to command antenna subsystems), and plotting, in real-time, of multiple, user selected monitor point values.

1.5.1.4 Data Quality, Data Production

Continuing around Figure 3 (page 10) in a roughly counterclockwise direction, one encounters components labeled IDCAF, and ITelcal.

In the Transition System as it will exist at the time the VLA Control System is replaced, flagging will be done in IDCAF. However, as of November 2006, IDCAF for the Transition System is still under development. The current EVLA Monitor and Control System interoperates with the VLA Control System, and depends upon the VLA Control System for the formation and writing of archive records. In this context, the Transition System forms flags based on alerts coming from EVLA MIBs. The flagging information is sent to the Modcomp-based VLA Control System and added to the archive records as formed and written by the VLA Control System. This form of flagging addresses only EVLA antennas and depends upon the VLA Control System. It is now running regularly as a component within the Transition System.

IDCAF (for Interim Data Capture and Format) is the process in the Transition System that will be responsible for forming and writing the archive records. IDCAF will collect the meta-data needed for an archive record, accept visibility data from the VLA correlator, and create and write records in the VLA archive format. IDCAF will also perform the flagging function. It will develop flagging information based on alerts from EVLA and VLA antennas (& other relevant system events) and will not depend upon the VLA Control System. Flagging will be done entirely within the EVLA Monitor and Control System, and will become a part of the archive records written by the EVLA Monitor and Control System.

IDCAF is not intended for use with the WIDAR correlator, or even for use with the Prototype Correlator. The current schedule calls for testing of the IDCAF component to begin in January or February 2007.

ITelcal (Interim Telescope Calibration) provides the same functionality, plus additional capabilities, for the EVLA M&C Transition System as does Antsol (Antenna solutions) for the VLA Control System. Eventually, ITelcal will provide raw self-cal (complex antenna gains) reference pointing offsets, focus settings, and delay settings to the Transition System. Currently it does not yet provide the focus or delay settings. ITelcal has been supplying the complex antenna gains and reference pointing offsets to the Transition System since February 2006. It multicasts this information, as XML messages. Any client software interested in receiving this data need only subscribe to the appropriate multicast group.

1.5.1.5 The Prototype Correlator

As of mid-September 2006, the Prototype Correlator (PTC) (shown in the upper right hand portion of Figure 3, page 10) is scheduled to begin on-the-sky (OTS) testing at the VLA site sometime in January-February 2008. The current plan is to initially use standalone software for configuration of the PTC and to deliver the needed models to the PTC from the Interim Observation Executor via prototype MCCC software.

The transmission of models is only one of the responsibilities of the MCCC software. One of its chief responsibilities is the translation of high level correlator configurations into the allocation of correlator hardware resources (configuration mapping) and the communication of these resource requests to the correlator hardware. As soon as sufficiently capable prototype MCCC software is delivered, the Interim Observation Executor will be modified to configure the PTC using the MCCC software.

Just as EVLA antenna subsystems are configured, monitored and controlled at the hardware level via module interface boards (MIBs), so too are correlator boards configured, monitored, and controlled, at the lowest levels, via software running on correlator module interface boards (CMIBs). The CMIBs plug directly into each correlator board and provide an environment and resources for the execution of device drivers and for a set of communication and configuration software modules residing immediately above the device drivers that are known as module access handlers (MAHs). The MAHs communicate with the MCCC software and with a number of correlator board specific GUI screens that can be used for manual configuration, monitor, and control of individual correlator boards. These board level GUIs provide a means to test the prototype hardware, and will serve as troubleshooting tools for the final WIDAR correlator.

The board specific GUIs will, in turn, communicate with system level GUI screens. The system level screens will be the standalone software that is used to configure the PTC for OTS tests, and they will also serve as problem diagnosis and troubleshooting screens for the final WIDAR correlator.

Support of the PTC will also require that prototype Correlator Backend (CBE), Fast Formatter (FF), and archiving capabilities be available. The CBE is responsible for long term integration and the Fourier transform of the correlator lags. The FF stitches subbands and produces the final binary data stream. Both the visibility data coming from the FF and ancillary data products coming from the correlator station boards and the correlator baseline board must be saved in a manner that allows for easy retrieval. The visibility data will be written as some form of an archive record. The formation of archive records implies meta-data in addition to visibility data, which raises the issue of a data capture and format (DCAF) software component. It is likely that an IDCAF2 will be written to support the PTC. PTC visibility data will not be written in VLA archive format. Currently, FITS or some early version of the Science Data Model (SDM) are likely formats for the archive records. Post processing capabilities, able to accept either the archive records directly or some transform of the archive records will also be needed.

A list of data products, data formats, and software needed to support the PTC is being compiled.

1.5.1.6 The VLA Correlator

The new VLA correlator controller (upper right hand portion of Figure 3, page 10) is the means by which the EVLA M&C Transition System will control the VLA correlator. The new correlator controller provides an Ethernet based control path for the receipt of commands and configuration information. It also replaces the current array processor with a newer array processor. The new correlator controller became operational, under control of the Modcomp-based VLA control System, on September 26, 2006. An initial, successful test of the ability of the EVLA Interim Observation Executor to control the VLA correlator was completed mid-November 2006. Continuum and correlator mode 1A were shown to work. Testing EVLA control of the VLA correlator proved quite difficult with the Modcomp computers still present. Further testing will be deferred until it is possible to operate the monitor and control system, even if in only a rudimentary fashion, with the Modcomps offline. The Visibility Pipe (section 1.4.4.2, page 7 and next paragraph) and IDCAF (section 1.4.7, page 8 and section 1.5.1.4, page12) are needed for this capability. It is anticipated that they will have reached the needed state of readiness sometime in February-March 2007. At that point, further development and testing of EVLA control of the VLA correlator will resume.

An additional bit of hardware, termed the Visibility Pipe (not shown in Figure 3), will be used to make visibility data available to the Transition System. The Visibility Pipe is simply a PC104 module that includes an Ethernet interface. The module's ISA bus will be interfaced to the output of the new array processor, and the visibility data obtained via this bus will then be multicast over the module's Ethernet interface. The Visibility Pipe is basically a hardware bridge, running only the simplest software. It will not reformat or process the visibility data in any way. As of mid-September 2006, the design and layout of the board needed to interface the ISA bus to the array processor output is complete, and wire-wrap of the board is ready to begin. Testing of the Visibility Pipe is expected to begin sometime early in December 2006.

1.5.1.7 The Alert Server

The alert server (center of Figure 3, page 10) is a central collection and processing point for all alerts multicast within the system. The processing performed does not involve interpretation of the alerts, but rather expansion of the rather terse alerts produced by MIBs into more informative messages. The alert server also provides a locus for filtering of the alerts – clients can request all alerts or various subsets of alerts with the filtering needed to produce the subsets done on the server side, providing a single place to implement the filters and a uniform result for all clients. The alert server also provides a single place at which alert off messages can be matched with alert on messages, thereby retiring the alert for all clients throughout the system. As shown in Figure 3, the Array Operator Screen is a client of the alert server.

1.5.1.8 User Interfaces

For the purposes of this section of the document, the user interfaces of interest are the screens released as EVLA Operator Software. These screens are standalone Java applications. They are distributed via Java Web Start, and can be downloaded by going to the URL

http://www.aoc.nrao.edu/asg-internal/jnlp

and clicking on the desired release. This web page is accessible only if one is working within an NRAO domain or via the NRAO VPN.

The list of screens currently available as of the 08Sep2006 release of the EVLA Operator Software is:

1. the Array Operator Screen, which is the main screen used for monitor and control of the array
2. the Device Browser, already briefly described in 1.5.1.3 on Antenna Control
3. the Critical Functions screen, which will eventually (when hardware installation is complete) allow an E-Stop, ACU reset, power reset, stow, park, or standby command to be sent to any antenna, selected group of antennas or to all antennas in the array
4. Ancillary screens – the Alerts, Antennas, Scripts, and Weather screenlets that are a part of the Array Operator Screen, displayed as standalone screenlets
5. Module screens for the ACU (antenna control unit), FRM (focus & rotation), F320 (front end monitor and control), L301 (12-20 GHz synthesizer), M302 and M303 - utility modules that provide a variety of monitor and control points including cryo pressures, the antenna anemometers, vertex room temperatures, the state of critical functions, fire alarms, and a number of other items
6. Telcal screens for the display of calibrator amplitude and phase, calibrator complex gains, pointing records, and reference pointing offsets

In addition to these screens, there are a set of configuration and monitoring screens for the new correlator controller that are based on a package written in C (the rscreens package, borrowed from the VLBA), a user interface to the parameters database, and any number of other screens such as Labview screens developed by engineers for specific hardware development and debugging tasks. With the exception of the ACU screen, EVLA-hosted screens for the monitor and control of VLA antennas, and data quality of the VLA correlator have not yet been developed (as of the 08Sep2006 release of the EVLA Operator Software).

1.5.1.9 Monitor Data Archive

While the actual monitor data archive is clearly the database, a functional view of the archive would include three components – the database filler, the database itself, and the software used to retrieve data from the archive.

Monarch (monitor data archiver) is the filler program. It subscribes to the monitor data archive multicast group, receiving all data intended for the archive. The monitor data, as multicast, is in an XML format. Monarch repackages the data into a form more suitable for insertion into the database tables, buffers a number of records (approximately 100 at this time), and then makes the appropriate call to insert the records into the database. Monarch makes no attempt to interpret or evaluate the data record content in any way and requires no knowledge of the nature of the data or of the data sources.

A web based interface to Monarch exists. It can be viewed at http://mcmonitor/evla-archive. The monarch interface gives status information on the state of the filler program's reception of monitor data archive multicasts and the connection between the filler program and the database. It also presents a (sorted) list of subsystems sending data to the monitor data archive, providing information on the subsystem host name (the sender), traffic volume, IP address, and date and time of first and last data packets sent since monarch was started.

The actual monitor data archive is an Oracle database. There have been and still are concerns that the ingest rate and data volume for the monitor database may become too large - if not too large for Oracle to handle, then too large for effective management within the constraint of the resources available to the project, &/or for timely search and data retrieval. Actually, it is the latter factors that are of concern. Oracle is designed for very large database systems, much larger than the monitor data archive, and it would be very surprising to find it not up to the task.

From time to time, spot checks on the amount of data being written to the monitor data archive have been made. In December 2005, for the case of a "few" (3 to 4) EVLA antennas plus all VLA antennas (monitor data obtained via the CMP), the monitor data archive was writing 9 million rows or approximately 350 Mbytes/day. It will be necessary to trim the data rate as more and more EVLA antennas come online since one EVLA antenna will have an estimated 7X the number of monitor points as a VLA antenna.

An initial version of the monitor data archive, with a basic data retrieval interface, was first brought online in December 2003. The driving factor in the early deployment of a monitor data archive was the need to have a monitor data archiving and retrieval capability to support hardware testing and development for the first EVLA test antenna.

The current monitor data retrieval interface is web based, and can be viewed at:

http://archive.nrao.edu/monarchive/evlamonitor.html

It allows for the retrieval of the values of one monitor point at a time, over a user specifiable time range. To submit a query, one must know the slot ID, the device name, and the monitor point name. For a completely naïve user, obtaining these quantities is a three step process. Without going into detail at this point in the document, the slot ID is the host name or host IP address of the subsystem sending monitor data to the archive. Its value is typically something of the form DCSxx (a virtual

VLA antenna presented by the CMP) or ea13-l301-1 (EVLA antenna 13, the 1st of two L301 12-20 GHz synthesizers). This value can be obtained most easily from the column of the monarch interface display labeled "Sender". Once the correct value of the slot ID has been obtained, an archive contents query can be performed to obtain first, the names of all devices associated with that slot ID, and second, the names of all monitor points associated with a device. Only after all of these values have been determined can one proceed with a search of the archive for the desired monitor data values. (The monitor data retrieval interface has a direct link to the web page used for archive contents queries.) Of course, experienced users already know these values for the monitor points of interest to them, and can proceed directly to a search of the monitor data archive.

For naïve users, the current monitor data retrieval interface is cumbersome and time consuming to use, and the limitation of retrieving only one monitor point at a time is irksome for all users. An alternative interface that allows the specification of multiple monitor points using a clickable tree of slot ID, device, and monitor point names is under development. The possibility of reorganizing monitor data retrieval (and the underlying database) for retrieval organized by device is also being examined.

1.5.2 The Final System

Figure 4, given below, is a diagram of some of the high level dataflows for the Final version of the EVLA Monitor and Control System. It is quite similar in many respects to Figure 3 (page 10), the diagram of the Transition System dataflows; however there are some notable differences.

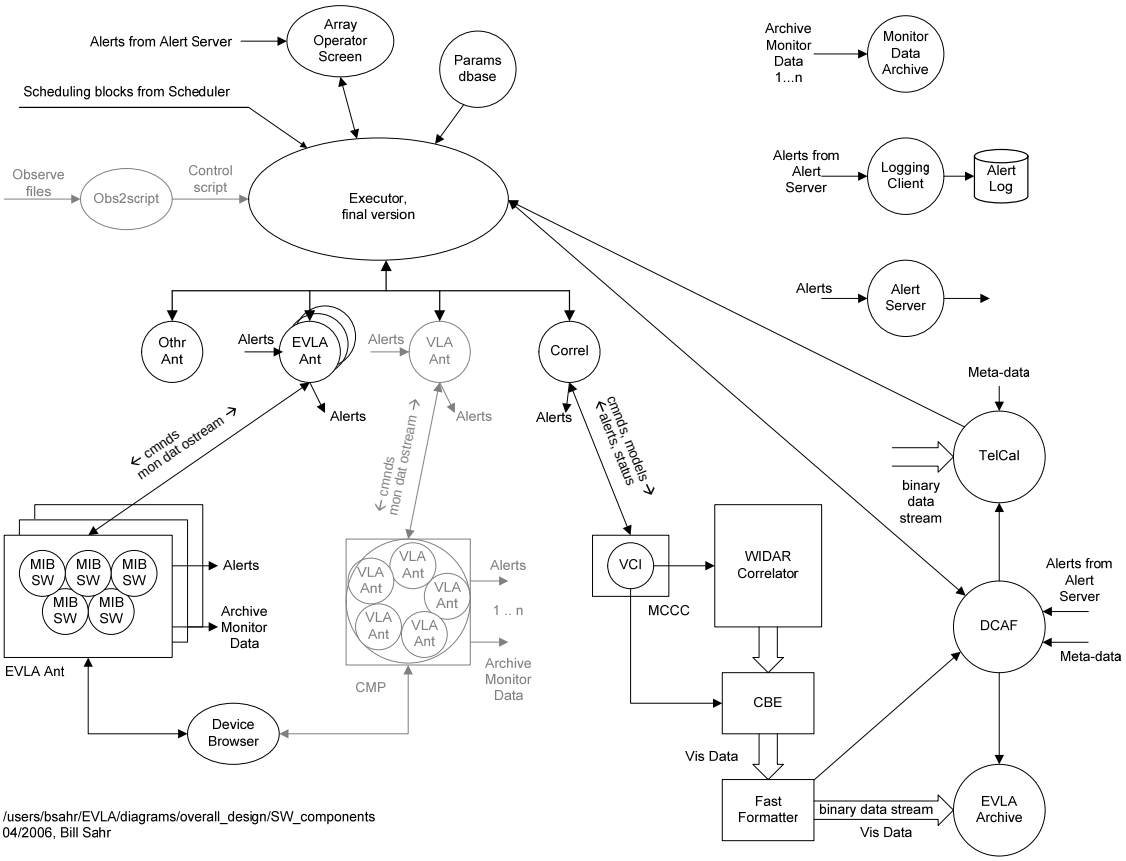


Figure 4 Final System Dataflows

1.5.2.1 Device Objects

Perhaps the most prominent difference between the dataflows for the Transition System and the Final System is the introduction of a layer of software entities between the Observation Executor and the hardware it configures and controls. Where in Figure 3, Transition System DataFlows, the Interim Observation Executor communicates directly with EVLA antenna MIBs and the VLA virtual antennas presented by the CMP, in Figure 4, Final System Dataflows, the final version of the Observation Executor is shown communicating with intermediate entities that, in turn, communicate directly with the hardware (or, for VLA antennas, the hardware surrogate). The intermediate entities are referred to as device objects, and it is expected that there will be one type of object for each major device type in the system – specifically an EVLA antenna object, a (WIDAR) correlator object, and, during the evolution from Transition System to Final System, there may well be a VLA antenna object (shown in gray rather than black). If Phase II of the EVLA is ever implemented, this architecture would be expanded to include a VLBA antenna object, and a New Mexico Array antenna object.

These device objects play several roles in the functional architecture and design of the system. First, they become the locus for knowledge about the hardware – its characteristics, peculiarities, timing, etc. For example, it might be possible to move some or even all of the knowledge contained in the Interim Observation Executor concerning the differences in the LO systems for EVLA versus VLA antennas into the EVLA and VLA antenna objects. Similarly, it might be possible to move some or all knowledge of the timing of waveguide cycles out of the Executor and into the VLA antenna object.

Next, the device objects become the place where the state of the devices is maintained. For the antenna objects, this statement means that both alerts and the real-time monitor data stream are received by the antenna object and assembled into a coherent picture of the state of the antenna.

That the device objects know the actual state of the devices makes them a natural location for the detection of faults, fault analysis, and the generation of higher level, more intelligent and informative alerts based on the fault analysis.

That the device objects also know the commanded state of the devices makes it possible to compare the commanded state with the actual state, leading to yet more useful alerts and the possibility of automating attempts to bring the actual state of a device into conformance with the commanded state.

The usefulness of the device objects does not end with the functions already described. In the Transition System, user interface complexity is substantially increased by the need to go to many different data sources. For example, to know what there is to know about a single EVLA antenna, a user interface must go separately to each EVLA antenna subsystem MIB. Given that the estimate for the number of MIBs in a final, production version of an EVLA antenna is 34 to 35, in the final system, if there are no EVLA antenna objects, a user interface that wants to present comprehensive data on a single EVLA antenna will be required to collect data from all 34 to 35 different data sources. If an EVLA antenna object is present in the final system, and if it does maintain a coherent, current description of antenna state, then that same user interface would need to collect data from only one data source. Further, it would be possible to implement server filters within the device objects allowing a user interface or any other software component to request only the data of interest, for example, all ACU/FRM data or only data relevant to the LO system.

A tremendous amount of vital and detailed knowledge is being accumulated in the Interim Observation Executor. This knowledge must be retained and used in the Final System. One possible approach is to implement the device objects and the Final Observation Executor by partitioning the functionality of the Interim Observation Executor. Higher level knowledge concerning the conduct of an observation and the scans and subscans of which an observing run is composed reside in the Observation Executor. Knowledge of hardware details, timing, and device state is relegated to the device objects.

1.5.2.2 The WIDAR Correlator

The components of the Final System dataflow diagram (page17) concerned with the WIDAR correlator are the:

1. correlator device server
2. Virtual Correlator Interface (VCI)
3. WIDAR correlator
4. Correlator Backend (CBE)
5. Fast Formatter (FF)

Conceptually, the relationship among these parts is as follows. As with the antenna servers, the final version of the Observation Executor communicates correlator configuration information to the correlator server. The correlator server passes this information, probably with additional transformation, to the VCI (and associated software). The configuration information received by the VCI is then either mapped onto correlator resources and parceled out to the CMIBs (correlator module interface boards) located on each of the 300+ correlator boards, or it is forwarded to the CBE. The correlator boards and CBE report status back to the VCI, which, in turn forwards this information back to the EVLA Monitor and Control System.

The models and model updates needed by the correlator will originate in either the Observation Executor or in the correlator server – the issue is undecided, but the correlator server is certainly a candidate location.

The VCI software will be hosted by the Master Correlator Control Computer (MCCC), a 1+1 redundant system located in the correlator room.

The baseline boards in the WIDAR correlator send lag frames to the CBE. The lag frames will be sent as UDP packets over a dedicated, switched Ethernet network. The CBE performs long term integration of the lag frames (if needed) and the FFT of the raw lag data. The baseline boards are, basically, subband correlators, so the CBE nodes (the CBE will be a cluster) are operating on subband data. It is the job of the Fast Formatter to combine the subband data into a single binary data stream, and to format that binary data stream. It is hoped, and discussions are ongoing, that ALMA and the EVLA will use the same binary data format (BDF). The resulting binary data stream will be written to the EVLA archive and pointers to the binary data will be made available to the EVLA data capture and format (EVLA DCAF) software component for inclusion in the archive records.

1.5.2.3 EVLA DCAF and EVLA Telcal

As with IDCAF, EVLA DCAF will be responsible for the formatting and writing of EVLA archive records. EVLA archive records will be written in a format that follows the Science Data Model (SDM). The SDM will be an adaptation of the ALMA Science Data Model (ASDM). It will be a format jointly agreed upon and used by both ALMA and the EVLA.

EVLA DCAF will write the SDM formatted archive records to an EVLA archive that uses the same underlying technology as the ALMA archive. It is likely that the data will not go directly to the archive, but rather to a staging area first and thence to the archive.

As shown in Figure 4 (page 17), EVLA DCAF will accept not only meta-data (from a variety of sources), and pointers to the visibility data, but also alerts from the alert server. The implication of this fact is that, as with IDCAF, flagging will be done within EVLA DCAF.

EVLA Telcal's core role in the Final System w.r.t to the monitor and control system, is the same as the role of ITelcal in the Transition System. It will provide raw self-cal (complex antenna gains) data, reference pointing offsets, focus settings, and delay settings to the Final version of the EVLA Monitor and Control System. The requirements for EVLA Telcal also assign to it responsibility for autophasing and the determination of the phase stability of the instrument. These and other requirements for the EVLA Telcal software component can be found in the document entitled "EVLA e2e Science Software Requirements", Bryan Butler, 04/15/2003. It is document # 26 of the Computing Working Documents web page:

http://www.aoc.nrao.edu/evla/techdocs/computer/workdocs/index.shtml

Please note that in that document, EVLA Telcal is referred to as Real-time Calibrator Analysis. The requirements can be found on pages 36-38.

EVLA Telcal will require access to the visibility data. Given that it has real-time responsibilities there is every reason to believe that acquiring the visibility data from the archive will be too slow. Two possibilities present themselves. First, EVLA Telcal may acquire a copy of the visibility data, perhaps via a socket connection or even via shared memory directly from the Fast Formatter. Or, if the Fast Formatter does write the visibility data (binary data stream) to a staging area rather than directly to the archive, EVLA Telcal might tap into the data stream at that point.

1.5.2.4 The Alerts Subsystem

In the final system, the Alerts subsystem is an elaboration and further development of the Alert Server as found in the Transition System. Figure 5, (given below) is a diagram of the Alerts Subsystem as conceived for the Final version of the EVLA Monitor and Control System.

Figure 5 presents a hierarchical tree for the data flow for alerts in the EVLA Monitor and Control System. The smallest, terminal leaves on the tree (outer ring, upper portion, labeled MIB_n, VLA_n, & $CMIB_n$) are alert sources. The next level in the tree (Antenna Server, CMP, AD Server, VCI/MCCC) are intermediate servers, and will actually be the Device Servers discussed in section 1.5.2.1 (page 18) and shown in Figure 4 (page 17). The intermediate servers will consult an Alerts Description Database to expand the information contained in the low level alerts coming from the MIBs, VLA antenna subsystems, and the CMIBs. The intermediate servers will also be the site of

fault analysis, and will likely suppress some low level alerts while generating additional alerts of their own.

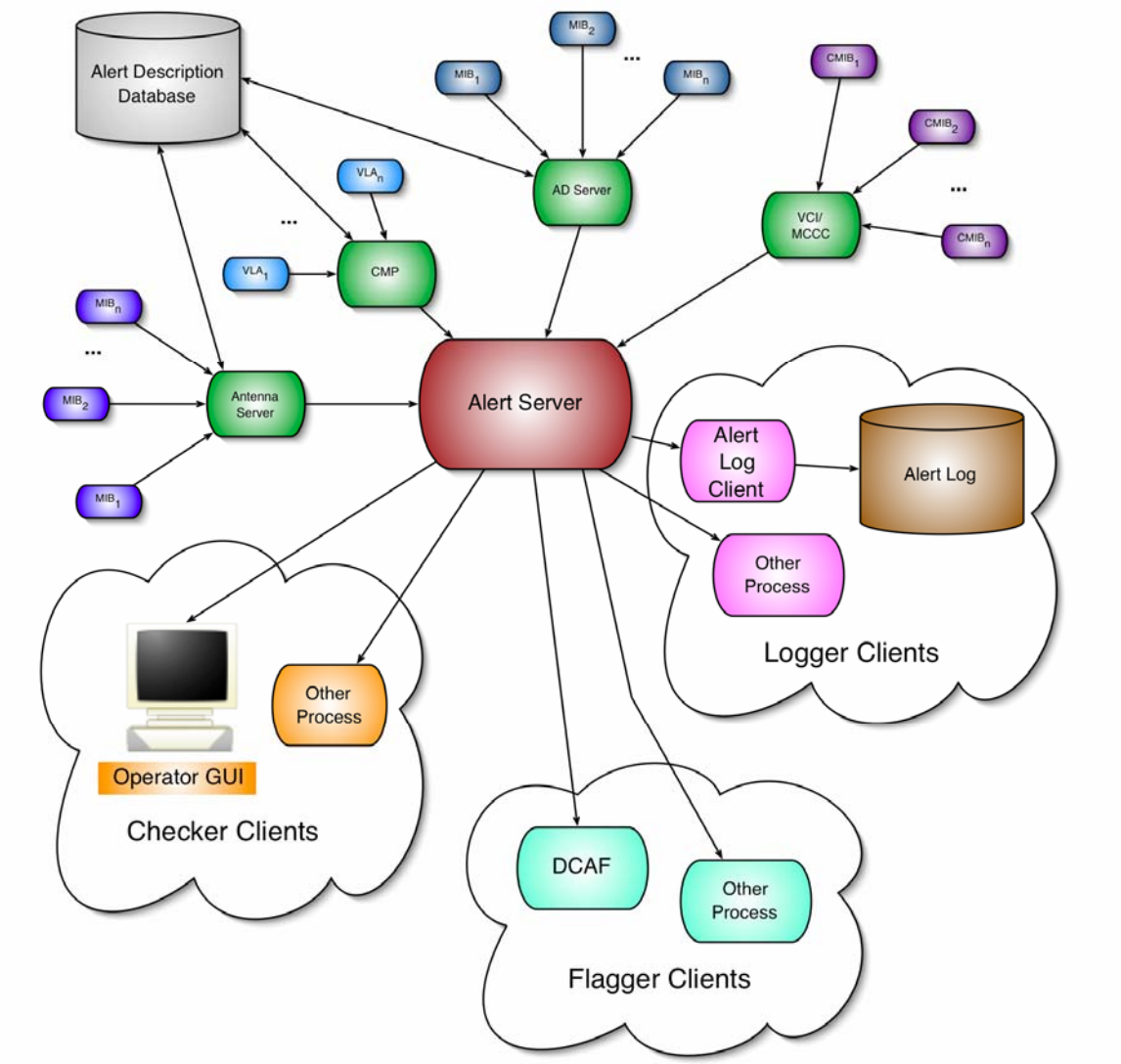


Figure 5 Alerts Subsystem

The fully formed alerts created in the intermediate servers will be forwarded to the Alert Server. The Alert server is the central collection point for alerts. Clients wanting alerts will go the Alert Server, and the Alert Server will implement server side filtering to make it easy for a client to request various subsets of the set of all alerts. While there may be many clients for alerts in the system, some of them quite specialized, it is certain that, at a minimum, there will be a logging client that will create a persistent log of all alerts, that DCAF will need the alerts for flagging, and that the Array Operator Screen, the central operator's GUI, will require the alerts.

1.6 *Carryover of Components from the Transition to the Final System*

A substantial amount of software must be written to implement the Transition version of the EVLA Monitor and Control System. Basically, it is a complete rewrite and expansion of the present VLA Control System – duplicating virtually all of its functionality and extending it to include new hardware. As such, it is a major undertaking, involving many person-years of effort. This section

of the design document discusses how much of that effort, of what type, and at what levels will carryover into the Final version of the EVLA Monitor and Control System.

1.6.1 Observation Executor

It is expected that there will be substantial transfer of work and effort from the Interim Observation Executor to the Final Observation Executor. Design reuse, the transfer of a great deal of accumulated knowledge concerning hardware behavior and timing, and considerable code reuse should all be possible. The current plan is to partition the Interim Observation Executor in such a way as to capture its higher level knowledge of the conduct of observations into a framework upon which the final version of the Observation Executor will be built, and its knowledge of hardware and timing in device server frameworks that will serve as foundations for the further development of the EVLA antenna server, the VLA antenna server, and the correlator server. Strictly speaking, there will be no VLA antenna server in the Final system, but it is expected that during the transition to the Final System, such an entity will exist.

1.6.2 Obs2script

The function of obs2cript is to convert VLA-style Observe files to EVLA control scripts. VLA-style Observe files will not be used in the Final version of the EVLA Monitor and Control System. Instead, an Observation Scheduler will pass scheduling blocks to the Observation Executor, and these scheduling blocks will contain control scripts that specify the conduct of an observation. Obs2script will not be needed in the final system. Over time, as VLA-style Observe files fall into disuse, obs2script will no longer be needed.

The Observation Scheduler is shown in Figure 1 (page 2), of the EVLA Top Level Data Flow diagram. The Observation Scheduler is not considered to be a part of the EVLA Monitor and Control System. However, it will interface directly to the Observation Executor.

1.6.3 Antenna Monitor & Control (AMCS)

1.6.3.1 Monitor & Control of EVLA Antennas

It is expected that there will be a 100% transfer of the MIB (module interface board) software from the Transition System to the Final system. The MIB interface is one of the foundations upon which the Final System will be built. Further, the Transition System in no way inhibits or constrains the MIB software from developing into its full, final, mature form.

1.6.3.2 Monitor & Control of VLA Antennas

There will be no VLA antennas in the Final version of the EVLA Monitor and Control System. The CMP (sections 1.4.3.2 and 1.5.1.3) hardware and software will have no role in the Final System.

1.6.4 Correlator Monitor & Control (CMCS)

1.6.4.1 Monitor & Control of the VLA Correlator

One of the defining characteristics of the Final version of the EVLA M&C System is the presence of the WIDAR correlator and the absence of the VLA correlator. Neither the hardware nor the software comprising either the new correlator controller or the visibility pipe will be used in the Final System. The code in the Interim Observation Executor developed for control of the VLA correlator will also not be used. It is possible that some portion of the architecture and design of the

software infrastructure used in the Interim Observation Executor for correlator control may survive in the correlator device server that will be a component of the Final system.

1.6.4.2 Monitor & Control of the Prototype Correlator

There will be little or no throwaway effort here. After initial on-the-sky (OTS) testing of the Prototype Correlator (PTC) is complete, and as soon as even a very primitive prototype MCCC software is available, monitor and control of the PTC becomes a testbed for monitor and control of the full production version of the WIDAR Correlator.

PTC-related software developed for the Interim Observation Executor, for the Correlator Backend (CBE), and the Fast Formatter (FF) will all have direct relevance for the Final version of the EVLA M&C System. Among other things, the data output of the PTC Baseline Board will provide the first opportunity to implement the Binary Data Format (BDF) for the output of the Fast Formatter, and it will also provide the basis for the first implantation of Science Data Model (SDM) formatted archive records.

Further, the PTC will also provide a valuable testbed for field testing the correlator board device drivers and module access handlers (MAHs), correlator board-level GUIs, correlator system-level GUIs, and correlator related utility software.

1.6.4.3 Interim Telcal (ITelcal)

ITelcal is the name given to Telcal as written for the Transition System. ITelcal will produce a set of results that are identical in nature (antenna gains, pointing offsets, focus & delay determinations) to the results needed from the real-time, core component of EVLA Telcal. ITelcal and EVLA Telcal will likely use the same method to make these results available – multicast. Unfortunately, the resemblance between ITelcal and EVLA Telcal will probably end at the level of these fairly shallow generalizations. To calculate the antenna gains, ITelcal uses the VLA Antsol, with a C language wrapper. It has an expectation that visibility data will conform to the VLA archive format - the largest input record it sees is on order 1.5Mbytes, and, worst case, it sees that record at most once every 6 2/3 seconds. These characteristics are not a good fit to EVLA Telcal. Antsol will probably not make a suitable skeleton for EVLA Telcal. EVLA Telcal will see visibility data in the Binary Data Format (BDF), the record size will be much larger, and the data rate will be much higher. Additionally, the EVLA version of Telcal will be called upon to satisfy a number of requirements for functionality that do not apply to ITelcal. (See EVLA e2e Science Software Requirements, Bryan Butler, 04/15/2003, document #26, on Computing Working Documents, http://www.aoc.nrao.edu/evla/techdocs/computer/workdocs/index.shtml, section 5.1, Real-time Calibrator Analysis, pps 36-38.)

1.6.4.4 Interim Data Capture and Format (IDCAF) & Flagging

As with ITelcal and EVLA Telcal, so it goes with IDCAF and EVLA DCAF. IDCAF will accept data from the VLA correlator and create and write VLA Archive format records. It will deal with VLA correlator data record sizes and data rates. EVLA DCAF will accept data from the WIDAR correlator. It will create and write records that conform to the Science Data Model (SDM), and those records will not contain the actual visibility data but, rather, pointers to the visibility data, which will use the Binary Data Format (BDF). Both software components will perform flagging, but that fact is about the only similarity between them. There will be little or no carryover from IDCAF to EVLA DCAF.

As mentioned earlier (section 1.5.1.5, The Prototype Correlator, page 13) an IDCAF2 will likely be written to support the Prototype Correlator. It is likely that there will be carryover from this software component to EVLA DCAF.

1.6.4.5 User Interfaces

The User Interfaces considered here are those listed in section 1.2, Components of the EVLA Monitor and Control System. For the most part, what will and will not transfer from the Transition System to the Final System is fairly obvious. The Array Operator Screen, EVLA Antenna Screens, and the Device Browser will transfer 100% from the Transition System to the Final System.

VLA Antenna Screens and VLA Correlator Screens (configuration and status) will not be needed in the Final System. Data Quality Screens developed for the VLA Correlator and ITelcal will be of no use in the Final System.

1.6.4.6 Parameters Database

The parameters database, purged of data relating to VLA antennas, will carryover 100% into the Final System.

1.6.4.7 Monitor Data Archive

All aspects of the monitor data archive – the filler program, the actual database, and the data retrieval interface - should carryover fully from the Transition System to the Final System.

1.6.4.8 The Alerts Subsystem

The Alerts Subsystem in the Transition system is a simplified precursor of the Alerts Subsystem as it will exist in the Final System. The current Alert Server does, in fact, contain a memory resident, simplified version of an Alert Description Database (see Figure 5 Alerts Subsystem, page 21), it does implement server side filters, it does implement an HTTP interface for client queries, and it does accept queries from its single client as URIs. All of these characteristics are similar or identical to characteristics of the Alert Server in the Final System. In short, the carryover to the final system will include the basic architecture and design, lessons learned, and very likely some actual code, plus at least one client – the alerts screenlet that is part of the Array Operator Screen.

1.6.4.9 Operations Software

Operations software is difficult to discuss succinctly as it can cover a very wide array of software and scripts, some of it not well documented or formally maintained. Generally, in the context of this document, the term “Operations Software” will be used to describe software that is used for a) antenna acceptance, and b) antenna checkout. By antenna acceptance is meant that software that is used to certify that an antenna coming out of the antenna barn after conversion from a VLA antenna to an EVLA antenna, or after general maintenance and overhaul, is functioning correctly in all of the particulars need to participate in observations. Antenna checkout refers to the suite of tests that are run on startup after maintenance days and after an antenna has been moved to verify its functionality.

Operations software will be discussed in more detail later in this document. Suffice it to say at this point in this document that with respect to the Transition and Final Systems Operations Software falls into two broad categories – operations software that will be needed when the Modcomp-based VLA Control System is turned off, and software that will be needed when the VLA correlator is retired. Operations Software developed to support operations in the absence of the VLA Control

System should transfer fully from the Transition System to the Final System. Operations Software designed to operate with the WIDAR Correlator is, by definition, Final System software.

To date, probably the most extensive, easily accessible lists of Operations Software categorized along these lines is to be found in the "EVLA Monitor and Control Near-Term Software Development Plan", Bill Sahr, 04/15/2005, which is document #41 on the Computing Working Documents web page –

http://www.aoc.nrao.edu/evla/techdocs/computer/workdocs/index.shtml

Please see pages 18-19, 22-23, and 25-26.

1.6.4.10 Carryover of Components, Summary Table

Transition System	**Final System**
Interim Observation Executor ----------→	Final Observation Executor
Obs2script	not needed
AMCS	AMCS
VLA Antennas	not needed
EVLA Antennas ----------→	EVLA Antennas
CMCS	CMCS
VLA Correlator	not needed
WIDAR Prototype Correl ----------→	WIDAR Correlator
Interim Telcal	EVLA Telcal
IDCAF & Flagging	EVLA DCAF & Flagging
User Interfaces	User Interfaces
Array Operator Screen ----------→	Array Operator Screen
EVLA Antenna Screens ----------→	EVLA Antenna Screens
VLA Antenna Screens	not needed
VLA Correlator Screens	not needed
Prototype Correlator Screens ----------→	WIDAR Correlator Screens
Data Quality, VLA Correl	Data Quality, WIDAR Correl
Data Quality, ITelcal	Data Quality, EVLA Telcal
Device Browser ----------→	Device Browser
Parameters Database ----------→	Parameters Database
Monitor Data Archive ----------→	Monitor Data Archive
Alerts Subsystem ----------→	Alerts Subsystem
Operations Software	Operations Software
All of the Operations Software developed to support the Transition System in the absence of the Modcomp-based VLA Control System will be used in the Final Version of the EVLA Monitor and Control System.	

Operations software developed for use in the absence of the VLA correlator will be unique to the Final System. Indeed, it is, by definition, Final System software.

Legend
Blue font (with arrow from Transition System to Final System) – substantial or full carryover from Transition System to Final System

Red font – unique to the Transition System

Green font – unique to the Final System

2 EVLA M&C System, Discussion of Selected Components

2.1 *Observation Executor, Interim Version*

The Transition System version of the Observation Executor is considered to be an interim version because changes are needed in the structure of the Interim Observation Executor to support the architecture of the final EVLA Monitor and Control System. The final version of the Observation Executor will not be a complete redesign and rewrite of the Interim Observation Executor. Instead, the goal will be to incrementally transform the Interim Observation Executor into the form desired for the final version of the Observation Executor.

Figure 6 is a diagram of the design or structure of the Interim Observation Executor. The diagram indicates that the Executor uses a single process, multithreaded model, with all of the components sharing the same address space. The Executor instantiates Array objects. Array objects represent the first level of subarraying. Array objects are not persistent. They may be created and destroyed as needed. Array Objects, in turn, contain Subarray Objects. An Array object may contain one or more Subarray objects. Subarray objects represent the second level of subarraying.

Each Array object has 28 Antenna Configuration or Antenna data objects that are mapped, as disjoint sets, into the one or more Subarray objects contained in the Array objects. In the diagram, since each Array is shown with only one Subarray all 28 Antenna Configuration objects are allocated to the single Subarray object in each Array.

The Executor also instantiates 28 AntennaPhysical objects. AntennaPhysical objects are persistent and represent the actual antennas in the physical array. Their lifetime is the same as the lifetime of the Executor. AntennaPhysical objects are mapped to Antenna Configuration objects. One way of looking at the association is that every Array has 28 logical antennas (Antenna Configuration objects) some of which are mapped to actual physical antennas (the AntennaPhysical objects) and some of which point to null objects.

The AntennaPhysical objects assigned to an Array object represent the actual antennas that are associated with a particular observation and control script. The control script cannot move antennas among Arrays. A control script can create subarrays within an Array. This capability is what is meant by second level subarraying. It is under the control of the observer.

Correlator configuration is handled at the level of the Subarray objects. Correlator delay models are generated at the level of AntennaPhysical.

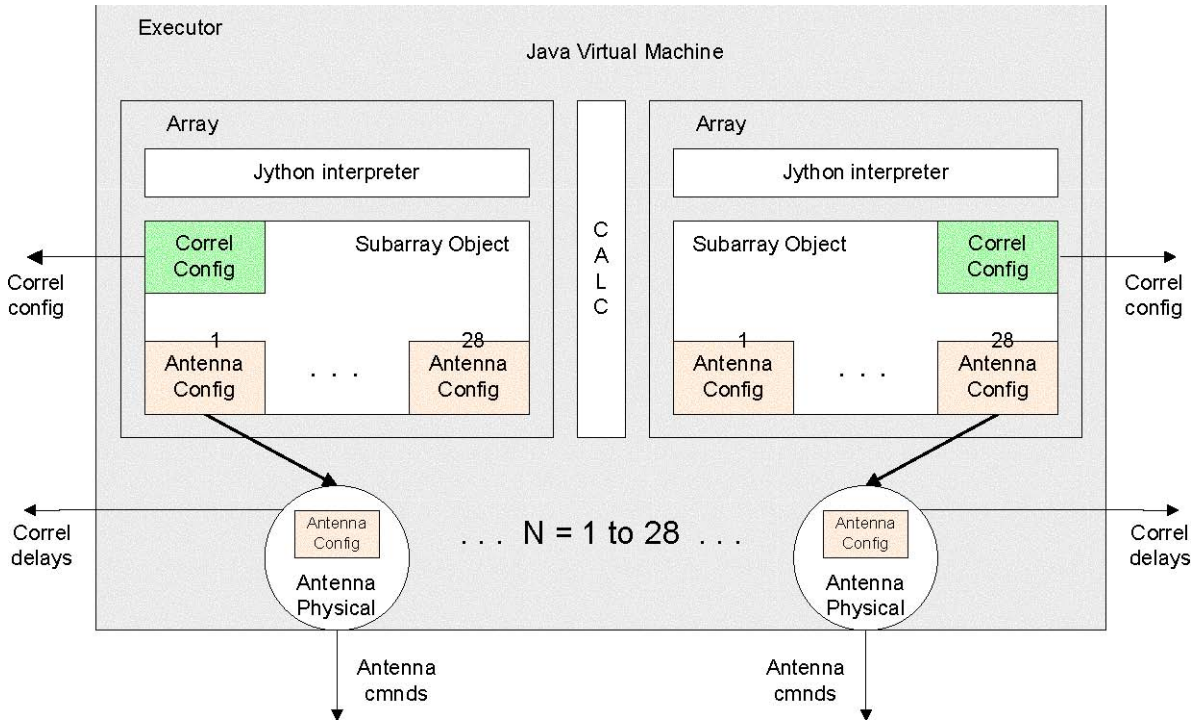


Figure 6 Observation Executor

Antenna Configuration objects are mapped to AntennaPhysical objects or to null pointers if a particular antenna has not been assigned to the Array in question. Cloning an Antenna Configuration object in the appropriate AntennaPhysical object specifies the desired state of an antenna. In the current implementation of the Executor, the AntennaPhysical objects issue commands directly to the EVLA MIBs to configure an antenna for an observation.

The geometric and astrometric aspects of running the array are provided by CALC. This is a system in general use by the geodetic and astrometric VLBI community. It is written in Fortran. Interface routines have been written to make CALC results available as the result of a Fortran subroutine call, a C function call or a java static method. As implemented, CALC supplies delays, pointing angles, and u,v values.

For each new observation, CALC is called three times by the Subarray for each (virtual) antenna to provide initial values for delays and derivatives. After the observation has begun, it is periodically called by each AntennaPhysical to get current values.

The most computationally extreme case for the Executor is on-the-fly-mosaicing (OTFM). Benchmarks show that the current implementation of the Executor may not execute CALC quickly enough to drive the array at the required rate. If this proves to be the case, we must arrange things so that more CPUs are used to execute CALC. This is most logically done by spreading the AntennaPhysical objects among several CPUs, each with their own copy of CALC.

Most of the lines of code of CALC, and much of the computational work, is involved with estimating the position, orientation, and velocity of the earth at the given timestamp. CALC recognizes that successive calls at the same timestamp need not repeat this part of the calculation, and therefore these run much faster than the first, perhaps as much as 90%. So handling 28 antennas requires about three times as much computation as handling one. If we are required to split AntennaPhysicals, each with their own copy of CALC, into different CPUS, we will probably want one CPU per antenna.

2.2 *Observation Executor, Final Version*

The Final Version of the Observation Executor will be a component of the Final Version of the EVLA Monitor and Control System. It is not seen as belonging to the Transition System.

At this point in time (11/2006) the design effort for the final version of the Observation Executor has not yet begun. A formal list of requirements for the final version of the Observation Executor is under development, but has not yet been released. However, it is clear that aspects of the interim Observation Executor that will be re-examined will include 1) the use of jython as the scripting language, 2) the lack of a standalone, network addressable antenna object, and 3) the relationship of CALC to the Observation Executor.

2.2.1 Jython

Jython was chosen as the basis for control scripts because of the intimacy of its binding to Java, the language used for the Observation Executor. Jython scripts can instantiate Java objects, and Java object methods can be invoked directly from Jython. However, there are concerns about support for Jython. In October 2000, the Jython project was moved to SourceForge, which was a positive step, but there is a low level of development activity. The latest production release of Jython is version 2.1, released on 31Dec2001. It supports the Java JDK version 1.2 and python versions 2.0 and 2.1.1. The latest development version is Jython 2.2a1, released on 17Jul2005. The “a” in 2.2a1 represents an alpha release. This release, which supports the Java JDK version 1.5 (Java 5), most of Python 2.2 and some features of Python 2.3, has not yet been promoted to a production release – not a good sign for the level of support.

To say that a particular release of jython supports a particular version of Java means that it will work with the supported version, and may or may not work with later versions. The interim Observation Executor is written using Java 5. Jython 2.1 did not work with Java 5, but a jar file was found (at http://www.interstel.de/jython) that reconciled the incompatibilities.

2.2.2 Antenna Object

The characteristic of the interim Observation Executor that is most relevant to architectural considerations is the lack of a discrete, identifiable, network addressable antenna object that can serve as a locus for the maintenance of antenna state and communications with other software components of the EVLA Monitor and Control System.

As originally conceived, the final version of the EVLA Monitor and Control System will contain standalone antenna objects that can be directly addressed by other software components running in the same or a different system. The functionality assigned to these antenna objects would be:

- To receive commands, presumably higher level commands that are agnostic w.r.t. antenna type, from the Observation Executor, translate the commands into lower level, hardware specific commands, and send these commands to the hardware subsystems.
- To receive a realtime monitor data stream (the ostream) from the antenna subsystems that is used to maintain a coherent representation of antenna state.
- To compare the commanded antenna state with the actual antenna state, and, as necessary, attempt to bring the actual state into conformance with the commanded state.
- To perform fault analysis and generate (higher level, more intelligent) alerts as necessary and appropriate.

The existence of antenna objects with functionality as described above clarifies and simplifies the system architecture. The antenna objects:

- Provide single sources for information on antenna state, removing the need for operator screens and other client software to gather information from many different components of the system. In the transition system client software is already becoming overly complex and in danger of losing its coherency because of the need to gather information from a large number of different sources.
- Are the obvious places to perform a fault tree analysis that will help to identify the root cause of a problem, helping to eliminate error cascades that serve only to confuse the operators.

A diagram follows that attempts to illustrate the antenna object functionality and its role in the architecture of the system.

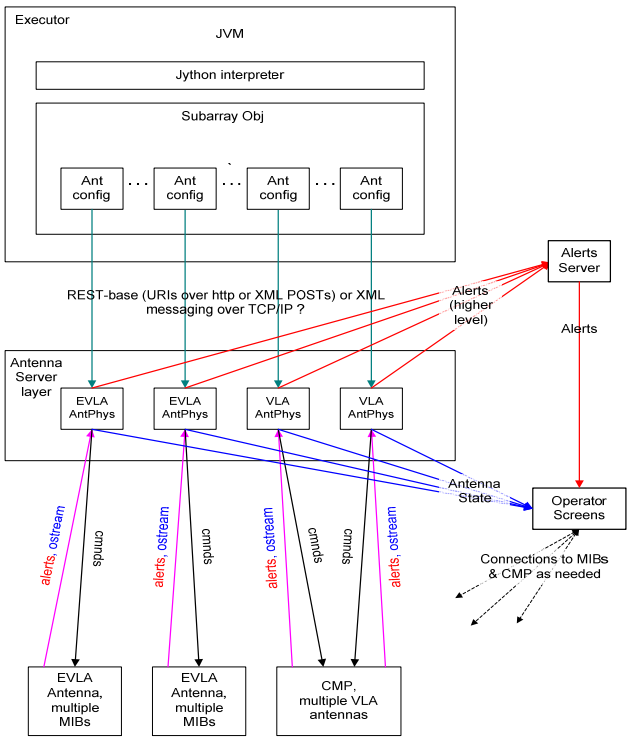


Figure 7 Observation Executor, Final Version

2.3 *Antenna Monitor and Control*

2.3.1 EVLA Antennas: MIB-based Devices

MIB is an acronym for the phrase module interface board. The term MIB refers to the in-house designed single board computer that mates to EVLA antenna module hardware and provides a physical interface (Ethernet) for communications. The term "MIB interface" originally referred to the generic software developed for the MIB, including the MIB Framework software, the MIB Data Port, and the MIB Service Port. Over time the MIB interface was ported to two additional platforms – the CMP and the CMIB-based deformatter boards. The MIB interface has now become the standard EVLA interface for control of antenna based subsystems. It will also become the standard for control of auxiliary devices such as the Atmospheric Phase Interferometer and the central weather station.

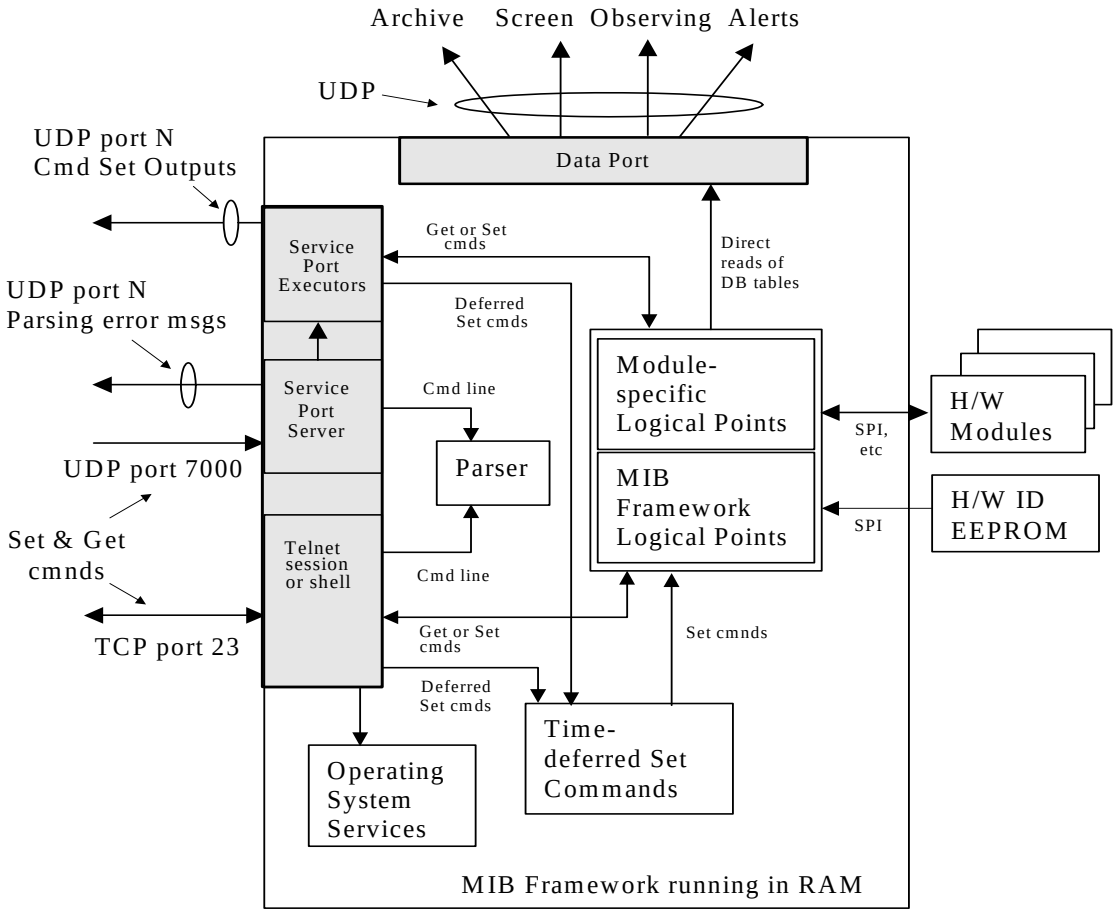


Figure 8 MIB Software Diagram

The MIB hardware and software have already been subjected to a critical design review (CDR). The MIB CDR, known as the "EVLA Monitor and Control Hardware Critical Design Review" was held on October 20, 2004 in Socorro, NM. The web page for this CDR, giving the list of reviewers, the agenda, design documents and drawings, the presentations, and the report of the review panel can be found at: http://www.aoc.nrao.edu/evla/admin/reviews/MIB/MIB.html. Overall, the report of the review panel was positive with respect to both the MIB hardware and the MIB software. One

conclusion of the review panel was that “no flaws in design or implementation serious enough to preclude quantity production of the MIB” were found.

The following documents define the MIB Framework and the interface it presents to the external world. They can be found on the Computing Working Documents web page:

http://www.aoc.nrao.edu/evla/techdocs/computer/workdocs/index.shtml

36	MIB Framework Software, Version 1.1.0	Pete Whiteis	10/08/2004	pdf
35	MIB Data Port ICD, Version 1.2.0	C. Frank Helvey	09/30/2004	pdf
34	MIB Service Port ICD, Version 1.2.0	C. Frank Helvey	09/22/2004	pdf

2.3.2 VLA Antennas: The Control and Monitor Processor (CMP)

The CMP is the gateway by which the EVLA Monitor and Control System will monitor and control VLA antennas. There is a device in the present VLA Control System, known as the Serial Line Controller (SLC), that is a clearinghouse for all VLA antenna monitor data and commands. Eventually, all monitor data from VLA antennas arrives at the SLC, and all commands destined for VLA antennas are first sent to the SLC.

The CMP provides an Ethernet accessible hardware and software interface to the SLC. The hardware layer of the CMP provides a physical interface to the SLC, pulling monitor data from the SLC into a processor accessible to the EVLA Monitor and Control System, and transferring commands generated by the EVLA Monitor and Control System into the SLC. The software layer presents VLA antennas to the EVLA Monitor and Control System as virtual antennas implementing the standard EVLA (MIB) interface, and performs format translation – VLA monitor data into EVLA formatted monitor data and EVLA formatted commands into VLA formatted commands.

Figure 9, given below, shows the position of the CMP in the EVLA Monitor and Control System deployment as it will look shortly after the Modcomp-based VLA Control System is retired.

The material in this section is an excerpted, condensed version of a yet-to-be-published document written by Hichem Ben Frej on the design of the Control and Monitor Processor (CMP).
The term “DCS” as used in this excerpt is a reference to the manner in which VLA antennas are addressed. While a DCS # actually refers to a particular antenna pad, for the purposes of this document, to communicate with a VLA DCS number is to communicate with a VLA antenna.

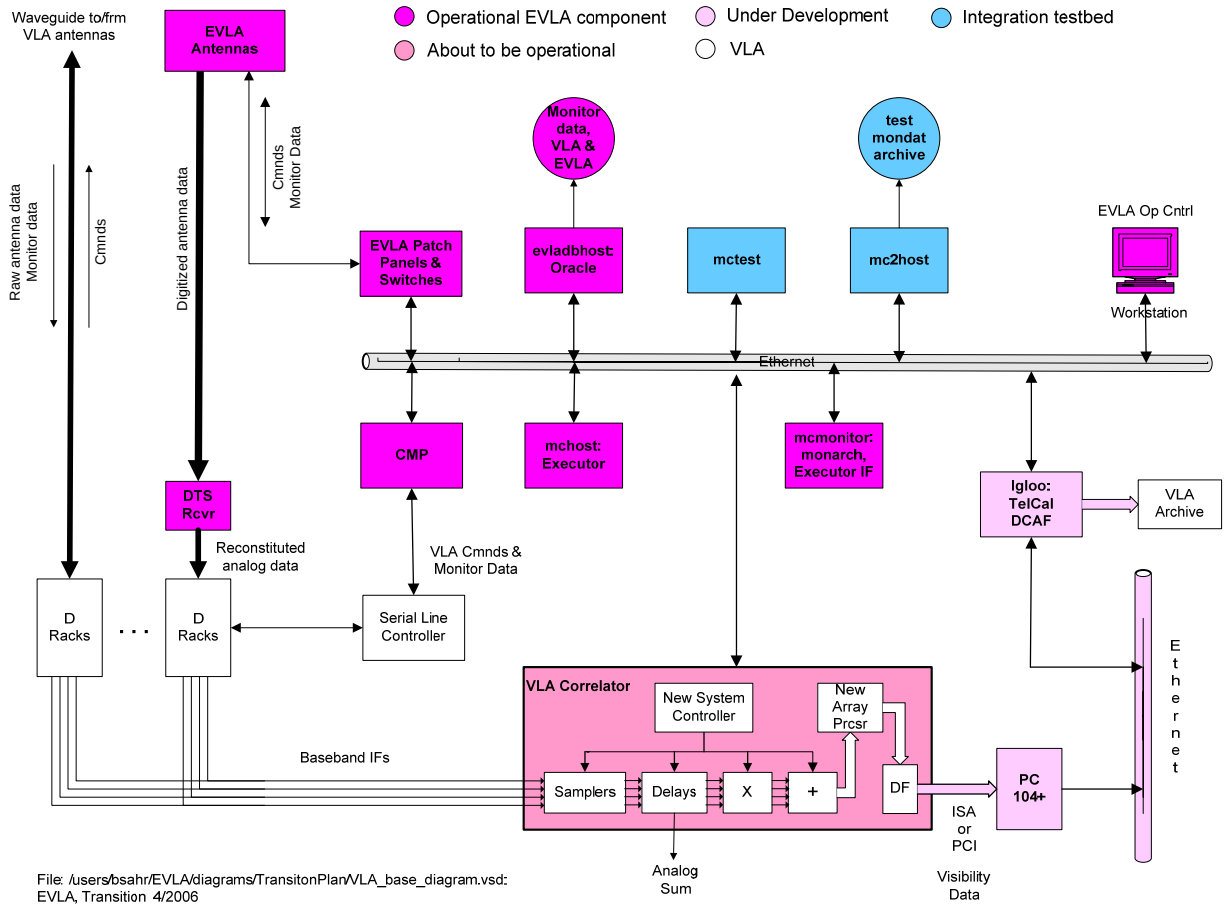


Figure 9 EVLA Transition System Deployment

CMP Design

Hichem Ben Frej

2.3.2.1 Introduction

This document describes the design and implementation of the CMP and its public interface. The CMP is a real-time system used in conjunction with the Serial Line Controller (SLC) to manage VLA antennas during the transition period.

The CMP exposes the VLA antennas to users as virtual antennas with an interface that is identical to the MIB interface presented by EVLA antenna subsystems.

The CMP presents 2 interfaces to the external world:

1. A MIB interface. The MIB interface includes a service port, a data port, and telnet session capability. The data port includes multicasts of an archive data stream, an ostream or observing data stream, and alerts in the EVLA format.
2. A REST interface offers the possibility to query or to command the DCSs through a WEB browser interface. The REST interface is handled by a servlet running within the HTTP server hosted by the CMP processor that runs the TimeSys Linux operating system. At this time the REST option is not being used.

The CMP is used to communicate with the VLA DCSs within an EVLA network. The CMP is responsible of collecting monitor data from the SLC and relaying higher level application/user commands to the VLA DCSs through the SLC. The CMP uses the concept of IP aliasing. IP aliasing is the process of adding more than one IP address to a network interface. One machine represents a virtual set of DCSs. Each VLA antenna, via its DCS, is addressed individually and concurrently.

2.3.2.2 CMP Architecture

The CMP is a cooperation of 2 VME boards:

1. A MVME 162-412 board running VxWorks (CMP-VX) and in contact with the SLC. This board is equipped with a Motorola 68040LC (no floating point hardware) running at 25MHz and 4MB of RAM. The board handles all deterministic and real time operations. The board has access to the waveguide cycle and a time server. Any calculation requiring knowledge of the DCS time and the waveguide cycle is performed on this board. The board includes 2 IP-Unidig-P cards used to DMA in & out data to & from the SLC. Also it uses a CIO-32 card to intercept interrupts from the SLC.

2. A MVME6100 board, running TimeSys Linux (CMP-LX). The board is equipped with 1.267 GHz MPC7457 processor, 1GB DDR memory, and 128MB Flash. The board is used for the public interface, including broadcasting the archive and alarm data, and to handle any non deterministic operations.

The two boards communicate through a shared memory. The 4MB memory of the MVME 162-412 is mapped to the memory of the MVME6100 board which makes all of it accessible.

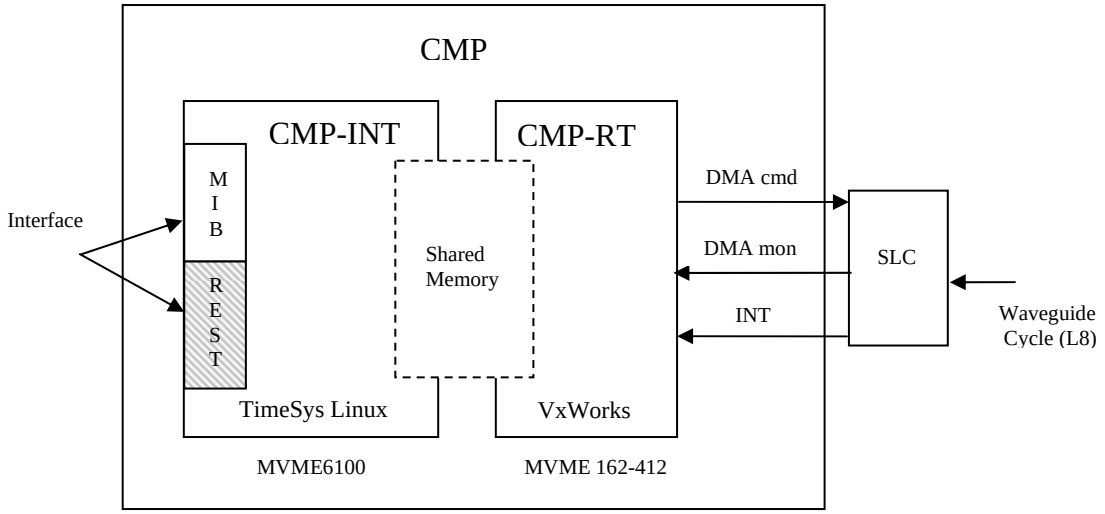


Figure 10 CMP Hardware Architecture

Design Notes:

- The CMP host running TimeSys Linux has its own IP address.
- The CMP host running VxWorks has its own IP address.
- The system is able to handle 0 to 32 DCSs. DCS 0 being a special case: the control building.

- A configuration file contains the setting for each DCS in term of DCS id, IP address and an operation flag.
- 3 XML configuration files: special case DCS 0, for the DCSs hosting VLA antennas, and a minimal configuration file for the DCSs hosting EVLA antennas.
- Each virtual DCS is assigned a unique IP address.
- Each DCS is represented by a thread.
- Each DCS has a service port.
- Each DCS can handle multiple telnet sessions simultaneously.
- Each DCS can handle multiple service port requests simultaneously.
- Each DCS sends alert, archive and ostream data to the multicast address set in the XML configuration file.

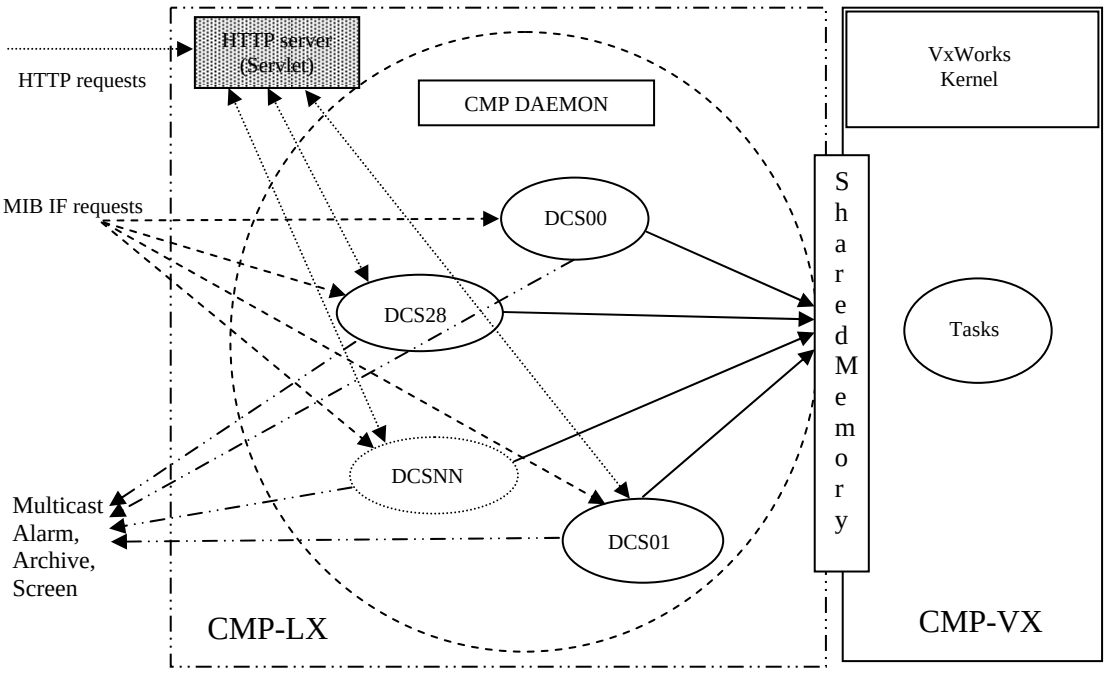


Figure 11 CMP Ports and Interfaces

2.3.2.3 CMP Real Time: CMP-VX

The hard real time portion of the CMP runs on a MVME 162-412 board running VxWorks. It interacts directly with the Serial Line Controller in term of data exchange and interrupts handling.

2.3.2.3.1 *Hardware Interrupts*

The CMP-VX handles 4 hardware interrupts

- Start monitor cycle
  This interrupt signals that monitor data is ready on the SLC.
- End monitor cycle
  This interrupt signals the end of the monitor cycle.
- Start command cycle

This interrupt signals the start of the command cycle; the SLC is ready to accept commands from the CMP.

- End command cycle
  This interrupt signals the end of the command cycle. The SLC is done accepting commands.

2.3.2.3.2 *Modules and Tasks*

The CMP-VX is composed of functional modules and cooperative tasks.

- Time module
  This module groups the tasks and function that handles time in the CMP-VX.
- Sequencer module
  This module groups the tasks and functions used for the sequencing of time deferred commands. The commands are passed from the command scheduler to this module to be executed in sync with the waveguide cycle.
- ACU module
  This module groups tasks and functions used for the ACU commands queuing and ACU calculations.
- Monitor Data Processing module
- Command Processing module
- Phase and Walsh module
  This module groups the tasks to perform phase switching and the Walsh task.
- WatchDog timer task
  This task makes sure that the system is in running state and no task is monopolizing the system. If this task fails to execute, it will cause the system (CMP-VX & CMP-LX) to reboot.

2.3.2.3.3 *CMP VxWorks Startup Sequence*

The MVME162 boots from flash and loads the VxWorks kernel image. A script is launched to expose the MVME162 memory as shared memory to the Linux board. Also it loads needed modules from the mounted directory.

Once all the modules are loaded, the initialization routine for each module is launched which in turns spawns its own tasks. The interrupt service routines are associated with their corresponding interrupts.

2.3.2.4 CMP non hard real time: CMP-LX

The non hard real time portion of the CMP runs on a MVME6100 board running TimeSys Linux. It handles all non-deterministic operations and it is the link to the users.

2.3.2.4.1 *CMPDEAMON*

It initiates the system. It is responsible of the system startup and setting up the virtual DCSs. The CMP daemon is in charge of handling the reboot request for all DCSs.

Every spawned thread keeps its own static information in a global structure. Every thread is running in a virtual separate memory space.

2.3.2.4.2 *MIB-like interface*

2.3.2.4.2.1 Data Port: Archive and Ostream Data, Alerts

The monitor data archive and ostream data, and alerts are multicast in the EVLA format. The data is sent on the same multicast IP address(es) used for the EVLA antennas.

2.3.2.4.2.2 Service port Interface

Each DCS presents its own service port interface. The user sends command(s) directly to the desired DCS.

2.3.2.4.2.3 Telnet interface

Each DCS presents a telnet interface similar to the one offered by the MIB. The telnet to the managed DCSs does not interfere with the telnet to the main IP address which is used for debugging and diagnostic.

2.3.2.4.3 *HTTP interface*

The REST interface offers the possibility to query or to command the DCSs through a WEB browser interface. It was agreed that the servlet handling the REST interface will be communicating with each DCS through the DCS's service port.

2.3.2.4.4 *CMP-LX Startup Sequence*

The MVME6100 board boots from flash and loads the Linux kernel image. Once the system initialization is complete, a shell script is executed to:

- Set up the IP aliasing.
- Mount the needed directory to access the API data from the host "Miranda".
- Launch the cmpdaemon process.

The cmpdaemon parses a configuration file to determine the number of DCSs to spawn and the IP address associated with each DCS. The configuration file has the following format:

DCSxx [space] IP@ [space] VlaFlag, where:

- DCSxx: is the DCS number in decimal.
- IP@: The IP address assigned for the DCS.
- VlaFlag: It is used to distinguish VLA and EVLA Antennas. The DCSs hosting EVLA antenna use a minimal XML configuration file.

This file has to be maintained with every new array configuration, addition of EVLA antennas, and removal of VLA antennas.

For each entry in the configuration file, the cmpdaemon spawns a thread to handle a DCS.
The first task for a thread is to parse the XML configuration file to determine the logical points.
There are 3 types of XML configuration files:

- XML file for DCS00 (special case)
- For DCS01 – DCS37, there is 2 sub-classes:
    - XML file for the DCSs hosting VLA antennas.
    - Minimal XML file used for the DCSs hosting EVLA antennas.

Once a thread is spawned it becomes autonomous and behaves like an independent MIB. Each DCS's thread accesses the monitor data in the shared memory. Each DCS's thread writes the

commands into the shared memory through an exclusive access mechanism. Only one thread will have access to the commands area in the shared memory at any given time.

2.4 *Correlator Monitor and Control (CMCS)*

2.4.1 VLA Correlator

2.4.1.1 New VLA Correlator Controller

2.4.1.2 Data Output Path (Visibility Pipe)

There is a board in the new correlator controller, known as the integrator interface board, where input from the VLA correlator integrator output from the array processor, and the interface to the correlator data formatter all converge. The visibility pipe will be implemented by modifying the integrator interface board to accept a PC104 board and its ISA bus as a daughter board. Visibility data will flow to the PC104 board via the ISA bus, and will then be made available to the EVLA Monitor and Control System via an Ethernet interface on the PC104 board.

Figure 12 is a diagram of the new VLA correlator controller showing the visibility pipe.

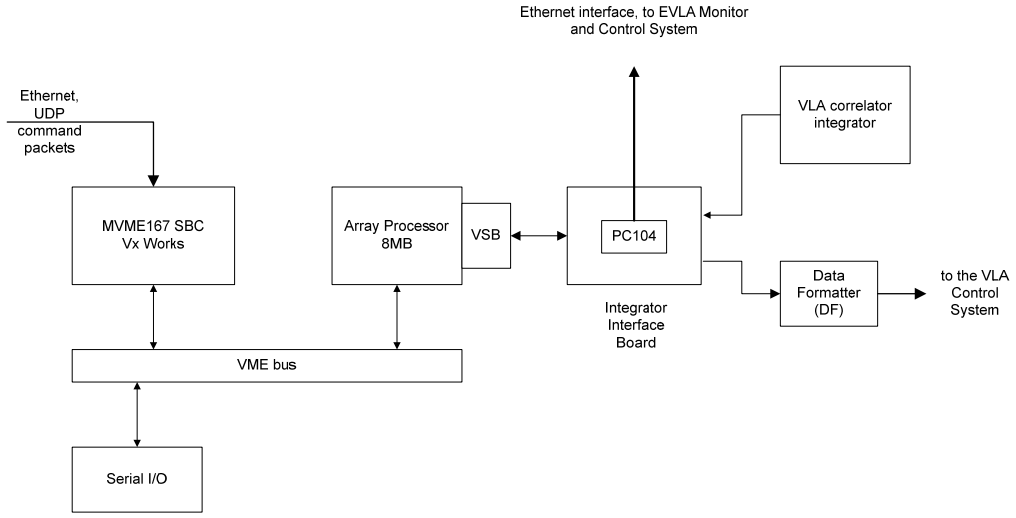


Figure 12 Visibility Pipe

2.5 *VLA TelCal (ITelCal)*

VLA TelCal or Interim TelCal (ITelCal) will occupy within the Transition System an expanded version of the role played by the program Antsol in the VLA Control System. Antsol is a software process that solves for the complex gains of the antenna. These results are then used for a variety of purposes, including:

- data quality displays (the “F” display in the VLA system)
- operational purposes including Tsys estimates

- real time determination of focus & pointing offsets
- phasing the array
- as a post-hoc diagnostic tool to determine when an antenna began to misbehave
- the contents of the Antsol files are piped to the AOC and placed in a flux calibrator database for use by astronomers

ITelCal currently exists as a component of the Transition System. As of early November 2006 it:

- produces complex antenna gains and multicasts them as XML documents
- produces pointing solutions and multicast the results as XML documents
- writes a file of pointing solutions, for all pointing scans, that readable by the software used to develop pointing model coefficients

This level of functionality is sufficient to support development of the same applications as those built on the Antsol results, to implement reference pointing, and to reduce the data from pointing runs for the determination of updated pointing model coefficients.

A data quality display that serves a function similar to the "F" display in the VLA Control System has been developed, and EVLA antennas have successfully participated in a phased array observation. Reference pointing has been implemented.

Eventually, ITelCal will be extended to:

- produce focus solutions, multicasting the results as XML documents and writing the solutions to disk
- produce delays, multicasting the results as XML &/or writing the results to disk

ITelCal currently obtains the archive records upon which it operates from the staging area (the system named "igloo") that is written to by the program in the VLA control system that produces the archive records. This staging area will continue to be a component of the EVLA Monitor and Control Transition System. IDCAF will write to it (see Figure 3, Transition System DataFlows, page 10) and ITelCal will continue to consume archive records from this source.

2.6 *VLA DCAF (IDCAF)*

IDCAF is being written by Walter Brisken of NRAO. The software design is Walter's, the text contained in this section is based on a discussion with Walter, and the diagram is based on a version supplied by Walter.

The purpose of IDCAF is to collect visibility data and the relevant meta-data, format this data into archive records and write those archive records as a multicast and to a disk-based staging area. IDCAF will also perform flagging of bad data in the archive records.

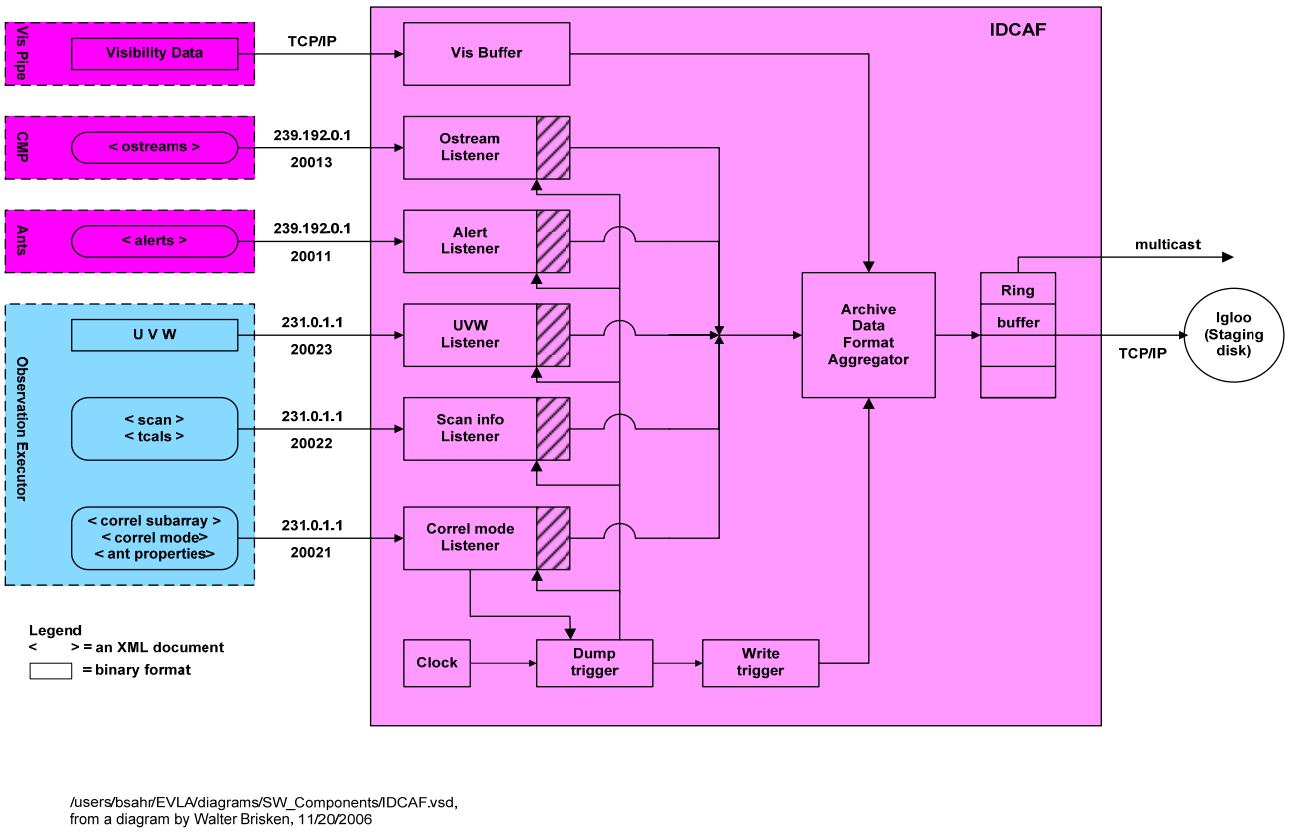
Figure 13, given below is a diagrammatic representation of IDCAF for the Transition System. The boxes on the left hand side of the diagram represent data sources. XML and binary formatted data is sent by these data sources either directly to IDCAF (for the case of visibility data) or as multicasts

that are subscribed to by IDCAF. Buffers/Listeners collect this data, and, on a dump trigger, write copies of the data relevant to the current integration time into mutex protected data areas. (The dump trigger, in effect, causes the data to be latched.) The striped areas in the listener boxes in the diagram represent these protected areas. These protected copies are made to avoid confusion with the arrival of data that is relevant to subsequent integration periods, and allows the process of creating a properly formatted archive record to proceed at a more leisurely pace.

The dump trigger is based on a simple relationship between the clock and the integration time. Whenever the clock (IAT because we are dealing with the VLA correlator) mod the integration time (as obtained from the correlator mode) is 0, a dump trigger is generated. In other words, it is assumed that the satisfaction of this relationship marks the end of an integration period.

Once the process of creating a protected copy of the data relevant to the just completed integration period is complete and the visibility data for that integration period has arrived, the meta-data is used to fill a skeleton of the archive record data structure. This process is done one subarray at a time. When this data structure is combined with the visibility data, the archive record is complete, and it is written - as a multicast, to a ring (circular) buffer that will be several integration periods deep, and, via TCP/IP, to a processor (igloo) whose disk is used as a staging area for archive records.

The ring buffer has been included in the design to allow for the possibility of access to the archive records for the past several integration periods via shared memory.



/users/bsahr/EVLA/diagrams/SW_Components/IDCAF.vsd,
from a diagram by Walter Brisken, 11/20/2006

Figure 13 IDCAF Design

2.7 *User Interfaces*

2.7.1 The Array Operator Screen

Figure 14, given below, is a screenshot of the Array Operator Screen that have been developed for the EVLA Monitor and Control System. The Array Operator Screen is the main screen used by the operators for monitor and control of the array.

The screen pictured is from version 0.9.3 of the EVLA Operator Software, released on 11/01/2006.

The upper left hand portion of the screen, labeled “SCRIPTS”, is concerned with job submission. It shows the currently active control script, scripts queued to be run, and a scrollable list of scripts that have been run. Selecting a script in the history section gives details on the job in the pane immediately below the scripts pane.

Figure 14 The Array Operator Screen

Not apparent from the screen shot is the fact that the scripts pane can be used to submit a job to the Observation Executor. A right click over the legend “SCRIPTS” produces a menu that offers a number of choices including “Run a Script” and “Cancel/Abort Script”. Selecting “Run a Script”

will produce a popup window that allows a script and the antennas to be associated with the script to be specified. Aids for browsing to the desired script and for antenna selection are included.

Still on the left hand side of the screen, there is a pane labeled "ANTENNAS". It provides a scrollable list of all antennas, EVLA as well as VLA, which gives the azimuth, elevation, azimuth error, elevation error, and iconic indicators for the state of computer control, digital position mode, antenna on source and the state of the emergency stop for each antenna.

Immediately below that pane is a bar graph that shows the azimuth and elevation of each antenna, with EVLA antennas in blue. This display is experimental, to determine if it provides useful, "at-a-glance" information.

The center of the Array Operator Screen is a thumbwheel scrollable display of the antennas along each of the three VLA arms. The thumbwheel allows the display to be expanded or compressed, with markers for the A, B, C, and D configurations of the array. Each antenna icon is labeled with its antenna ID, for EVLA antennas the quadrapod structure is shown in blue, the antenna icons rotate to show azimuth, an antenna for which a data update has failed will be circled in red, and mousing over an antenna will produce a legend giving the antenna id, current azimuth, host name, DCS number, modem number and distance form the center of the array. Double clicking on one of the antenna icons will produce an ACU display for the selected antenna, and there are plans to associate a critical functions menu with a right click on an antenna icon.

Immediately below this center display pane is a CONSOLE pane that displays messages produced by the Observation Executor.

On the right hand side of the Array Operator Screen is a series of information screens. The upper right hand pane displays time in 5 different formats – Year.Day, MJD, UTC, IAT and LST. The next pane down is a weather display and the 3rd pane displays alerts. The alerts pane is a replacement for the VLA Checker screen. If one selects a particular alert, the alerts pane reduces in size in terms of the numbers of rows and reveals a pane that gives additional information on the selected alert. Right clicking on the label "ALERTS" produces a menu that includes the option of selecting a number of alerts filters.

The Scripts, Antennas, Weather, and Alerts screens are all available as standalone "screenlets", selected via the "Screens" dropdown menu in the Array Operator Screen menu bar (running horizontally along the top of the Array Operator Screen.

2.7.2 Module/Subsystem Screens

Via the "Screens" dropdown menu in the Array Operator Screen menu bar (and by other means) it is also possible to select a number module/subsystem screens. Currently specialized screens for the ACU (antenna control unit), FRM (subreflector focus/rotation module), F317, F320 (front end, i.e., receiver controllers), the L301 (a synthesizer), and the M302 and M303 (utility modules) are available. Figure 15 is a screenshot of the ACU screen. Figure 16 shows the FRM screen.

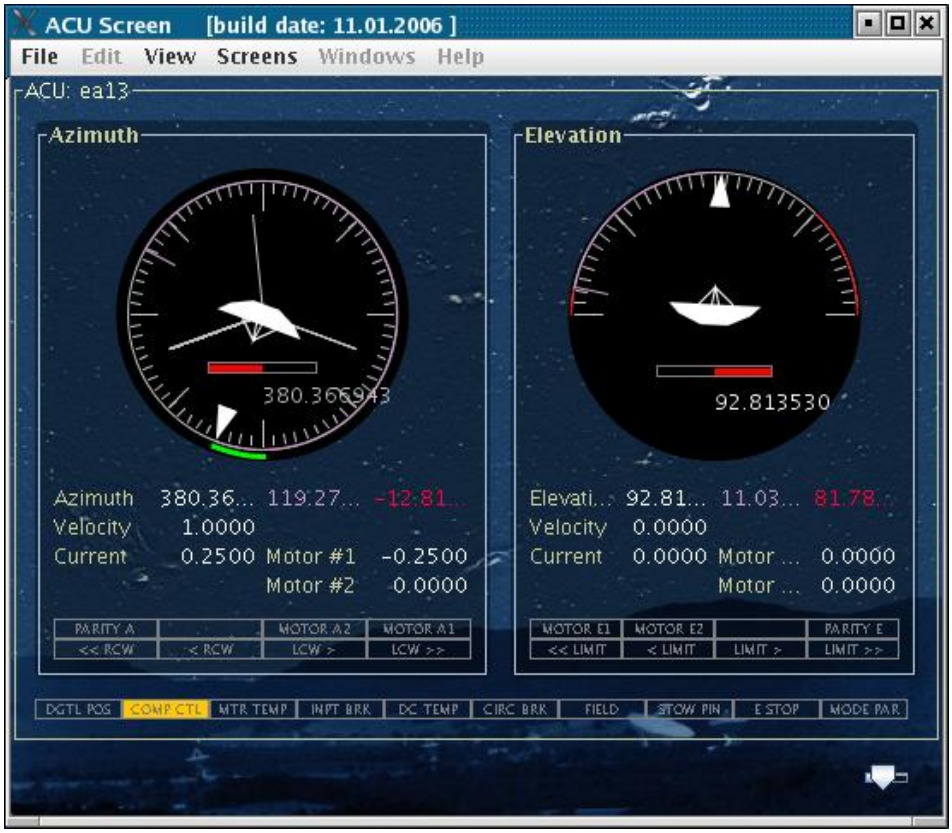


Figure 15 ACU Screen

Figure 16 FRM Screen

2.7.3 VLA Antenna Screens

The bulk of what is meant by "VLA antenna screens" is the development of an EVLA equivalent for the Array Operator Interface (AOI) software that is part of the VLA Control System. AOI consists of approximately 24 different screens, each screen dedicated, for the most part, either to overviews essential to monitoring the status of an observation, or to monitor and control of VLA antennas. For the EVLA, monitoring the status of an observation is done via the Array Operator Screen. For the VLA antennas, a list of AOI screens, the information displayed on each, and the sources of that information within the system has been produced. Further, a unified user interface to allow the monitor and control of both EVLA and VLA antennas has been prototyped and is under development.

2.7.4 Critical Functions Screen

The Critical Functions screen (Figure 17) will allow an E-Stop (emergency stop), an ACU reset, a Critical Power reset, a Track (digital position mode) command, Stow, Park, or Standby commands to be set or sent to any antenna, selected group of antennas, or to all antennas in the array.

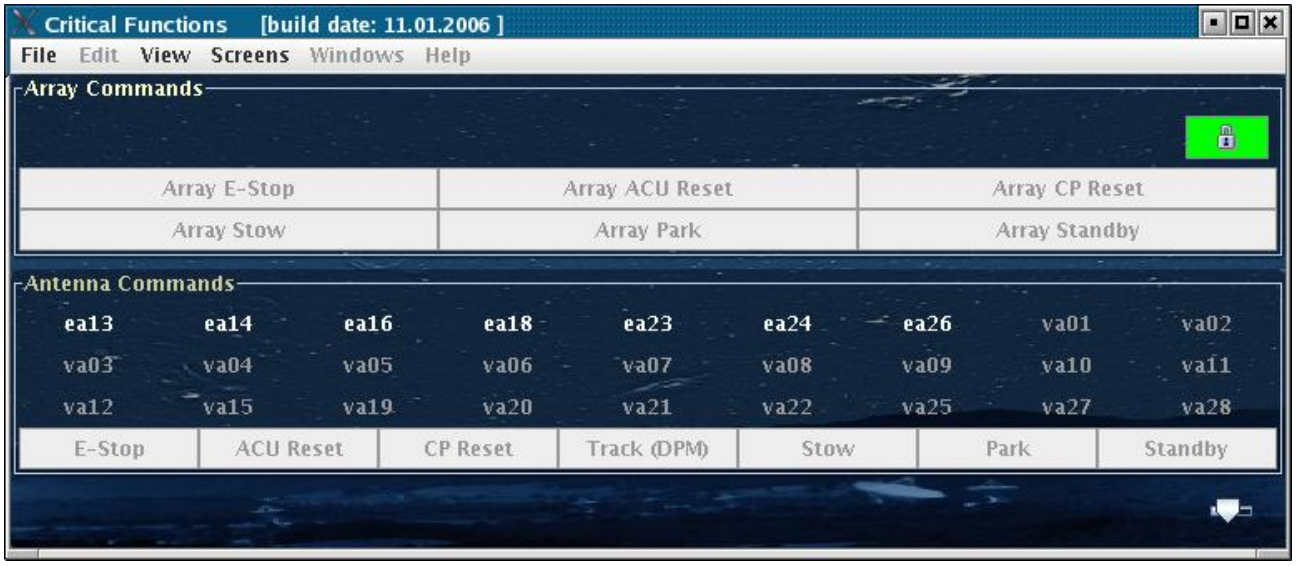


Figure 17 Critical Functions Screen

The functionality of the Critical Functions screen depends upon the installation of the M302 and M303 modules in the antennas. Final prototyping of the M302 and M303 hardware is nearly complete. Final field testing M302 module is scheduled for the end of November 2006.

2.7.5 Device Browser

The device browser was described in section 1.5.1.3, Antenna Control, page 11 of this document. To quote from that section, it is "a low level user interface that provides a GUI-based interaction with any device implementing the MIB interface. Its capabilities include the display of a tree of all devices with all monitor points and command points for a single or multiple MIBs, the display of all attributes of a selected monitor point or command point, the ability to set the values of monitor and command point attributes (which means that it can be used to command antenna subsystems), and plotting, in real-time, of multiple, user selected monitor point values. Figure 18 is a screenshot of

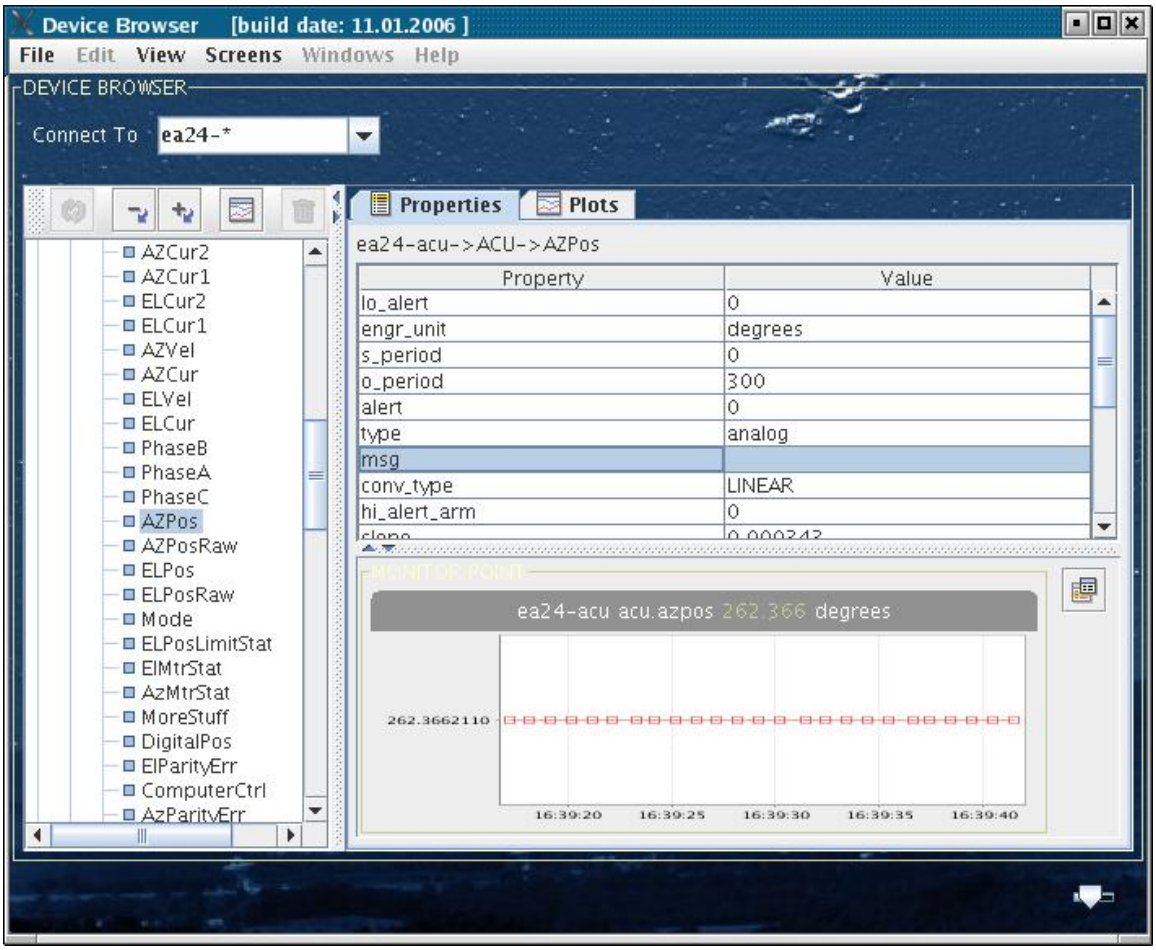


Figure 18 The Device Browser

the Device Browser. The left hand pane shows a portion of the list of monitor points for the ACU in antenna 24, an EVLA antenna, with the AZPos monitor point selected (highlighted in blue). The top right hand portion shows a display of the properties or attributes for the selected monitor point, and the bottom right hand portion is a plot of selected monitor point value.

This brief description of the Device Browser really does not do it justice. It was the first user interface tool developed, has been in continual use and under continual development for several years now, is stable, robust, customized to user requests, and is heavily used by astronomers, hardware and software engineers, and technicians working on the EVLA antennas.

2.7.6 Web Interface to the Observation Executor

Up to this point no mention has been made of an early, web-based interface to the Observation Executor that has been maintained and may someday provide a remote viewing capability for any user from any location. Figure 19, given below, is a screen shot of the web-based interface to the Observation Executor. The URL for this interface is simply http//<host_name>/evla-screens/ and the Executor Interface is the default screen for this URL. (The host name is currently mcmonitor.)

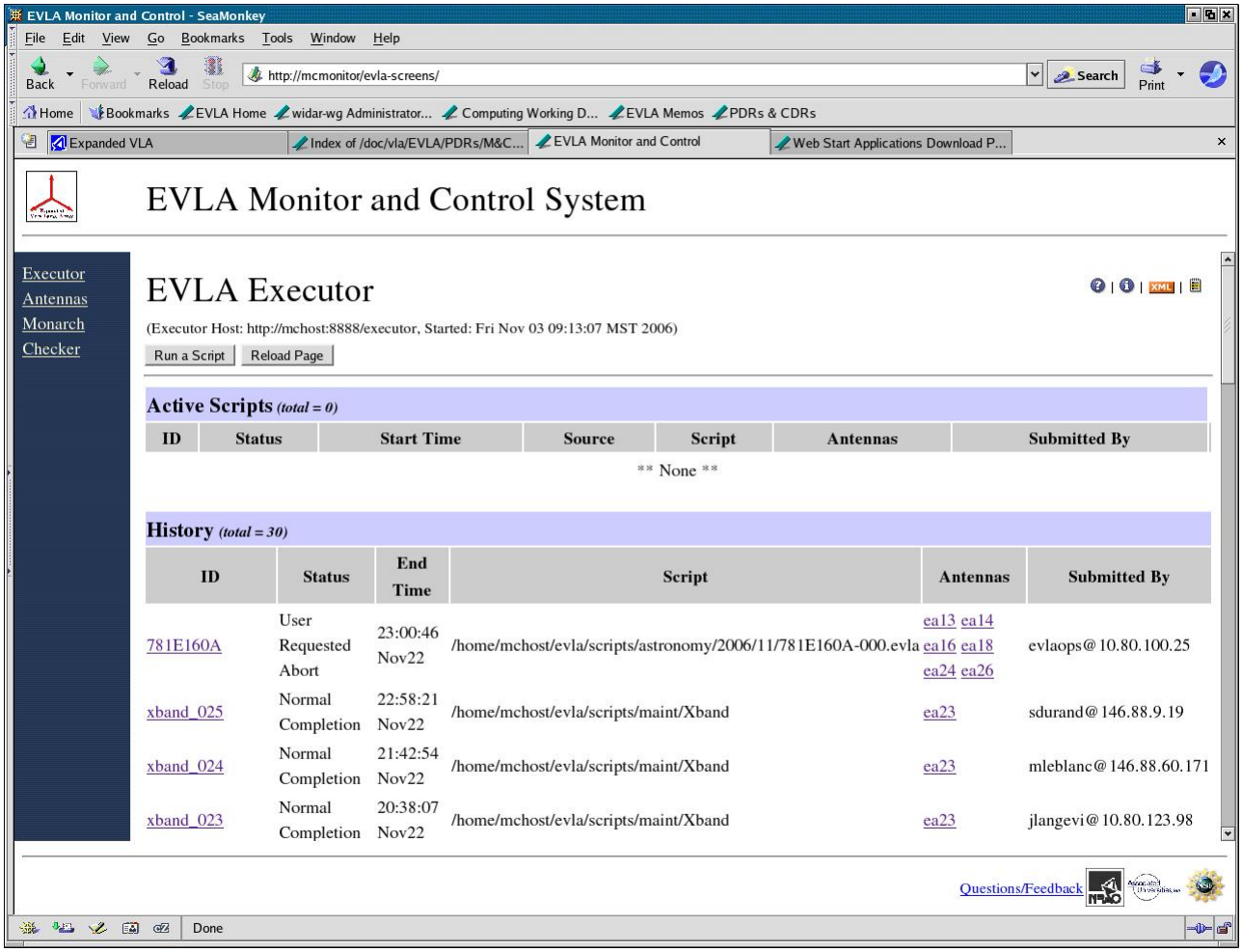


Figure 19 The Web-based Observation Executor Interface

This web-based interface to the Observation Executor provides the same information and capabilities as the SCRIPTS pane of the Array Operator Interface. It shows the currently active script, a history of past scripts, and provides a job submission capability.

To submit a job to the Executor, one clicks on “Run a Script” in the Executor interface, which produces a Job Submission popup (see Figure 20). This popup presents a number of required and optional fields, with provisions for assistance in the completion of those fields. When all fields in the Job Submission popup have been completed, one selects “Submit” in the Job Submission popup, and a process begins that results in the submission of a control script to the Observation Executor for the antennas specified.

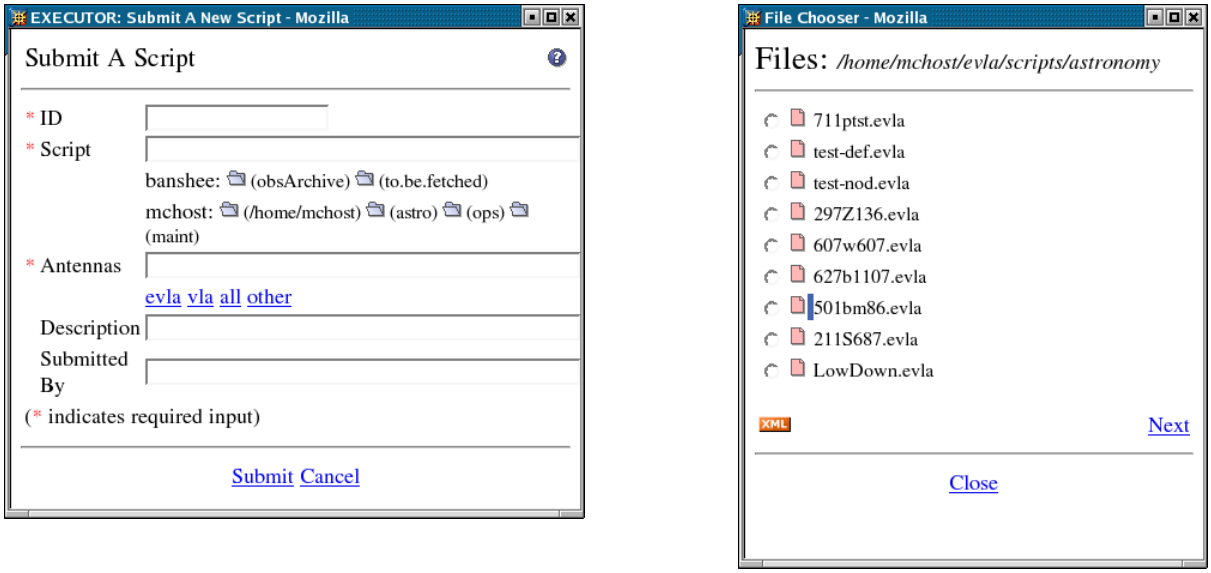


Figure 20 The Job Submission & Script Selection Popups

2.8 *Monitor Data Archive*

2.8.1 Data Ingest

A functioning monitor data archive with a web-based retrieval capability exists and has been in use for approximately 3 years. This archive stores both EVLA and VLA data. The VLA data comes from the CMP, which reformats it into the EVLA format. Currently, the primary data sources are EVLA and VLA antennas, but data from the EVLA test rack (located in the AOC basement), from test benches, and from auxiliary devices, such as the central weather station, is also stored in the monitor data archive.

The basic scheme is that monitor data is multicast as UDP datagrams containing ASCII messages in an XML format. A filler program (Monarch) subscribes to the monitor data archive multicast group, massages the received XML messages into a database friendly form, buffers the messages, and then forwards them to an Oracle database. The filler program needs no knowledge of the content of the data or of the data sources.

The XML format of EVLA monitor data is defined in document # 35, MIB Data Port ICD, Version 1.2.0, C. Frank Helvey 09/30/2004, on the Computing Working Documents web page, located at:

http://www.aoc.nrao.edu/evla/techdocs/computer/workdocs/index.shtml

The rules governing the formation of multicast addresses are also on the Computing Working Documents web page, as document # 42, MIB Multicast Address Usage, C. Frank Helvey, 12/02/2004.

The monitor data archive is not considered to be in its final form. Work is being done now on issues of table management that promises more efficient use of disk space and faster searches, especially for the case of searches that span day boundaries.

There is also an issue of anticipated ingest rate. An EVLA antenna has approximately 7 times the number of monitor points as a VLA antenna (~ 2100 monitor points per EVLA antenna vs. ~ 300 monitor points for a VLA antenna). Even in the face of this increase, the ingest rate would be manageable if the archive rate used for the EVLA monitor points were the same as the archive rate used in the VLA control system (approximately one every 15 minutes). [1] However, the current average archive rate for an EVLA monitor point is closer to once every minute, and in some cases, a 1HZ archive rate is used, if not on a permanent basis, then for weeks at a time.

The first line of defense in dealing with the ingest rate will be to reduce it – by working with the Electronics Division to review and adjust the archive rates for every monitor point in the system. Archiving rates are adjustable on a per monitor point basis. There is both an out-of-alert and an in-alert archive rate for every monitor point, the latter rate generally being set to a higher value to permit finer resolution to facilitate debugging. The default values for the out-of alert and in-alert archive rates of a monitor point are governed by the values set in a XML file that is read by a MIB (and the CMP for VLA monitor data) when it boots. Additionally, these values can be altered in a running MIB using simple "set" commands. A value thus altered will remain in effect until either another "set" command is issued or the MIB reboots.

2.8.2 Data Retrieval

The current web-based monitor data retrieval interface is limited, and limiting. It contains fields for slot ID (roughly equivalent to an antenna subsystem name), device name, and monitor point name that are case sensitive, and the search will fail if the time range is not fully qualified, including a specification for seconds, even if the seconds are 00. The interface does not provide feedback about the ongoing status of a search in progress, data delivery is awkward and ineffective for large volumes of data, and it is not possible to search for multiple monitor point values in a single query. Further, if one does not know the exact strings to use for the slot ID, device name, and monitor point name, a preliminary search of the archive contents must be made to determine these values.

An effort is underway now to rectify these shortcomings. The design for the new web-based monitor data retrieval interface will provide a series of cascading menus, followed by a prompt for the time range to fully specify one item in an accumulated list of multiple monitor points that can be submitted as a single query. The initial goal for data delivery is to find an efficient and effective way of delivering, for a web-based query, a day's worth of data for one antenna, for a single monitor point that has been archived at a 1Hz rate – 86,400 monitor point values.

2.8.3 Data Pruning

Tests of the use of a data pruning program to reduce overall data volume in the monitor data archive have been performed. The pruning program reads the data tables and performs operations such as the removal of values that are identical to one another. Currently, one of the problems with data pruning is that the database is perturbed rather frequently in a manner that interferes with the operation of the pruning program, and it is not restarted. Over the long term, once the monitor data archive is more stable, data pruning may prove to be a useful tool for reducing the overall size of the monitor data archive.

---

[1] See "EVLA Monitor and Control Communications Infrastructure", Version 1.0.2, Bill Sahr, 02/16/2005, section 3.1.4.3.1 for an estimate of EVLA monitor data archiving rates. This document is document # 39 on the EVLA Computing Working Documents web page: http://www.aoc.nrao.edu/evla/techdocs/computer/workdocs/index.shtml.

2.9 *Alerts*

The following material is an excerpt from a yet to be published document on a design for alerts in the Final version of the EVLA Monitor and Control System. The alert system as implemented in the Transition System is a simplified version of the design described in the following sections. The Transition System lacks the Intermediate Servers described below. Instead low level alerts go directly to the Alert server. The Transition System does contain the functional equivalent of an Alert Description Database, but it is simply a table that is memory resident within the address space of the Alert server. Another major departure from the description given in this section is that there is presently no single, uniform Alert Log Client. Logging within the Transition System is currently handled either by individual applications or by a class of applications. For example, the Interim Observation Executor maintains its own log, and MIBs implement a logging scheme that is uniform across all MIBs, but is not used by other applications.

Alerts in the EVLA Software System

Bryan Butler & Rich Moeser

2.9.1 Introduction

This document describes the design and implementation for the elements of the EVLA software system related to "Alerts". An Alert is a signal generated at some level within the hardware or software that indicates that hardware or software is not operating as it should and whether or not data is affected. The Alerts must be made available to various parts of the software system, as status indicators, maintenance indicators, data flagging indicators, or in extreme cases as an indicator that the operator should intervene and stop observations. In this document we present a subsystem design.

2.9.2 Overview

The Alert subsystem for EVLA software must facilitate the transmission of indicators that hardware or software is misbehaving to other parts of the software that need it, as well as displaying it for interested parties. It is made up of several main components:

- MIB and CMIB software;
- Intermediate Servers;
- Alert Server;
- Various databases;
- Clients.

The overall subsystem should transmit Alerts, along with ancillary information, through various parts of the subsystem to the clients. Any client is supported, by adhering to a well-defined communication protocol and structure of the information passed.

We consider five main sources of Alerts that will have to be organized in this way (and hence dealt with by the overall subsystem):

1. Antenna MIB;
2. Correlator MIB (CMIB);
3. Auxiliary Device (AD) MIB;
4. Control and Monitor Processor (CMP);
5. Software.

The antenna and correlator MIBs are self-explanatory, the AD MIBs include any not in the other two categories, for example the Site Test Interferometer, the central weather station, MIBs not

associated with antennas (L352, L353, etc.), MIBs in test racks in the AOC, etc., and can be either MIBs or CMIBs. The CMP is the hardware and software entity that connects the older VLA antenna hardware into the EVLA software system, and is only present during the transition from VLA to EVLA.

Figure 1 shows a block diagram of the overall Alert subsystem, each component of which will be explained further.

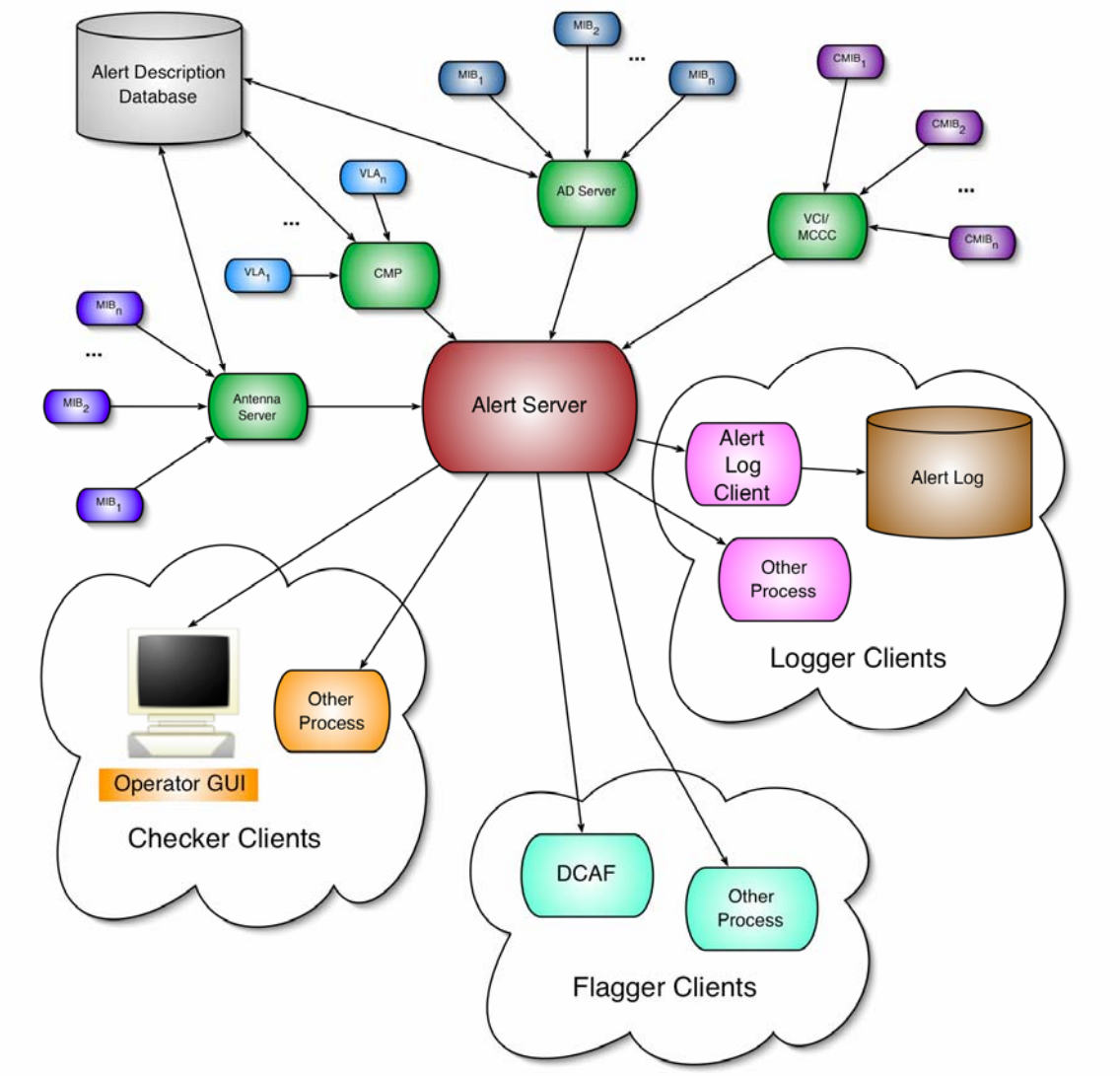


Figure 1. Block diagram of the Alert subsystem, illustrating the connections between the main components of the subsystem: MIBs, CMIBs and VLA antenna devices (blue and purple); Intermediate Servers – Antenna, Correlator, Auxiliary Device, and CMP (green); Alert Server (red); Checker Clients (orange); Flagger Clients (cyan); Logger Clients (pink); the Alert Description Database (gray); and the Alert Log (brown).

2.9.3 Subsystem Components

There are six main components of the Alert subsystem:

1. MIBs and VLA devices. The devices themselves, where the lowest level Alerts are generated.
2. Intermediate Servers. These intercept the low level Alerts, perform any fault tree analysis, combine Alerts with information in the Alert Description Databases, and broadcast the resultant Alerts (many will be passed directly through after combination with the database information).
3. Alert Server. This accepts the Alerts from the Intermediate Servers, and communicates them to the clients.
4. Alert Description Database. These contain descriptions associated with each Alert, any action necessary when activated, and flagging information.
5. Alert Log. This is the persistent log (database) of Alerts.
6. Clients. These are processes that wish to receive the communicated Alerts. There are three main types of these:
    a. Checker clients – clients interested only in displaying information about Alerts;
    b. Flagger clients – clients interested in the flagging information carried along with the Alerts;
    c. Logger clients – clients used to log results.

    In principle any type of client can ask for and receive the Alerts, and in addition some clients may combine some of the functionality listed separately above. We list the three main types here only to expose our thinking on the early implementation.

2.9.3.1 MIBs and VLA devices

The MIBs generate Alerts when monitor points get out of range (in some cases higher logic is used, for example combinations of monitor points). There are two kinds of MIBs: MIBs that are in the antenna or in Auxiliary Devices (called simply MIBs), and MIBs that are associated with the correlator (called CMIBs). The VLA devices have no software embedded within them, but are simple electronic devices. They communicate via the CMP, which essentially acts as the MIB software for all of the VLA devices.

2.9.3.2 Intermediate Servers

These are processes that intercept lower level Alerts and combine them if necessary into higher level Alerts. This is the fault tree analysis discussed in EVLA Computing Memo 33 (EVLA High Level Software Design). The Intermediate Servers also take the low level Alerts (which are often just an indicator with no ancillary information) and combine them with more complete information contained in the Alert Description Database.

2.9.3.3 Alert Server

The Alert Server serves as the collection point for Alerts multicast from the Intermediate Servers. The Alert Server then makes Alerts available to clients. It is a single process from which all Alerts can be retrieved. The Alert Server is a natural location for the implementation of alert filters.

2.9.3.4 Alert Description Database

This database contains descriptions of all Alerts for all MIBs and VLA devices, along with "severity levels" (a description of the importance of the Alert), and a requisite action. Formally, each entry in the Alert Description Database has the following fields:

- Name. This is the name of the Alert (typically the same as the name of the monitor point associated with that Alert);
- Flagger. This indicates whether data is to be flagged when this Alert is raised;
- Severity. A level from 0 to 4 indicating the severity of the Alert ;
- Description. A complete description of the Alert;
- Action. A requisite action.

2.9.3.5 Alert Log

The Alert Log is the repository (database) for all Alerts received by the Alert Server. In fact, the process that populates the Alert Log is a special type of Alert Client, with the added feature that it must store (in some persistent storage) the Alerts. The problem of accessing the Alert Log is beyond the scope of this document and will be treated elsewhere.

2.9.3.6 Alert Clients

These are the processes that collect Alerts and do something with them. Some clients will take the Alerts and display them in some way, but there is no reason to deny a Client that simply wishes to collect Alerts (and potentially take action under certain circumstances), and in fact the Logger client will do just that. The primary Alert Clients will be of three types: Checker, Flagger, and Logger clients. Checker clients are interested in the Alerts as an indicator of improper operation, and will typically use GUIs to display this information to operators and engineers. Flagger clients are concerned with how the improper operation affects data. Logger clients simply collect and log the Alerts to disk. In all cases, Clients must be able to specify *filters* for which Alerts they are interested in. For instance, an engineer might only be interested in Alerts involving a particular antenna or MIB.