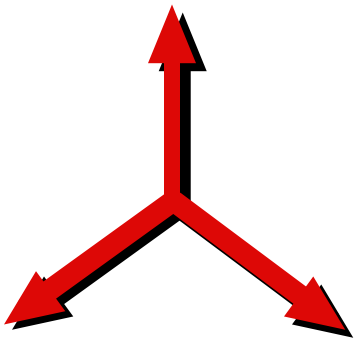
EVLA High Level Software Design



June 7, 2004

NATIONAL RADIO ASTRONOMY OBSERVATORY
P.O. Box 0, Socorro, New Mexico 87801
Operated by Associated Universities, Inc.
Under Contract with the National Science Foundation

Revision History

Date	Version	Description	Author
7-June-2004	1.0	Initial draft	Tom Morgan, Kevin Ryan, Ken Sowinski, Boyd Waters

1 Introduction

1.1 About this Document

This document was prepared by the EVLA Architecture & Design Team at the end of its initial phase of design work. It was produced to serve as a reference companion to the presentation materials used in the first complete review of the overall EVLA software design to be held in mid-June 2004. The intent of this document is to provide more in-depth discussions of the points covered in the mid-June review as well as additional back-up materials in areas not specifically addressed in the presentations.

A great number of people contributed to the contents of this document either directly with written material or indirectly via interviews and comments on various drafts. The design team wishes to thank them for their time and efforts.

Reference materials are listed in the Resources section of Appendix I. In places where requirements are given, those that derive from documented sources are marked with the document number and requirement number within the document. The document number is the same as the listing number in the references. These should be consulted for clarifications and expansions of the brief requirement write-ups in the text.

1.2 Scope of this Document

A description of the overall EVLA software design is presented herein. The contents include a layout of the highest level components of the EVLA instrument conforming to the NRAO Observatory model, descriptions of the data and functional components of the software system conforming to the NRAO Project Model, and discussions on the operational aspects of the major system components and human actions taken while using the system conforming to the NRAO Observing model. By presenting these topics in the context of the NRAO models, commonalities and divergences with other NRAO instruments, primarily ALMA, can be discovered. In the case of commonalities, opportunities for design and software reuse may be assessed and exploited.

Work to date has carried the design down to the level of major software components. Each of these components is addressed in terms of inputs, outputs and behaviors. Input and output data, parameters, and other arguments are presented in as much detail as practicable given what is currently known about requirements, use cases and hardware system design. In certain instances this is quite a lot, while in others only the most general statements can be made. Behavior is discussed in terms of the two or three dominant high level functions of each major system component. Additional details will be worked out by the individual subsystem design teams and is beyond the scope of this effort (see Charge to the Design Team in Appendix I).

In addition, a number of other topics of direct concern to system design are addressed. These include communications, archiving, transition from VLA to EVLA operations,

user interfaces, and other general system services. Requirements employed in the design effort are included along with references to the related sections in the original requirements document or documents.

1.3 EVLA Software

Software needed to operate and maintain the EVLA telescope and conduct scientific observations is being designed and implemented in parallel with EVLA hardware development. The deployment of the new software system and its sub-systems will be staged over the course of the hardware upgrade effort. In some instances existing software will have to be replaced with new systems, in other cases completely new systems automating previously manual operations will be introduced. The deployment of all EVLA software components must be accomplished with minimal disruption to observing activities.

The EVLA Software System as a whole can be characterized as a hierarchically structured, static set of components. All the sub-systems must be encountered in a fixed order during the course of proposing, preparing, scheduling, executing, archiving and reducing the data from an observation.

All functional services in the System will generally be needed during observational activities. There are very few functions that will be performed on time irregular intervals of more that a few tens of minutes. As a result, the vast majority of software functionality must be online and available during observing operations. Failures must be immediately detected and remedied even for components not currently in use. Except for start-up, shutdown, and failure recovery, it is not expected that there will be a large amount of creation and destruction of processes or objects.

The System consists of a number of major components or sub-systems. These will be introduced and discussed one-by-one later in the document. The sub-systems fall into three distinct categories defined by operational context, especially in the area of time constraints imposed on communications.

First are the offline sub-systems: Proposal Construction and Submission, Proposal Management, Program Preparation (including Observation Preparation), and Data Reduction and Archive. Each is an independent software application or in the case of Data Reduction, a package of applications. External communications are limited to the User Interface and an Archive or Database for bulk data I/O. They are run independently on possibly widely geographically located processing hardware. There is a logical ordering to their use (e.g. a Proposal must be submitted before Program Preparation can be done, Data Reduction cannot be done until the observation has been executed) but there are no other restrictions to their use with respect to other parts of the system.

Next are the online Subsystems: Observation Scheduler, Data Capture and Format, and the Quick-Look Pipeline. These subsystems require external communications and hence

must maintain or periodically establish direct communications with other parts of the system. This communication however does not have to be done in “real-time”. It will be permissible for some delays to be present in the delivery and receipt of external data and some data may be sent or received in advance of its need. These sub-systems will perform no real-time operations, and the output products of these subsystems are either not needed to perform real-time functions, or can be determined and delivered in advance of their need by a real-time function.

The third group is the Real-time Subsystems: Observation Executor, Monitor and Control (including both Antenna and Correlator Monitor and Control) and several independent modules such as Calc and TecCal. These components either come in direct contact with the real-time hardware systems, require real-time input to perform their function, or produce output products that are needed to perform real-time functions. At least some of the external communications of each component will be continuous or periodic with a high frequency of occurrence. A premium will be placed on rapid and secure delivery and receipt of data. Communications failures will frequently result in a loss of Science Data.

The online and real-time subsystems will be started together and must remain in a correct functional state for continuous, long term observing progress to be made. Brief outages in online components can be tolerated only to the extent that storage buffers do not overflow or empty waiting for resumption of service. Outages of real-time components in the critical operational path will result in immediate cessation of operations and loss of Science Data until the fault has been corrected.

1.4 EVLA Overall Software Design Process

This document and all presentations derived from it were produced by the EVLA Overall Software Design Team. The team’s responsibility was to develop an overall architecture and design for the EVLA software, including both the e2e and M&C elements. Further details on responsibilities and procedures, including management’s charge to the team, can be found in Appendix I along with reference materials consulted. Relevant acronyms are listed in Appendix II; definitions of frequently used terms that have specific meanings to the EVLA Software Design are given in Appendix III.

1.4.1 Requirements Analysis and Subsystem Decomposition

Requirements with considerable amounts of detail were provided in the general areas of: Science, Engineering, Post-Processing, and Operations. Major requirements from these were consolidated into a single set of EVLA software system requirements (see Resource Document 2). These were analyzed in the context of the existing VLA hardware and software systems and the EVLA instrument model to produce a high level hierarchical layout of the major functional components of the EVLA software systems.

1.4.2 Operational Analysis

Once the main components of the system were laid out, a further analysis of how an observing project would be organized and conducted was done. This focused on the data and information needed to define the content of the project, and the producers and consumers of the data and information. The result was a set of core concepts and terminology, a specification of the connection topology of the main system components and a description of how the observing process will run on the system. This effort was done in conjunction with the E2E Oversight Committee's work to coordinate all NRAO instrument software design and has produced a result that is in conformance with the committee's models. In addition, the coordinated effort has lead to a design that is very similar to the ALMA design at equivalent levels of system detail and should allow significant opportunities for savings through cooperative implementation.

1.4.3 System Components

A specification for each major system component has been developed which includes details of input and output data, and major functional behaviors. The underlying requirements for each component are also provided. This is meant to serve as a high level starting point for the individual subsystem design processes that will proceed at the next lower level.

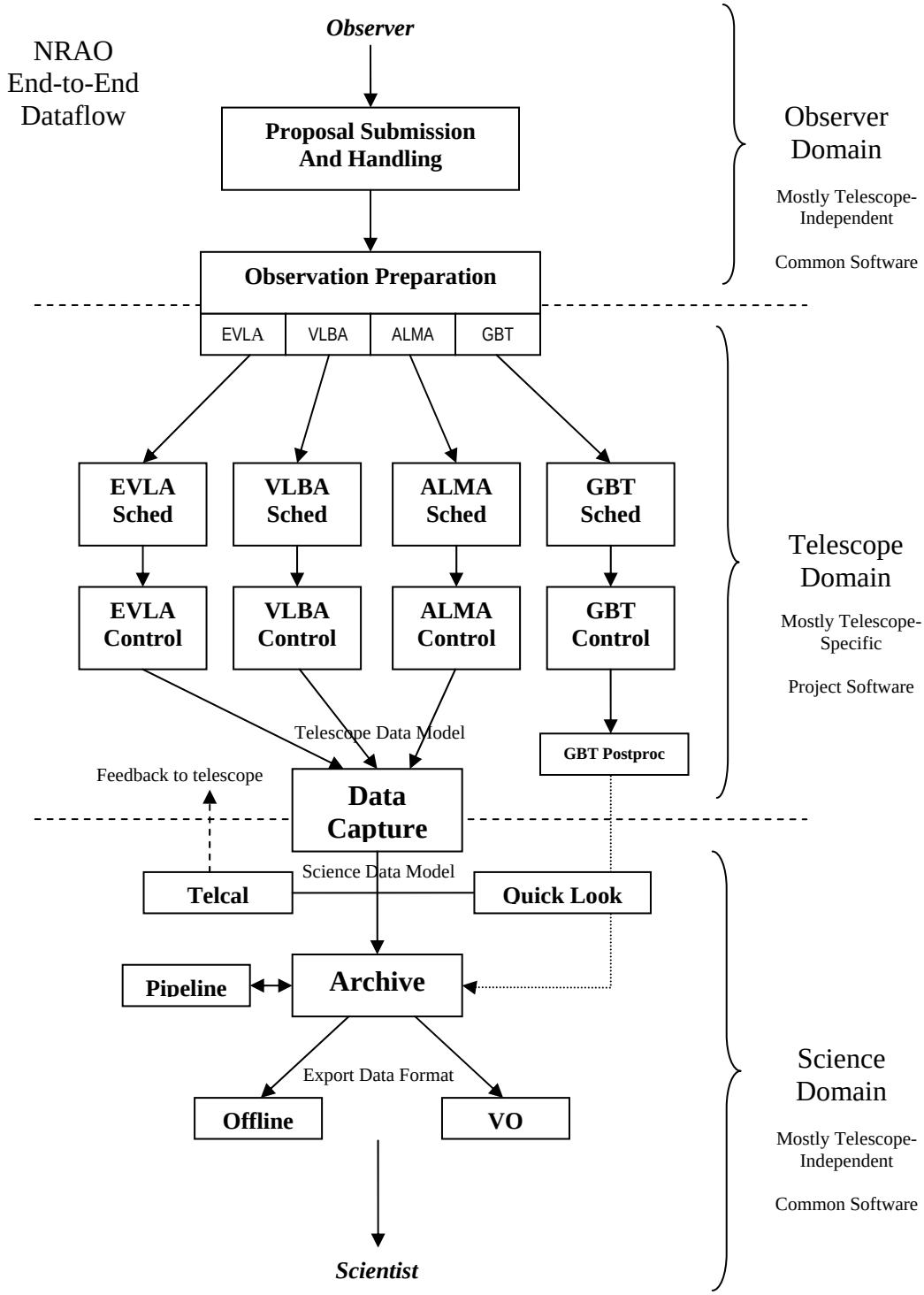
1.4.4 System-wide Services and Utilities

A number of system-wide services and utilities were derived from the specific requirements provided to the team plus general requirements that apply to all software systems of this type and scope. Many of these latter requirements derive from the distributed nature of the processing that will be done and the real-time activities of some of the major components.

2 System Overview

2.1 Observatory

2.1.1 NRAO Observatory Model



2.1.2 EVLA Observatory Architecture

The real-time part of the EVLA system is comprised of the Executor, M&C (includes AMCS and CMCS) and the TelCal and Calc functions. The online part of the system is made up of the real-time components plus the Scheduler, DCAF, QLP and Archive. Proposal Construction and Submission, Program Preparation, Observation Preparation and Data Reduction are considered offline system components.

The high level EVLA software system components are laid out in Figure 1. The components are organized in the order in which they will be utilized for normal science observations. Major data flows are depicted along with some of the critical data types and key parts of the hardware system.

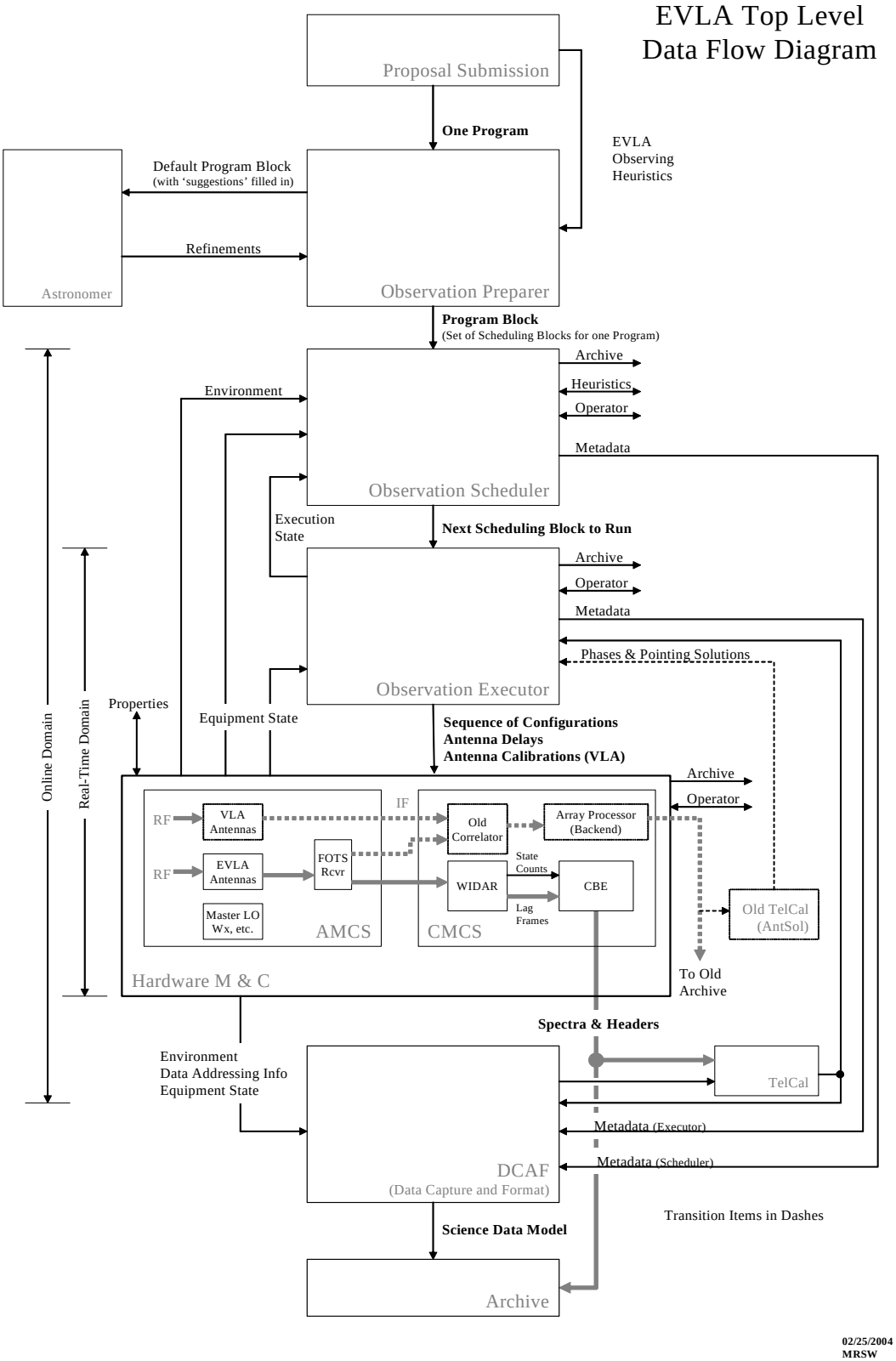


Figure 1. EVLA high level system layout.

2.2 Project

2.2.1 NRAO Project Model

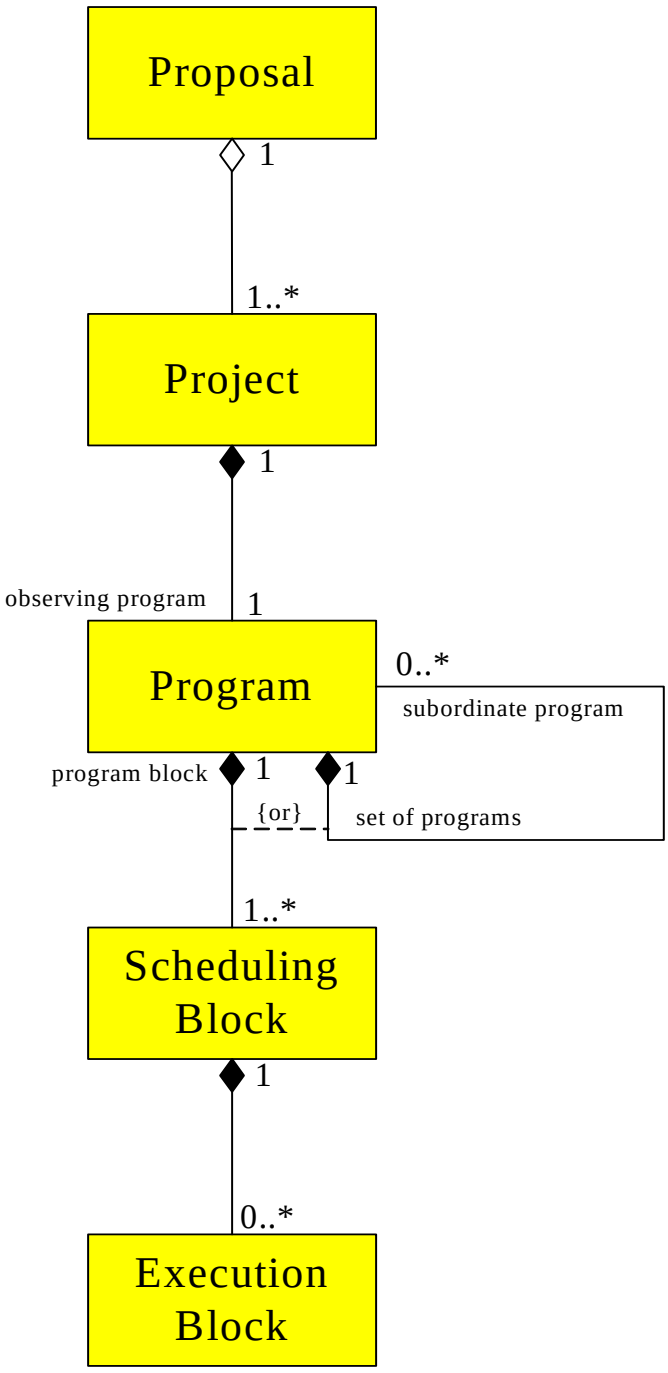


Figure 2. Major Project concepts.

2.2.2 EVLA Project Model

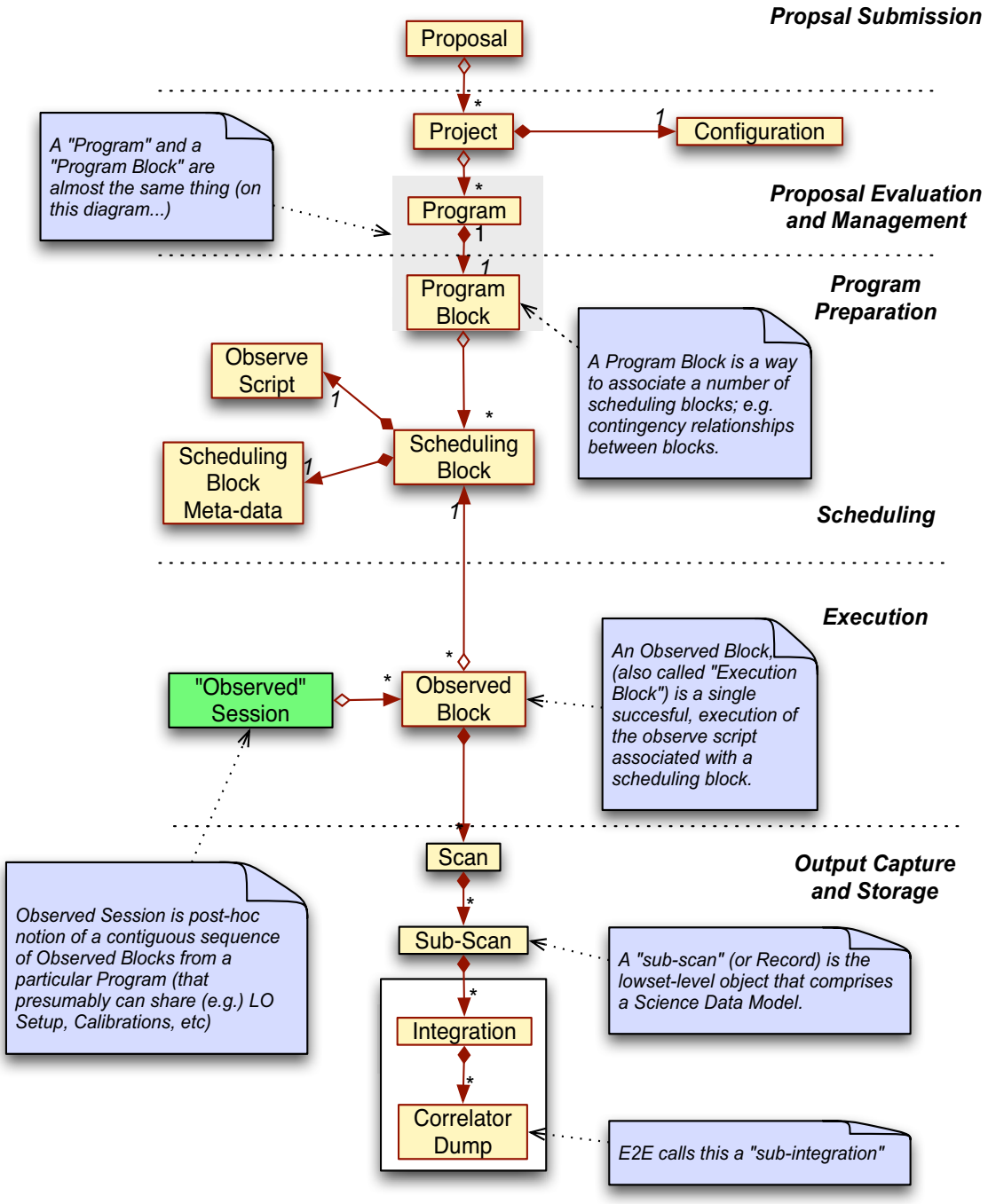


Figure 3. EVLA Project Model UML diagram.

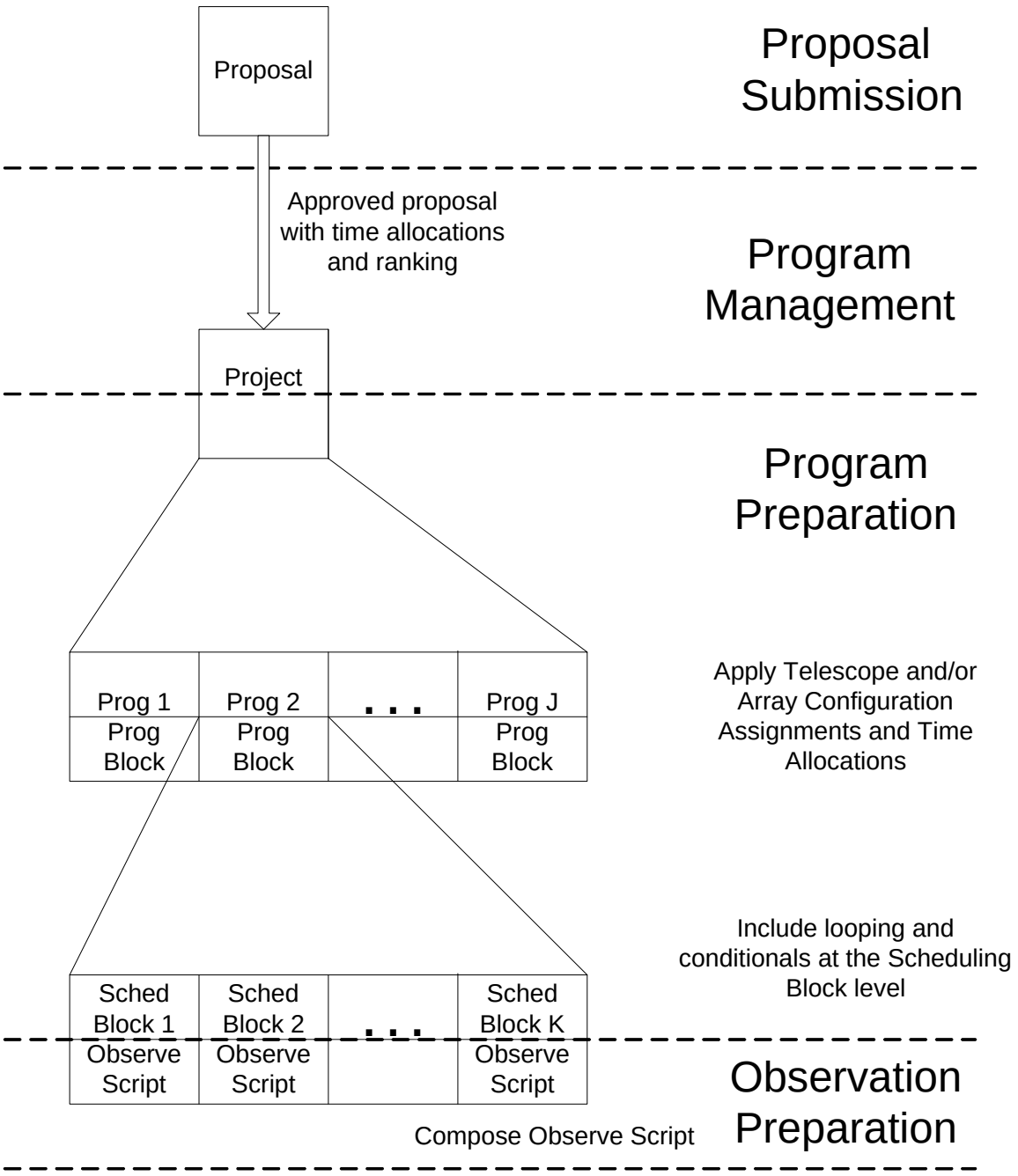


Figure 4. EVLA Hierarchy of Project components from Proposal down to Scheduling Blocks.

Logic at the Program level can be used to set contingency relationships between and among Scheduling Blocks. Recursive layers of Programs and subordinate Programs called Sub-programs are provided for in the e2e Project Model (see Figure 2) and will be allowed, although EVLA is not expected to make significant use of this feature early on (see Figure 3).

Initially, the construction of the Observe Script part of the Scheduling Block will be done via a separate Observation Preparation function. It will be developed early and deployed as a standalone tool. Eventually, it will be absorbed and integrated into the Program Preparation System.

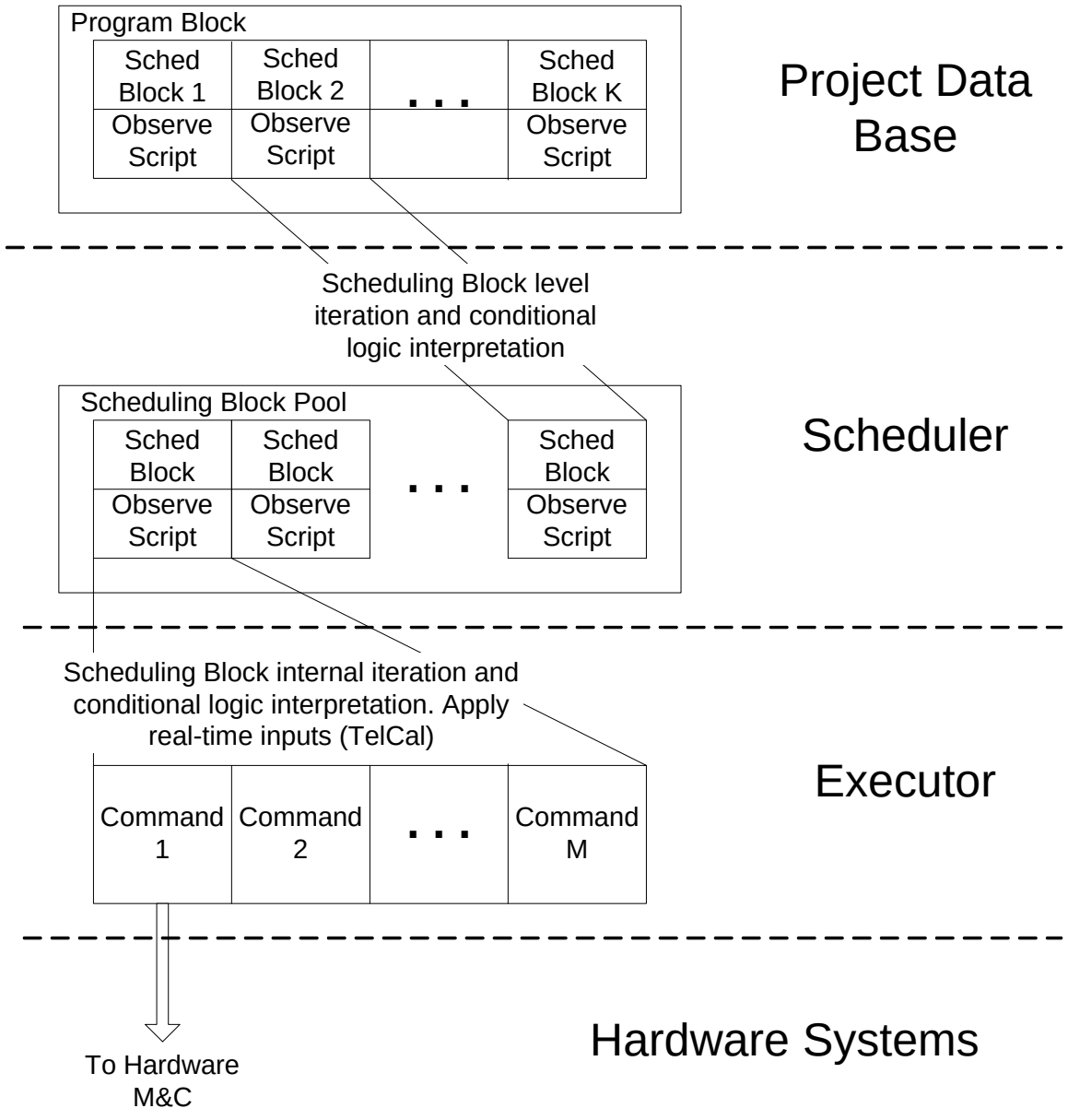


Figure 5. Handling of a Program Block and its constituent Scheduling Blocks by the Scheduler and Executor. Generation of Commands by the Executor.

Program Blocks for a Project are stored in the Project Database. The Scheduler will periodically access the Program Database and harvest Program Blocks that have been marked as “ready” and store them in a local pool. The Scheduler examines the control

logic and scheduling information of the Scheduling Blocks in its pool and determines the best Scheduling Block to submit to the Executor based on priorities, heuristics and an internal scheduling algorithm. The Observe Script portion of the Scheduling Block is shipped to the Executor. The Executor translates the Observe Script into a series of commands to operate the hardware systems (Figures 4, 5 and 6).

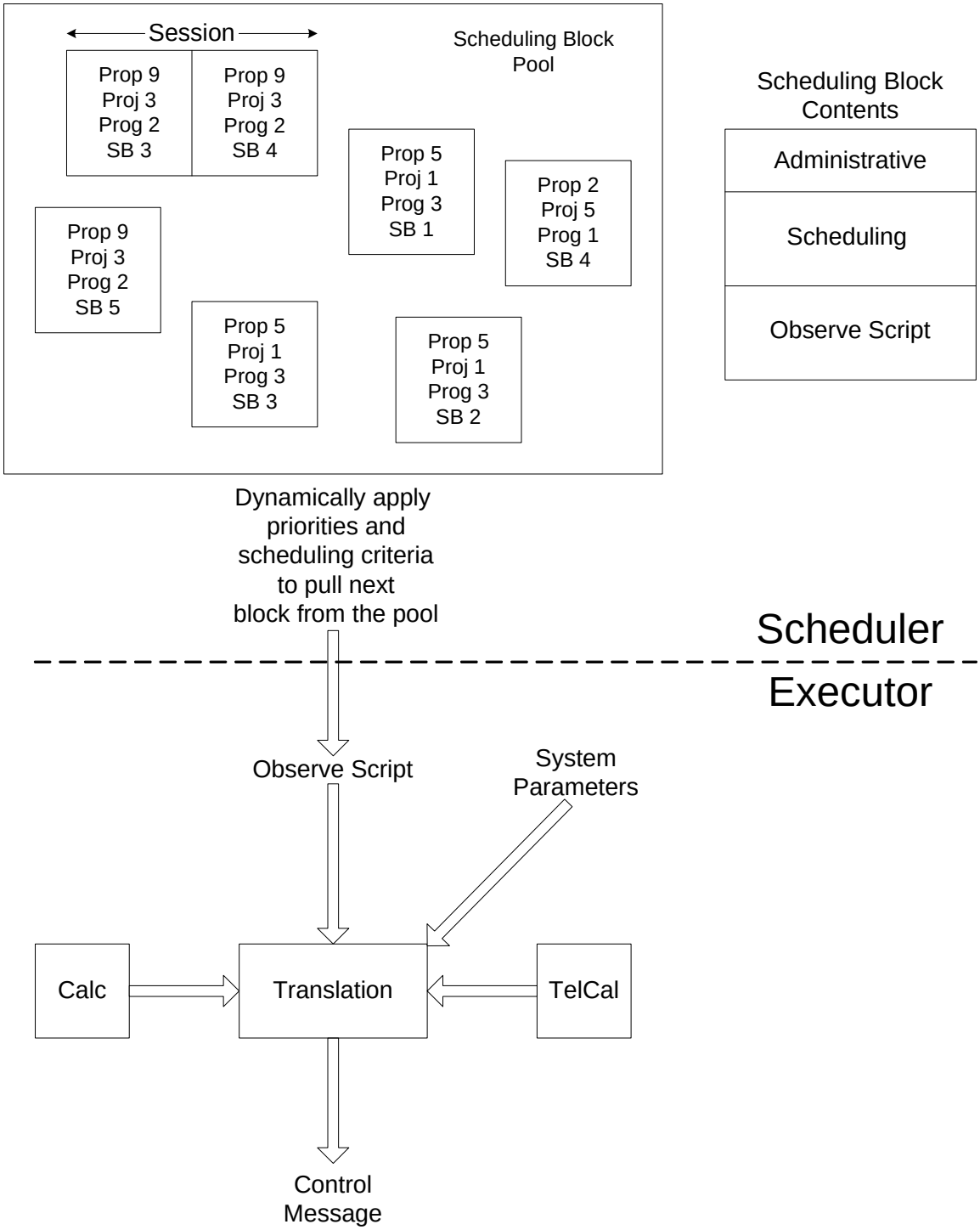


Figure 6. Detail of Scheduling Block content and handling by Scheduler and Executor.

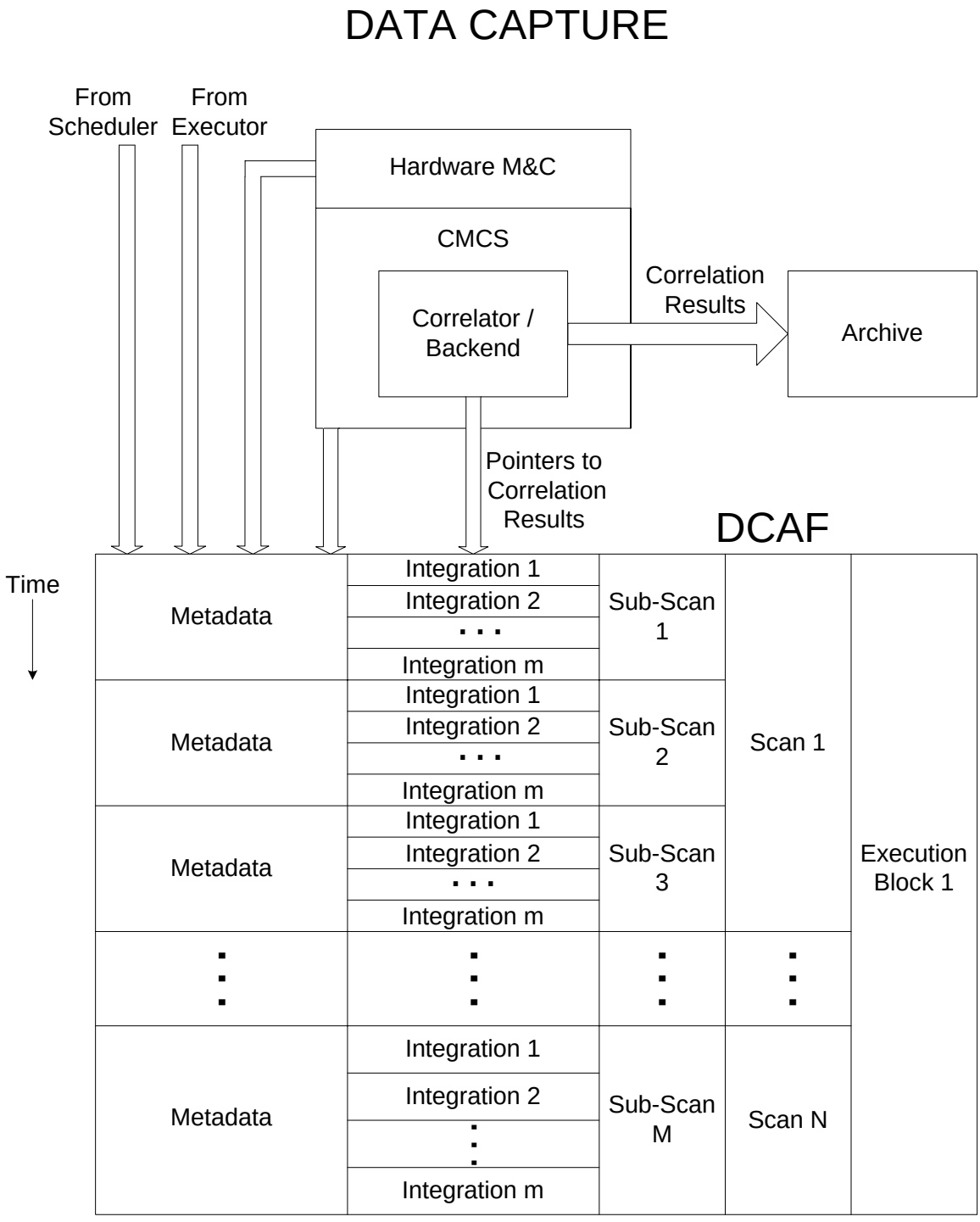


Figure 7. Hierarchy of Project output data products.

Meta-data flows into DCAF continuously (Figure 7). Note that the CBE does not send the actual visibility data to DCAF. This goes directly to the Science Data Archive. The CBE will send reference information that points to the location of the equivalent output visibility data.

All arriving data will contain time tags. Meta-data from the various sources will be associated via the attached time tags and stored in time order, so it is vital that accurate time synchronization be maintained among sending entities. The internal organization of DCAF storage is time ordered from earliest to latest integration in a Sub-scan, Sub-scan in a Scan, and Scan in an Execution Block. Any point at which the output is sufficiently complete, reformatting and exporting to the Science Archive can occur.

2.3 Observing

This Section describes what happens during observing. It discusses the way the main EVLA software sub-systems are employed and how and when the main data entities are referenced. It is meant to follow the NRAO Observing Model which can be found in the Software area of the NRAO Wiki. This is not meant to be an exhaustive treatment; rather its purpose is to cover the major high level actions taken by the component systems and their users and the participation of various parts of the Project Model. Further details as well as the actual mechanisms used to accomplish various activities will be defined during individual sub-system design.

2.3.1 Proposal Creation and Submission

A Principal Investigator acting as author in possible collaboration with coauthors will input all information required by the standard Observing Application Form. Draft versions of the Proposal may be stored by the Proposal Construction and Submission Sub-system for later completion. Copies may be circulated among collaborators and the Proposal Database may be searched for previously submitted proposals. Upon completion, the Proposal is submitted to the NRAO Proposal Database.

2.3.2 Proposal Management

Submitted Proposals are evaluated by external reviewers. Proposals will be rated, and instrument configuration(s) and time allocations will be made for approved work by the Time Allocation Committee and other NRAO staff. This latter task may be done using the Program Management System.

2.3.3 Program Preparation

The Principal Investigator of an Approved Proposal, possibly in conjunction with co-investigators and NRAO Scientific Staff, will create Projects and Programs for deposit into the Project Database with the Program Preparation Sub-system. All administrative, scheduling and execution information needed to make complete Project and Program definitions will be provided. The Observation Preparation function, logically a part of the Program Preparation Sub-system, will be used to prepare the actual Observe Scripts for the individual Scheduling Blocks that make up the Programs.

Each lowest level Program or Sub-program (subordinate Program) in a Project's hierarchy of Programs and Sub-programs has a single corresponding Program Block of Scheduling Blocks. Program Blocks are marked ready for scheduling once they have been completely defined. All Scheduling Blocks in a Program Block must be completed before the Program Block can be scheduled. Program Blocks for a Project may be made ready for scheduling one-by-one, in groups or all at once. Program Blocks for different Projects in the same Proposal may be made ready for scheduling in any order. Final approval from NRAO staff may be required before a Program Block can be marked ready for scheduling.

The Project Database will also be used to store special administrative Projects. These will be used by NRAO staff for testing, maintenance and other internal uses.

2.3.4 Sub-arrays

Multiple astronomical sub-arrays will originate from the same Scheduling Block and as a result will not be apparent to the Scheduler. All antennas needed by each sub-array must be specified in the Scheduling Block as well as the conditions under which antennas enter or leave active use by the sub-array. The Executor, via the Scheduling Block Observe Script, will control the activities of astronomical sub-arrays.

Multiple astronomical sub-arrays originating from different Programs, Projects or Proposals will be required if the New Mexico Array (EVLA Phase II) is constructed. Currently, the plan is to address this need by using multiple Schedulers, one for each sub-array (e.g., there could be one sub-array and Scheduler for the EVLA "Y" and another for the New Mexico Array antennas when independent observations are being done on the two sets of antennas). It may also be necessary to have multiple copies of DCAF as well.

Subsets of the antennas in the array may be withdrawn from availability to the automated scheduling and execution system. These will be administered and operated interactively without the use of a scheduler. All commands will be issued directly to the instrument from a user interface or via an Observe Script supplied via the user interface. The use of this type of administrative sub-array will generally be for testing and maintenance purposes, although some science activities (e.g. VLBI) may be more easily accommodated in this manner.

2.3.5 Scheduling

The Scheduling Sub-system periodically, or on command from a user's console, checks the Project Database for ready Program Blocks. It copies the Scheduling Blocks from each ready Program Block into a local pool. Upon request from the Executor, Scheduling Blocks are drawn one-by-one from the pool based on control logic at the Program and Scheduling Block levels, Program Block and individual Scheduling Block scheduling conditions and priorities, overall instrument scheduling policies, scheduling heuristics, and an internal scheduling algorithm.

Each Scheduling Block is marked as "scheduled" in the Project Database as it is pulled from the pool and sent to the Executor. When all its Scheduling Blocks have been marked "scheduled", the Program Block is marked as scheduled.

Administrative Program Blocks, which may be as succinct as a single Scheduling Block, will be downloadable to the Scheduling Sub-system on command from an interactive user's console.

2.3.6 Execution

The System Executor will from time-to-time (usually at or just prior to the completion of an executing Scheduling Block Observe Script, or on command from a user's console) request the Observe Script from another Scheduling Block in the scheduling pool. The Observe Script is used along with real-time inputs from other parts of the system (e.g., TelCal, and the Telescope Configuration Database) to produce commands to operate the instrument.

Upon submission of the first command by the Executor to the hardware control sub-systems, The Scheduling Block is marked as "executing". After the last command has completed it is marked as" executed".

Employing manual mode, an interactive user may take control of the instrument at the System Executor level. The Executor will be directed to accept Observe Script input from the user's console. Upon receipt of one or more lines of Observe Script, execution will proceed as if the command had originated from the automated system.

2.3.7 Online Data Analysis

The system shall be able to perform a variety of numerical processing tasks to support configuration, control, signal conditioning and data quality assurance objectives. This will include calibrations, atmospheric modeling, bad data recognition and flagging, RFI detection and excision, and basic imaging. Some of these functions will be embedded in one of the main Subsystems while others may operate as independent entities. Final determination of operational context for many of the Online Data Analysis components has not yet been made.

2.3.8 Output Capture and Storage

As the Scheduling Block passes from Scheduler to Executor and as commands are generated from the executing Observe Script, time tagged metadata will flow to DCAF. The arriving metadata will be associated and organized according to data type and time tag by DCAF into a time ordered stream. This stream is segmented on-the-fly by DCAF into Integrations, Sub-scans, Scans and Execution Blocks.

Astronomical data will flow directly from the CBE to the Science Archive without going through DCAF. Additional streams needed from time to time by functions like TelCal will be teed and directed from the appropriate source.

Each Execution Block corresponds to one complete execution of the entire Observe Script of a Scheduling Block. As each Execution Block is completed, the corresponding Scheduling Block is marked in the Project Database as having been executed one additional time. The Scheduling Subsystem will compare the updated execution count for the Scheduling Block against the block's iteration limit. If the limit has been reached, the Scheduling Block will be removed from further consideration by the Scheduler and marked as finished in the Project Database. Once

all Scheduling Blocks in a Program Block have been finished, the Program is marked as finished. Once all Programs in a Project have been finished, the Project is marked as finished.

2.3.9 Data Export

When a sufficiently complete block of metadata is available in DCAF, it will be ready for export to the Science Archive. Sufficiently complete means that one or more data reduction processes may be meaningfully applied to it. The point at which this will occur may vary with the nature of the observation and the purpose of various data reduction tasks and may happen at the Execution Block, Scan or Sub-scan level. The data will be output according to a prescribed format.

Data needed by the QLP will be exported directly to it by DCAF in a format appropriate to the needs of the QLP services being provided.

2.4 NRAO Science Data Model (SDM)

This model defines the content of science data products stored in the Science Archive or exported to the user community. All post-processing is defined in terms of the SDM. At the end of the Observing process data is produced which is conformant to the SDM. Data is stored in the Science Archive in this form, and is exported to the user in an Export Data Format, which is a mapping of the SDM to some external representation such as FITS or XML.

An Export Data Format will be specified that will be used to convert input received by DCAF into a form suitable for storage in the Science Archive and use by Data Reduction systems. Active efforts are underway to specify the format in conjunction with the needs of ALMA and GBT.

3 General Requirements

3.1 Software

3.1.1 Processing Applications

All math/science and other numerical application software shall take optimal advantage of all language, compiler, and system computational features and resources to reduce run times to the minimum practical level. They shall be coded in such a manner as to minimize the possibility of floating point exceptions during processing.

Non-EVLA built software shall be debuggable and preferably have source code available. In cases where it is possible to trap failures of non-EVLA software, the EVLA software shall do so. Detected failure and out of spec conditions shall be reported.

3.1.2 Management

All management software functions shall take optimal advantage of all language, compiler and system features and resources to reduce overheads to the minimum practical level.

3.1.3 I/O

All communications, and storage and retrieval operations shall take optimal advantage of all system resources to reduce overhead and latency to the minimal practical level.

3.1.4 General

Operating system, message passing and other middle-ware, and programming language(s) used shall follow industry standards and be commonly available and widely used. Availability of source code for the OS will be very important.

3.2 Reliability/Availability

3.2.1 Auto-correction

The System shall be self-monitoring. It will be capable of detecting, reporting on and automatically taking action to remedy or lessen the impact of, at a minimum, the following types of abnormal conditions: processor hardware failure, operating system hangs or crashes, software process hangs or failure, computational performance below minimum specifications, computational error rates above maximum specification, internal communications failures, and external communications disruptions.

3.2.2 Software

The software part of the system shall be self-righting to the maximum extent possible (see Section 6.4, Failure Detection and Recovery). A reliability goal is to be able to

perform without the need of a total system restart to clear an internal failure for a duration equal to the time between system maintenance windows.

3.2.3 Computational Hardware

Through the use of software detection of computational hardware failure and standby compute resources, the system shall be able to perform indefinitely without complete loss of compute services, except in the event of total failure of primary and backup power.

3.2.4 Network

The system shall be able to survive the loss of individual network connections or an entire network and shall be able to rapidly regain normal operations upon restoration of full network service.

3.3 Serviceability

3.3.1 Hardware Accessibility

All system processing and interconnect hardware shall be readily accessible for maintenance, repair, replacement and/or reconfiguration.

3.3.2 Software Accessibility

All systems and application source code shall be available to or on the systems that execute it.

3.3.3 Debugging

All software application modules shall be debuggable.

3.4 Maintainability

3.4.1 Software tools

Software tools and pre-built applications that do not have source code available shall come with a complete diagnostic package and customer support.

3.4.2 Operating Systems

Operating system software shall either have source code available or come with sufficient diagnostics and customer support.

3.5 Scalability

3.5.1 Hardware

I/O, communications, and processing hardware shall be easily expandable, reconfigurable, augmentable and replaceable to meet increasing data transfer and processing demands imposed by EVLA science, and availability of new technology.

3.5.2 Performance

The system shall be scalable to an extent limited only by hardware technology and budget constraints.

3.6 Security

The System needs a robust security mechanism in place so that unauthorized users are not allowed access to system internals. Authorized users are expected to be restricted to software and hardware development, testing, maintenance and operations personnel.

All users of the System must be uniquely identified. This could be done via a username and associated password scheme that would authenticate and authorize the user access to the system and, if applicable, grant the user access to restricted or controlled parts of the system. If a user cannot be identified, they will not be given access. In order to monitor all past access to the system, all attempts to access the system should be logged.

Users' needs and expectations from the system will be different. Systems operations might be given unrestricted access to all aspects of the system and have the authority to grant and revoke privileges on a per-user basis. Development, testing and maintenance personnel, on the other hand, may require access to some parts of the system, but not all, indicating that varying levels of access are needed that allow privileges to be granted on a per-user and what-do-you-need-to-do basis.

Note: Requirements derived from documented sources are marked with the document number and requirement number. The document number is the same as the listing number in the references. These should be consulted for clarifications and expansions of the brief requirement descriptions given here.

3.6.1 Requirements

3.6.1.1 All users of the system shall login using some form of unique identification (e.g., username and password) (7; 2.3-R1).

3.6.1.2 All login attempts shall be done in a secure manner (e.g., encrypted passwords) (7; 2.3-R3).

3.6.1.3 There shall be a system administrator with unrestricted access to all aspects of the system (7; 2.3-R4).

3.6.1.4 Each user shall have a set of system access properties that defines the user's privileges within the system. (e.g., the subsystems a user may control or system tools the user may access).

3.6.1.5 The administrator shall have the ability to create and add a new user to the system.

3.6.1.6 The administrator shall have the ability to remove a user from the system.

3.6.1.7 The administrator shall have the ability to edit a user's system access properties.

3.6.1.8 The administrator shall have the ability to block all access to the system for all users or selectively by user. (All blocked users with active sessions shall automatically be logged off.)

3.7 Installation and Upgrade

3.7.1 Transparency

Upgrades shall be accomplished in manner that is transparent to processing, communications and I/O software functions with the possible exception of recompilation of executables.

3.7.2 Seamlessness

Upgrades shall be accomplished in a manner that is seamless, in that it does not affect hardware modules or software functionality that it meets at interfaces.

3.7.3 Operations Activities

The system shall continue operations, although not necessarily at full capacity, on all unaffected computing resources during partial shutdowns for maintenance, repair and/or upgrade.

3.7.4 Test Mode

The system shall be able to handle non-real-time operations in a transparent fashion (i.e., as if real-time).

3.7.5 Replacability

Modular design principles shall be employed to the maximum extent possible. Maximal practical use of available "hot-swappable" devices and components shall be made.

3.8 Documentation

3.8.1 Hardware

Complete and comprehensible hardware systems specifications and configuration information shall be readily available.

3.8.2 Software Coding Practices

Software system and application code shall be well documented and written in a generally familiar language or languages (preferably not more than two). Software shall be written in a style that is easily read and using practices that allow for minimal confusion.

4 Subsystem Specifications and Requirements

Note: Requirements derived from documented sources are marked with the document number and requirement number. The document number is the same as the listing number in the references. These should be consulted for clarifications and expansions of the brief requirement descriptions given here.

4.1 Proposal Construction and Submission (PCS)

Scientific users will be able to build and submit a proposal for a set of observations in electronic form to an automated system. This system will assist the user in the determination of correct physical parameters, instrument configurations and hardware settings, plus scheduling information needed to satisfy the proposed scientific objectives.

A Proposal Construction and Submission System is currently under active development. It is meant to eventually serve the needs of all NRAO instruments including the EVLA. The first version is targeted for use by the GBT. A summary of the design (excerpted from a memo by Doug Tody) as it presently exists along with EVLA specific requirements is provided below.

4.1.1 Strategy (Proposal Construction and Submission)

4.1.1.1 General

- provide uniform proposal submission and handling for all NRAO telescopes, including ALMA when it comes online
- integrate interactive observation planning tools in Phase II
- changes which affect users should be infrequent and carefully planned
- avoid writing code now which is thrown away later
- coordinate development with ALMA where it makes sense

4.1.1.2 Phase I

- target is sometime in 2004
- supports VLA, VLBA, GBT
- scope includes anything which is required for a fully operational facility, e.g.,
  proposal preparation and submission
  online proposal database
  proposal handling tools

4.1.1.3 Phase II

- target is sometime around 2007
- add support for ALMA, EVLA
- add interactive observation planning capabilities
- integrate with Observation Preparation tools

- preserve proposal database and handling tools from Phase I

4.1.2 Summary of Major Components (Phase I)

4.1.2.1 Proposal Submission Tool (PST)

- for creating, editing, and browsing proposals
- can query proposal database and download proposals
- can save partial proposals to a workspace (local or server)
- can submit proposals when done

4.1.2.2 Proposal Database (PDB)

- stores proposals
- provides query facilities for searching proposals
- provides data access methods for retrieving, uploading proposals including entire proposals, or individual elements of a proposal
- provides user authentication and secure access

4.1.2.3 Proposal Handling Tools

Initial tools (more can be easily added as we use this):

- query for proposals matching some criteria
- print a proposal (PostScript or PDF)
- retrieve a proposal (XML)
- reports, conflict detection, annotation, e.g., for grading, referree comments
- associate proposal with telescope schedule or archive data
- edit/browse (using proposal tool)

4.1.2.4 NRAO User Database

- general facility not specific to proposal handling
- maintains profile for each NRAO user
- contact and other information
- used for global login authentication
- used to help fill in forms

4.1.3 Inputs

4.1.3.1 All items in the present VLA Observing Application.

4.1.3.2 Calibrator selections by the user.

4.1.3.3 Online processing function selection and parameter specification by the user.

4.1.3.4 Sources from external lists and catalogs.

4.1.4 Outputs

4.1.4.1 Proposals. Incomplete to local storage (on the computer running the Proposal Construction software), complete to the NRAO Proposal Database.

4.1.4.2 Custom source lists to private user storage.

4.1.5 Functions

4.1.5.1 Proposal Creation – fill out the Observing Application.

4.1.5.2 Online processing specification.

4.1.5.3 Calibrator selection.

4.1.5.4 Source List construction.

4.1.6 EVLA Requirements

4.1.6.1 Methods, including construction aids, for obtaining all necessary proposal information from the user shall be provided (4; 1-R1, see 1-R6 for data types).

4.1.6.2 An electronically based interface (UI) usable from remote locations shall be provided for proposal input submission and editing (4; 1-R2, 1-R3, 1-R5).

4.1.6.3 All proposal data shall be stored in digital form (4; 1-R2).

4.1.6.4 Proposals in progress shall be storable for subsequent access (4; 1-R3).

4.1.6.5 It shall be possible to control proposal access (e.g. via passwords) (4; 1-R4).

4.1.6.6 Access to and methods of incorporating administrative information into the proposal shall be provided (4; 1-R6).

4.1.6.7 Access to catalogs shall be available and a method shall exist to build and import source lists (4; 1-R7).

4.1.6.8 The input of scientific justification information shall be supported (4; 1-R8).

4.1.6.9 It shall be possible to specify allowable observing conditions (4; 1-R9).

4.1.6.10 A method of indicating Target-of-Opportunity Observations shall be provided (4; 1-R10).

4.1.6.11 Proposal contents shall be automatically checked and verified (this shall include integration time, calculated data rates, volumes and processing load

plus feasibility of using the correlator at the requested spectral resolution and total bandwidth) (4; 1-R11, 1-R12, 1-R13).

4.1.6.12 It shall be possible to generate observational configurations to aid in the checking and verification process (4; 1-R11.3).

4.1.6.13 Access to previous proposals shall be provided. All proposals shall be archived for future reference. (4; 1-R14).

4.1.6.14 It shall be possible to generate instrument control parameters to aid in the checking and verification process (4; 1-R15).

4.1.6.15 All proposal submission materials shall be printable (4; 1-R16).

4.2 Proposal Management

Proposal Management will deal with the needs of the proposal evaluation and rating process as well as time allocation and array configuration assignment. Use will be strictly by NRAO personnel.

4.2.1 Inputs

4.2.1.1 Proposals from the Proposal Database.

4.2.1.2 Referee comments and grades.

4.2.1.3 Proposal ratings.

4.2.1.4 Time allocations.

4.2.1.5 Array configuration assignments.

4.2.2 Outputs

4.2.2.1 Approved Projects with ratings, configuration assignment(s) and time allocation(s) to the Project Database.

4.2.2.2 Referee assignments.

4.2.3 Functions

4.2.3.1 Referee selection.

4.2.3.2 Referee grade normalization and Proposal rating.

4.2.3.3 Project creation – proposal configuration assignments and time allocations.

4.2.4 EVLA Requirements

4.2.4.1 All proposal data shall be stored in digital form (4; 1-R2).

4.2.4.2 It shall be possible to control proposal access (e.g. via passwords) (4; 1-R4).

4.2.4.3 Access to and methods of incorporating administrative information into the proposal shall be provided (4; 1-R6).

4.2.4.4 It shall be possible to generate observational configurations to aid in the checking and verification process (4; 1-R11.3).

4.2.4.5 Access to previous proposals shall be provided. All proposals shall be archived for future reference. (4; 1-R14).

4.2.4.6 It shall be possible to generate instrument control parameters to aid in the checking and verification process (4; 1-R15).

4.2.4.7 All proposal submission materials shall be printable (4; 1-R16).

4.2.4.8 An automated proposal management process (including referee selection and review) shall be available (4: 1-R17).

4.3 Program Preparation

A Program Preparation system is also planned. It will facilitate the automated subdivision of a Proposal into Projects, Programs and Sub-programs by the Principal Investigator or his designate (possibly with the aid of an NRAO staff expert). It will include the ability to apply logical constraints, looping constructs and iteration counts to Program Blocks and their constituent Scheduling Blocks.

4.3.1 Inputs

4.3.1.1 Approved Projects from the Project Database.

4.3.1.2 Hierarchical breakdown input on Programs, Sub-programs from the user.

4.3.1.3 Control logic supplied by the user that describes dependencies between and among the Scheduling Blocks that makeup a Program Block.

4.3.1.4 Administrative information from the user.

4.3.1.5 Scheduling information from the user.

4.3.1.6 Calibrator selections by the user.

4.3.1.7 Online processing function selection and parameter specification by the user.

4.3.1.8 Sources from external lists and catalogs.

4.3.1.9 Standard Program hierarchies with default parameters from templates stored in the Project Database.

4.3.2 Outputs

4.3.2.1 Projects with Program, Program Blocks and Scheduling Blocks specified to the Project Database.

4.3.2.2 Custom source lists to user private local storage.

4.3.3 Functions

4.3.3.1 Proposal retrieval.

4.3.3.2 Project creation – based on configuration assignments and time allocations.

4.3.3.3 Program creation – division of Projects into Programs based on time allocations, operational restrictions and scientific considerations including set-up of Program Blocks of Scheduling Blocks (capture and application of administrative, scheduling, control logic and hierarchical breakdown information).

4.3.3.4 Online processing specification.

4.3.3.5 Source specification.

4.3.3.6 Calibrator selection.

4.3.4 EVLA Requirements

4.3.4.1 An electronically based interface (UI) usable from remote locations shall be provided.

4.3.4.2 All Project and Program data shall be stored in digital form.

4.3.4.3 It shall be possible to control Project and Program access (e.g. via passwords).

4.3.4.4 Observation preparations in progress shall be storable for subsequent access (4; 2.1-R6).

4.3.4.5 Calibrator selection shall be supported (4; 2.1-R7.3, 2.1-R10).

4.3.4.6 Online data processing functions (e.g. real-time calibration and Quick Look Pipeline) requests shall be supported (4; 2.1-R7.4, 2.1-R11).

4.3.4.7 Source list construction shall be supported (4; 2.1-R8.2, 2.1-R8.3).

4.3.4.8 Observation contents shall be automatically checked and verified (4; 2.1-R8.5, 2.1-R8.11, 2.1-R13, 2.1-R14, 2.1-R15).

4.3.4.9 Standard observing modes and default parameters shall be provided to assist user achievement of science goals (4; 2.1-R8.7, 2.1-R8.8).

4.3.4.10 Analytical tools to support observation prediction shall be available (4; 2.1-R12).

4.4 Observation Preparation

For proposals that have successfully passed through the submission process and have been decomposed into Projects, Programs and Program Blocks, runtime details in the Scheduling Blocks must be specified by the Principal Investigator or his designate (possibly with the aid of an NRAO staff expert). The system shall provide defaults to the user and guide him or her through the process of preparing an observing script for operating the instrument. This functionality needs to be provided at an early stage in the EVLA Project in order to conduct tests on prototype hardware. A test version of the code already exists and will either evolve into or be replaced by a permanent version. This will likely occur before the rest of the Program Preparation software has been built. It will be absorbed into Program Preparation at some point. The makeup of the permanent version of the software is described below.

4.4.1 Inputs

4.4.1.1 Projects made-up of Programs with their constituent Program Blocks and Scheduling Blocks from the Project Database.

4.4.1.2 Standard Observe Scripts with default parameters from templates stored in the Project Database.

4.4.2 Outputs

4.4.2.1 Program Blocks with Observe Scripts in the Scheduling Blocks returned to the Project Database.

4.4.2.2 Finished Program Blocks returned to the Program Database and marked as ready for scheduling.

4.4.3 Functions

4.4.3.1 Observe Script generation.

4.4.3.2 Program Block approval for scheduling.

4.4.4 Requirements

4.4.4.1 A method for obtaining all necessary Observation Preparation information shall be provided. (Specific inputs shall be defined)

4.4.4.2 Scheduling Blocks containing information sufficient for Observation Scheduling shall be produced in the required format. (4; 2.1-R5.1)

4.4.4.3 Proposal submission input shall be converted into the parameters needed to schedule the observation (4; 2.1-R2).

4.4.4.4 An electronically based interface (UI) usable from remote locations shall be provided for Observation preparation and editing. (4; 2.1-R4.1).

4.4.4.5 Observation preparations in progress shall be storable for subsequent access (4; 2.1-R6).

4.4.4.6 Standard observing modes and default parameters shall be provided to assist user achievement of science goals (4; 2.1-R8.7, 2.1-R8.8).

4.5 Project Manager

The Scheduler will be will be unaware of observational entities of scope larger than a Program. The rest of the EVLA Software System (Executor, M&C, DCAF) will only be aware of Scheduling Blocks. There is, however, a need to track and report on the progress of observing activities at the Project level. This will be one of the responsibilities of the Project Manager Sub-system.

The Project Manager may also be the logical place to manage multiple astronomical sub-arrays for the Phase II New Mexico Array.

4.5.1 Inputs

4.5.1.1 Start of a Scheduling Block notification from the Executor.

4.5.1.2 End of a Scheduling Block notification from DCAF.

4.5.1.3 Project and Program information from the Project Database.

4.5.1.4 Project status requests from the Principal Investigator.

4.5.2 Outputs

4.5.2.1 Start of Project notification to the Principal Investigator.

4.5.2.2 End of Project notification to the Principal Investigator.

4.5.2.3 Project Status information to the Principal Investigator.

4.5.2.4 Updated Scheduling Block, Program, and Project status to the Project Database.

4.5.3 Functions

4.5.3.1 Track the execution status of all observing activities on a Project basis.

4.5.4 Requirements

4.5.4.1 The Principal Investigator shall be notified at the beginning and end of a Project.

4.5.4.2 Scheduling priority shall be based in part on the completion state of the Program and Project of which the Scheduling Block is a member.

4.6 Observation Scheduling

Scheduling Blocks shall be dispatched for execution based on a set of priorities, environmental inputs, scheduling heuristics, and an internal scheduling algorithm. The project may be made up of a number of separate programs with different configurations. These must be scheduled and run in the presence of programs from other projects. Each lowest level piece of schedulable work shall be verified against the current system state for feasibility of execution. Accepted work will be available to be dispatched for execution.

4.6.1 Inputs

4.6.1.1 Program Blocks from the Project Database.

4.6.1.2 Manually supplied Scheduling Blocks from an interactive user.

4.6.1.3 Weather conditions.

4.6.1.4 Equipment State from the Telescope System Configuration Database.

4.6.1.5 Scheduling heuristics.

4.6.1.6 Status requests.

4.6.1.7 State change requests (e.g., shutdown).

4.6.2 Outputs

4.6.2.1 Next Scheduling Block Observe Script to the Executor.

4.6.2.2 Metadata to DCAF.

4.6.2.3 Status requests to the Telescope System Configuration Database.

4.6.2.4 Status reports.

4.6.2.5 Alarms and alerts.

4.6.3 Functions

4.6.3.1 Program Block Harvesting – periodic inspection of the Project Database for ready Program Blocks and retrieval of them for scheduling of their component Scheduling Blocks.

4.6.3.2 Online, interactive receipt of Scheduling Blocks.

4.6.3.3 Individual Scheduling Block priorities and relationships plus equipment state, weather conditions, scheduling heuristics and an internal algorithm are used to determine the next Scheduling Block from the pool to be dispatched to the Executor. Control logic at the Program level is used to resolve and maintain inter-Scheduling Block relationships.

4.6.3.4 Observing Script Dispatch – transfer of the Observing Script portion of the next Scheduling Block. Selected items in the administrative and scheduling portions of the Scheduling Block are transferred for consumption by DCAF.

4.6.3.5 Error Handling – trap and report all internal errors, recover from faulty states and resume regular operations.

4.6.4 Requirements

4.6.4.1 Scheduling Blocks input from Observation Preparation shall be pooled and selected for execution.

4.6.4.2 An electronically based interface (UI) usable from remote locations shall be provided for Observation scheduling.

4.6.4.3 Scheduling shall be done without need for human intervention.

4.6.4.4 Manual submission of Scheduling Blocks to the pool shall be supported (4; 3.1-R3.1) (7; 6).

4.6.4.5 Manual management of schedules shall be provided (4; 3.1-R6, 3.1-R7, 7; 6).

4.6.4.6 Scheduling shall be dynamic (4; 3.1-R3.2, 3.2-R1).

4.6.4.7 Priority based scheduling shall be supported (4; 3.1-R5, 3.2).

4.6.4.8 Scheduling shall take atmospheric effects into account (4; 3.2-R5).

4.6.4.9 Offline operation to support testing and maintenance shall be supported (4; 3.1-R8).

4.6.4.10 Automated calibrator selection shall be supported (4; 2.1-R10.5, 3.1-R11, 3.1-R12, 3.1-R14).

4.6.4.11 Scheduled work shall be reported (4; 3.1-R15).

4.6.4.12 Work submitted for execution shall be reported (4; 3.1-R15).

4.6.4.13 Multiple observational sub-arrays shall be supported (3; 3.2.6) (4; 3.1-R4)

4.6.4.14 Multiple administrative sub-arrays shall be supported (3; 3.2.6).

4.6.4.15 The offline default archive image pipeline shall be triggered upon completion of an observation program (4; 5.3).

4.7 Observation Execution

Using an Observe Script, the system shall convert the script into a series of commands necessary to set-up instrument hardware and software in order to execute the scheduled work.

4.7.1 Data Types

4.7.1.1 Observe Script – the portion of the Scheduling Block that describes how the instrument hardware and software is to be set-up and controlled to satisfy the objectives of the Scheduling Block's creator.

4.7.1.2 Command – the atomic unit of control information dispatched to the real-time control systems. It contains all parameters and settings needed by the EVLA (or VLA during transition operations) control systems and is time stamped to provide accurate coordination of all real-time activities.

4.7.1.3 Execution Metadata – all execution information required by DCAF to assemble output results at the Integration, Sub-scan, Scan and Execution Block level.

4.7.1.4 Equipment State – all current real-time system hardware and software system status and state information needed during script interpretation and command formation.

4.7.2 Inputs

4.7.2.1 Observe Scripts from Scheduler.

4.7.2.2 Manually supplied Observe Script or control commands from an interactive user.

4.7.2.3 Online data analysis results from TelCal.

4.7.2.4 Equipment State from AMCS and CMCS (via the Telescope System Configuration Database).

4.7.2.5 System parameters from the database.

4.7.2.6 Status requests.

4.7.2.7 State change requests (e.g., shutdown).

4.7.3 Outputs

4.7.3.1 Executable commands to AMCS and CMCS.

4.7.3.2 Metadata to DCAF.

4.7.3.3 Status reports.

4.7.3.4 Alarms and alerts.

4.7.4 Functions

4.7.4.1 Observe Script Interpretation – receives (from the Scheduler or directly from an online user) and interprets the Observe Script. Resolves intra-Scheduling Block control logic and forms control commands. Consumes equipment state information and online data analysis results and finalizes all parameter values and control settings.

4.7.4.2 Command Dispatch – transfers control commands for consumption by the AMCS and CMCS.

4.7.4.3 Error Handling – trap and report all internal errors, recover from faulty states and resume regular operations.

4.7.5 Requirements

4.7.5.1 Real-time parameter conversions shall be done (e.g., right ascension and declination to azimuth and altitude).

4.7.5.2 Execution state shall be reported.

4.7.5.3 Instrument control systems shall be capable of handling both existing VLA antennas and modified EVLA antennas. (Operations shall continue with the existing correlator until the WIDAR correlator is installed and functional) (3; 3.1.1).

4.7.5.4 Instrument control systems shall be capable of operating single antennas or the entire array as a VLBI antenna (3; page 5).

4.7.5.5 Interactive control (including manual interrupt) of the instrument shall be supported (3; 3.1.3) (4; 3.1-R3.1, R3.3) (5; 2-R2) (7; 3.4.4-R2).

4.7.5.6 Bad data flagging shall be done automatically (4; 5.1-R5) (6; 10-R2).

4.7.5.7 An electronically based interface (UI) usable from remote locations shall be provided for Observation execution. (5; 2-R5, 2-R6).

4.7.5.8 All instrument hardware settings shall be readable by authorized users. (5; 2-R7, 7; 3.3).

4.7.5.9 It shall be possible for authorized users to override instrument hardware settings (5; 2-R1, 7; 3.4.3, 3.4.4).

4.8 EVLA Monitor and Control System (M&C)

The M&C system represents the physical EVLA instrument. It is divided into two primary components, the Antenna Monitor and Control System (AMCS) and the Correlator Monitor and Control System (CMCS).

It is desirable that the EVLA be able to be operated by non-expert users while still affording expert control for special operation and maintenance. This implies that the EVLA M&C should make available, to the user, every control point and monitor point in the system but should not make their access mandatory for normal operation.

The EVLA M&C must support operation with a variety of antenna types and possibly different correlator types.

4.8.1 Inputs

4.8.1.1 Execution commands from the Executor.

4.8.1.2 Execution commands from the User Interface (manual mode).

4.8.1.3 Requests for monitor point output (initialization, termination, change) from User Interface.

4.8.1.4 Requests for equipment state from Scheduler and Executor.

4.8.1.5 Requests for descriptive parameters (e.g., board id numbers) from the User Interface.

4.8.1.6 Status requests.

4.8.1.7 M&C software state change requests (e.g., shutdown).

4.8.2 Outputs

4.8.2.1 Monitor data from hardware and software monitor points per request to User Interface and Monitor Data Archive.

4.8.2.2 Raw data products in the form of records of spectra with attached headers from the Correlator Backend to the Science Data Archive.

4.8.2.3 Raw data products in the form of records of spectra with attached headers from the Correlator Backend to TelCal.

4.8.2.4 Raw data products in the form of records of spectra with attached headers from the Correlator Backend to QLP.

4.8.2.5 Raw data products addressing information from the Correlator Backend to DCAF.

4.8.2.6 Error and warning reports to the alarm and alert system.

4.8.2.7 Equipment state (functional state and availability of all components) to Scheduler, Executor and DCAF.

4.8.2.8 System descriptive parameters to the User Interface and the Parameter Database.

4.8.2.9 M&C System status.

4.8.3 Functions

4.8.3.1 Monitor and control EVLA (and VLA during transition operations) antenna hardware systems (AMCS).

4.8.3.2 Monitor and control WIDAR Correlator hardware and software systems including the Correlator Backend (CMCS).

4.8.3.3 Error handling.

4.8.4 General Requirements

4.8.4.1 An electronically based interface (UI) usable from remote locations shall be provided for manual operation of the system (5; 1.1) (7; 1.4, 1.5, 1.7, 3.4).

4.8.4.2 All users shall be authenticated before access is permitted (7; 1.2).

4.8.4.3 Monitor data shall be retrievable in near real time (5; 1.1).

4.8.4.4 User directed storage of monitor output shall be supported (5; 1.1.1-R6).

4.8.4.5 User selection of monitor point output rate/interval shall be supported (5; 1.1.1.1, 1.2.1.1).

4.8.4.6 Warning and error messages shall be issued upon occurrence (5; 1.2.1-R6.1) (7; 3.2).

4.8.4.7 System status and auxiliary data shall be reported on a regular basis (5; 1.2.1.1-R1).

4.8.4.8 It shall be possible to perform analyses on monitor data (5; 1.3).

4.8.4.9 User controlled display and plotting of monitor data shall be provided (5; 1.4) (7; 3.5).

4.8.4.10 It shall be possible to manually set control points within system modules (5; 2-R1) (7; 3.4.3-R1).

4.8.4.11 It shall be possible to determine the state of all systems components via their monitor points (5; 2-R2) (7; 3.3-R2, 3.3-R3).

4.8.4.12 User selection of control points shall be supported (5; 2-R3).

4.8.4.13 It shall be possible to manage system access and privileges (5; 2-R7) (7; 1.3).

4.8.4.14 User direction of monitor output to external systems shall be supported (5; 2-R9) (7; 1.6).

4.8.4.15 User logging and report writing facilities shall be available (7; 4.1, 4.2, 4.5, 4.7),

4.8.4.16 A user notes and reminder facility shall be provided (7; 4.3).

4.8.4.17 It shall be possible to schedule monitoring tasks for later execution (7; 4.4).

4.8.4.18 Access to the maintenance database shall be provided (7; 4.6).

4.8.4.19 It shall be possible to manage system descriptive parameters (7; 5).

4.8.5 Real-time Requirements

4.8.5.1 All commands sent to a MIB shall begin execution within 100 microseconds of the time tag associated with the command.

4.8.5.2 All commands shall be time-tagged. The MIBs contain a queue that can hold up to 50 commands. Time-tagging is required to maintain the queued commands in a time-ordered sequence.

4.8.5.3 Antenna positions shall be updated every 50 milliseconds in order to maintain a sub-arcsecond level of pointing accuracy.

4.8.5.4 Frequency changes within a band shall be completed within one second.

4.8.5.5 A 'nodding' source switch rate of once per ten seconds shall be met.

4.8.5.6 Delay polynomials at the rate of 10 per second shall be supplied to the Correlator to meet the requirements of on-the-fly mosaicing.

4.8.6 AMCS

The AMCS is that part of the EVLA Monitor and Control system that represents antennas and ancillary equipment such as the weather station and the Master LO's in the Control Building. There is no single 'AMCS' program; instead the AMCS is a collection of components layered hierarchically with control and monitoring done recursively from general at high layers to specific at low layers.

The AMCS will support two antenna types at the beginning (EVLA & VLA) and may support at least two more (VLBA & NMA) in the future. Section 8 on Transition Plans presents time lines, milestones and major issues concerning moving from VLA to EVLA in a continuous manner that will not seriously disrupt science operations.

4.8.6.1 VLA Antenna Monitor and Control

VLA antennas that have yet to be upgraded will be monitored and controlled in the AMCS through the Interim Control and Monitor Processor (CMP). The CMP communicates to VLA antenna Datasets (hardware interface modules) via the Serial Line Controller (SLC) and waveguide similar to current Modcomp control software.

To the EVLA, the CMP presents the VLA antennas as AMCS Antenna objects. Access to Datasets, while not for operation, will be maintained for maintenance purposes throughout the interim period.

4.8.6.2 EVLA Antenna Monitor and Control

EVLA antennas will also appear as AMCS Antenna objects from their 'front' but will interface directly with the upgraded antenna hardware through the Module Interface Board (MIB). There will be approximately 30 MIBs in each EVLA antenna. Each MIB will be accessible via a fiber optic Ethernet connection and directly controllable for special use and maintenance purposes.

4.8.7 CMCS

The CMCS is currently undergoing joint, active design and development with the WIDAR Correlator development group at the Dominion Radio Astronomy

Observatory in Penticton, British Columbia. The CMCS is responsible for correlator configuration and real-time status monitoring. It is part of the overall EVLA monitor and control (M&C) system, but provides a level of abstraction to the rest of the EVLA system via the Virtual Correlator Interface (VCI). The VCI allows the EVLA M&C system to control and monitor the correlator without having to know about the intimate details of the correlator.

Documents providing details on the internals of CMCS can be found at the following web address:

http://www.drao-ofr.hia-iha.nrc-cnrc.gc.ca/science/widar/private/Software.html

4.9 Online Data Analysis

The system shall be able to perform a variety of numerical processing tasks to support configuration, control, signal conditioning and data quality assurance objectives.

Some of these processes will produce real-time or periodic results that must be delivered to specific system components. The primary system components for these operations are TelCal, Calc and the Atmospheric Phase Modeler, which will be discussed below.

The system shall also provide a means of producing images or spectra on short time scales. This is the role of the Quick Look Pipeline (QLP), also considered to be part of online data processing.

Some additional processing tasks will be done in real-time by the Correlator Backend (CBE). These are described in external documents referenced in the resources section and will not be covered further here.

Data analysis requirements in addition to those contained in the next three sections are provided below.

4.9.1 Requirements

4.9.1.1 Bad data recognition and automatic flagging shall be provided (3; 3.2.10) (4; 5.1-R5).

4.9.1.2 RFI detection and excision shall be available (3; 3.2.12) (4; 5.1-R5).

4.9.1.3 Atmospheric modeling shall be available (4; 5.4-R1).

4.10 Telescope Calibration (TelCal)

4.10.1 Complex Gain

4.10.1.1 Inputs – correlator results for each integration from the CBE, calibrator flag (indicating a calibrator is being observed) from DCAF, calibrator information, flagging information and other metadata from DCAF.

4.10.1.2 Output – phases for all antennas in the sub-array of interest to Executor during phased array operations, complex gains to DCAF.

4.10.1.3 Function – during phased array operation when observing a calibrator, for each subband of each baseband complex gain is calculated (AntSol) for each antenna and the phase component is delivered to the Correlator. For non-phased array operations the same data is sent to DCAF for retention with the rest of the science data to provide a history of gain results for use by the pointing algorithm and potential use during offline data reduction.

4.10.2 Pointing Offsets

4.10.2.1 Input – complex gains for a calibrator pointing scan, pointing positions, metadata from DCAF.

4.10.2.2 Output – pointing offsets to Executor.

4.10.2.3 Function – for each polarization for each antenna, averaged over all subbands and basebands, calculate the local pointing offset to be folded into the pointing model.

4.10.3 Focus Offsets

4.10.3.1 Input – complex gains for a sub-reflector scan, sub-reflector positions, metadata from DCAF.

4.10.3.2 Output – sub-reflector corrections to the Executor.

4.10.3.3 Function – for each antenna, averaged over all subbands, basebands and polarizations, calculate the correction to be folded in with sub-reflector positioning information.

4.10.4 Requirements

4.10.4.1 Auto-phasing determination with feedback to configuration and control shall be provided (3; 3.2.9) (4; 5.1-R2).

4.10.4.2 Pointing offset determination shall be provided (3; 3.2.9) (4; 5.1-R7.3).

4.10.4.3 The generation of antenna-based complex gain solutions shall be provided (3; 3.2.10) (4; 5.1-R3).

4.10.4.4 Phase stability data shall be available (4; 5.1-R4).

4.11 Calc

4.11.1 Inputs

4.11.1.1 Antenna locations.

4.11.1.2 Source coordinates.

4.11.1.3 Time.

4.11.1.4 IERS parameters.

4.11.1.5 Weather information.

4.11.2 Output

4.11.2.1 Delay Model coefficients or polynomial.

4.11.2.2 Pointing angles for each antenna.

4.11.3 Function

4.11.3.1 Calculate the wave delay for each antenna.

4.11.4 Requirements

4.11.4.1 A per antenna wave delay shall be generated on a periodic basis (4; 5.1-R7.2).

4.12 Atmospheric Modeling

4.12.1 Input

4.12.1.1 Site atmospheric parameters from the site weather instruments.

4.12.1.2 Atmospheric emission in the observed EVLA bands.

4.12.1.3 Water Vapor Radiometer (WVR) data.

4.12.1.4 Measured atmospheric profiles (as available) of temperature and water content from atmospheric sounders.

4.12.1.5 Measured optical depths from tipper meters and/or FTS as available.

4.12.1.6 Measured electron content from GPS or other techniques.

4.12.2 Output

4.12.2.1 Conversion factors between WVR data and water contribution to astronomical phase sent to the Data Reduction Pipeline (QLP).

4.12.2.2 Opacities in all astronomical bands sent to the Data Reduction Pipeline (QLP).

4.12.3 Function

4.12.3.1 An Atmospheric Model will be generated periodically to be used by the Observation Scheduler and QLP. It will predict absorption, emission and path length along the line of sight through the atmosphere at all EVLA bands.

4.12.4 Requirements

4.12.4.1 Atmospheric Model (4; 5.4).

4.13 Quick-Look Pipeline (QLP)

The Quick-Look Pipeline produces images and/or spectra on short time scales for feedback (to both the operator and the observer) during observing. It will most likely be a streamlined version of the full offline science pipeline.

4.13.1 Inputs

4.13.1.1 Activation/deactivation signals.

4.13.1.2 Metadata from DCAF.

4.13.1.3 Raw data from the CBE.

4.13.2 Outputs

4.13.2.1 Images.

4.13.2.2 Spectra.

4.13.3 Functions

4.13.3.1 Flagging.

4.13.3.2 Calibrations

4.13.3.3 Dirty image generation.

4.13.3.4 Dirty image deconvolution.

4.13.4 Requirements

4.13.4.1 Images and spectra in near real-time shall be available (4; 5.2).

4.13.4.2 Access via a GUI shall be supported (4; 5.5).

4.13.4.3 Users shall have control over the display of near real-time results (4; 5.5-R2.9).

4.13.4.4 Access to near real-time output displays shall be supported (4; 5.5-R1, 5.5-R6).

4.14 Data Capture and Format (DCAF)

4.14.1 Inputs

4.14.1.1 Metadata from Scheduler

4.14.1.2 Metadata from Executor

4.14.1.3 Addressing information pointing to the storage location of output Correlator Integrations with accompanying headers from the CBE.

4.14.1.4 Weather data from M&C.

4.14.1.5 Equipment State from M&C.

4.14.1.6 Status requests.

4.14.1.7 State change requests (e.g., shutdown).

4.14.2 Outputs

4.14.2.1 Science Model in Export Data Format to the Archive.

4.14.2.2 Science Model in Export Data Format to QLP.

4.14.2.3 Metadata to TelCal.

4.14.2.4 Status reports.

4.14.2.5 Alarms and alerts.

4.14.3 Functions

4.14.3.1 Construct science data output suitable for consumption by the EVLA Offline data reduction system.

4.14.3.2 Supply TelCal with required inputs.

4.14.3.3 Error handling – trap and report all internal errors, recover from faulty states and resume regular operations.

4.14.4 Requirements

4.14.4.1 DCAF shall output science data according to a standard data format definition.

4.14.4.2 DCAF shall organize received data according to a hierarchy of: Sub-scans, Scans, Execution Blocks as defined in the NRAO Project Model.

4.14.4.3 All inputs supplying information for the science data model shall be time stamped to assure proper coordination with associated information from other sources.

4.15 Offline Data Reduction

Offline data reduction package(s) shall be available for the processing of EVLA scientific data produced by the system. EVLA System output shall be producible in a form or forms compatible with the requirements of external offline data reduction packages and the Virtual Observatory. Offline Data Reduction is not a formal part of the EVLA System.

4.15.1 Requirements

4.15.1.1 An offline data processing package that fulfills the needs of EVLA data reduction shall be provided (6; 1.1, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 3.4, 4, 5, 6, 7, 8, 9).

4.15.1.2 Antenna location determination shall be provided (4; 5.1-R7.1).

4.15.1.3 Global pointing model determination shall be supported (4; 5.1-R7.2).

4.15.1.4 Complex band pass determination shall be supported (4; 5.1-R7.4).

4.15.1.5 Focus analysis shall be supported (4; 5.1-R7.5) (3; 3.2.9).

4.15.1.6 Delays determination shall be supported (4; 5.1-R7.6).

4.15.1.7 Polarization calibration terms determination shall supported (4: 5.1-R7.7).

4.15.1.8 Flux density scale determination shall be supported (4: 5.1-R7.8).

4.15.1.9 TIP analysis shall be supported (4; 5.1-R7.9) (3; 3.2.9).

4.15.1.10 Antenna aperture efficiency determination shall be supported (4; 5.1-R7.10).

4.15.1.11 Atmospheric modeling shall be available (4; 5.4).

5 Communications Infrastructure

The EVLA online system will be made-up of a set of inter-communicating distributed software processes deployed across a number of separate physical processors. Careful selection of the communications methods employed to service the connections among system components at a variety of levels will be key to successful operations. It should be recognized that use of a single methodology throughout the system for all communications may not be possible. However, a single solution would be highly desirable from the standpoint of minimizing implementation complexity and future software maintenance and upgrade.

Final Selection of communication solutions will not be made until significant design and prototyping has been done for components at the subsystem level. An overall system prototyping effort will be undertaken in parallel with subsystem design and prototyping. The objective of this would be to analyze and test candidate communications methodologies in an environment that closely models the system that will ultimately be deployed. This should allow informed decisions to be made on communications technologies.

5.1 Physical Layer Description

Ethernet has been selected as the physical communication layer for all areas of the EVLA. TCP/IP and UDP/IP will be used either directly, or indirectly through middleware. Processors within the control building will be connected with 1 Gbit full duplex multimode fiber. Each antenna is connected to the control building with 1 Gbit full duplex single mode fiber pairs. Within an antenna, components are connected with 100 Mbit multimode fiber. The nodes in each antenna, along with antenna specific control building equipment, will form a class C network with an IP address of the form 10.80.1xx.yyy. Where xx is the antenna identifier and yyy identifies a MIB within that logical antenna. yyy will have the same meaning (connect to the corresponding MIB) in all antennas. The IP address of each component in an antenna will be derivable at run time using a slot ID mechanism.

5.2 General Design Considerations

EVLA has some unique issues that must be carefully considered while designing the communications infrastructure. These important factors shape the design not only of the communications system but also of EVLA subsystem components.

5.2.1 Real Time Control

IP over Ethernet is not conducive to real-time control. Timely delivery of commands cannot be guaranteed over this type of interface.

5.2.2 Resource Limited Processors

There will be approximately 900 small, resource limited devices (MIBs) connected to the EVLA LAN; commercial middle-ware packages will be limited to those that can physically fit on them.

5.2.3 Traffic Congestion

Because of the large number of devices on the LAN the potential exists for traffic congestion when considering archive, monitor and control data flows.

5.2.4 Loose Coupling Means Less Communications

A pattern that has emerged during the design process is to reduce the amount of communication between components by making the components more functionally self-contained. In computer science terms, module *cohesion* (functional 'completeness') is increased resulting in a decrease in *coupling* (interaction) with other modules. Autonomous components can be made to control themselves and need to be told only what to do and when to do it. This helps resolve the problem of real-time control using non real-time communications since control information can be sent ahead of time and execute on time completely within the component.

5.2.5 Hierarchical Layering Means More 'Meaningful' Communications

Hierarchy is another pattern that was identified during design. In a hierarchical system data is more generally defined at higher levels and more specific at lower levels reducing the amount of data flow in the overall system. A source to be observed will be specified very generally (by its name) at the highest (proposal) levels but, by the time it reaches a MIB, it will have been transformed into very specific servo position values. Refinement through layers eliminates the need for higher-level processes (say those concerned with science) to know low-level details of how the instrument works in order to operate it.

Monitor data in a hierarchical system is naturally self-filtering since details are maintained at appropriate levels. Power supply voltages are not needed at the Observation Scheduler level. Information of a general nature ('is it functional on this band?') is all that needs to be communicated when determining if a particular antenna can be used for the next observation. In a similar fashion, alerts will be maintained at appropriate levels so that users and processes only see those of interest to them.

Hierarchical data flow reduces the number of communications channels that must be maintained by any one component of the system. Instead of Observation Executor maintaining 900 connections to the MIBs, it will only communicate with 27 antennas that will, in turn, each communicate with its 30 or so MIBs.

5.2.6 Simplicity

The EVLA LAN is remarkably similar to the Internet – a network of many processes providing individual services, although the dependencies among the processes is fixed for the EVLA rather than changeable and time varying as the Internet. The EVLA communications system can benefit from lessons learned and readily available tools that have evolved from the Internet.

The fact that all components of the system will have their own IP address means that we can take advantage of the Domain Name Services (DNS) already in place in

NRAO rather than use more complex registry facilities associated with managed communications systems.

'Lightweight' clients such as the Generic Device Browser and standard web browsers will allow greater and more flexible access to the various system components. Client connections and operation should be identical amongst the various components to allow simpler client development – a client should not have to know what type of device it is communicating with in order to communicate with it.

5.3 Inter-process Communications

In general, control flows down and monitor data flows up through the system hierarchy. Higher level processes control and monitor lower level processes. The controlling process sends control information directly to, and specifies the monitor information to be retrieved from, the controlled process.

The following issues must be considered during sub-system level design.

5.3.1 Topology

Determine the overall connective structure of the system. Identify all communications paths between and among system software components. Be especially cognizant of paths between sub-system components that span one or more system levels.

With the exception of the monitor data archive, a process should not be programmed to send monitor data to any particular place. Instead, monitor data should be made available to anyone who asks for it at runtime.

5.3.2 Style

Determine the communications style that best suits each identified path. These may be peer-to-peer, client-server, broadcast, publish-subscribe or other.

If it is necessary to control a process by making function calls over the network then the design of the process should be questioned. Required use of remote function calling indicates that a process is not autonomous.

5.3.3 Date Volumes and Rates

Examine all provider-consumer relationships and identify message sizes and frequency, data volumes over time, supply capabilities and delivery requirements. Specify required delivery rates needed to achieve target operational needs for real-time processing.

There will be areas in the system of high rates and volumes of data. Areas such as Correlator Backend output should be considered special. Support by the overall EVLA communications system may not be prudent.

There may also be areas of high rate/volume traffic that are a result of poor subsystem design. Where large amounts of data are being passed between two processes,

consideration should be given to determining if data processing functionality has been misplaced.

5.4 Monitor Data Delivery

Monitoring is done on both hardware and software processes; monitor data can be from a physical monitor point in hardware or derived in software. All monitor data in the system will be archived periodically. The monitor data archive is the only place where monitor data is sent without request – there will be no 'screens broadcast' or other similar types of blind monitor data transmission.

During operation, the controlling process *requests* monitor data from the controlled process. The communications infrastructure must provide the means for one process to both subscribe to, and poll for, the monitor data of another.

The synchronous nature of ELVA operation means that data volumes may spike at certain times. If testing deems it necessary, staggered archive times for periodic archiving should be considered. So, for example, those monitor points to be archived hourly will not all be transmitted at the same time in the hour.

5.5 Alert Reporting

Alerts are used to communicate exceptions to 'normal' operation. The efficient propagation of alarm and alert messages throughout the system from originator to ultimate consumer must be addressed in the context of the distributed nature of system functionality. Hierarchical approaches to raising and managing alarm and alert messages as they move upward through the system structure must be well designed in order to avoid overwhelming consumer processes and display consoles with unnecessary traffic.

EVLA alerts will be reported in two ways. Every alert will be archived and alerts may be 'subscribed' to from one process to another. The latter implies that alerts in the EVLA will be used not simply to notify the operator but may also be used by components in the system to 'automatically' adjust operation or flag data where feasible.

5.5.1 Alert Types

Four types of alerts are used in the EVLA:

Failure – No longer functional; a 5V power supply reads 0V.

Warning – Functional but outside of normal limits; the power supply reads 4.5V.

Error – Associated more with human or software errors such as asking for the value of a non-existent monitor point; a divide-by-zero exception in software.

Information – A client might subscribe to the weather station's relative humidity monitor point and ask to be notified if it reaches a specified value.

5.5.2 Information Available for Each Alert

Information about each alert will include *Time, Device ID, Type, Description, Consequences* (impact to the system or operation), *Cause* (to the level known), and the suggested *Action* that the operator should take. Some of this information will not necessarily be sent with the alert, but may be linked to an offline retrieval mechanism such as a web page or database.

5.5.3 Alert 'Noise' Reduction

Important information can sometimes be 'buried in the noise' if too many alerts are reported. EVLA addresses this problem as follows:

5.5.3.1 Reduce alert 'flicker' - alerting will employ methods to reduce repetitive alerting such as when a borderline value passes rapidly in and out of tolerance.

5.5.3.2 Send only those alerts asked for - during normal operation, a client (for example Observation Executor) will subscribe to the alerts from the server to which he is connected (in this example the one serving antenna objects). This allows the client to receive only those alerts that are of interest to his operation.

5.5.3.3 Receive meaningful alerts - alert subscription allows hierarchical reporting which will reduce the number of meaningless alerts. Observation Scheduler, for example, simply has to know if an antenna is available or not, it does not need to see alerts about a failed power supply in one of the MIBs.

5.5.4 An Alert Reporting 'Use-Case' Example

The following diagram in Figure 8 (read from the bottom) shows how alerts move up the system hierarchy. The power supply failure itself causes an alert along with the four Front End receivers that went down as a consequence of the power supply failure. The antenna reports the loss of the 4 bands and the Executor makes decisions about the observation based on the loss of the band on which he might have been observing. The Observation Scheduler makes scheduling decisions to either not use the antenna at all, or to schedule a program block that uses a different band. Simply being alerted that P302-2 failed, would require that the Observation Scheduler know the internal workings of antennas or else the alert would be meaningless.

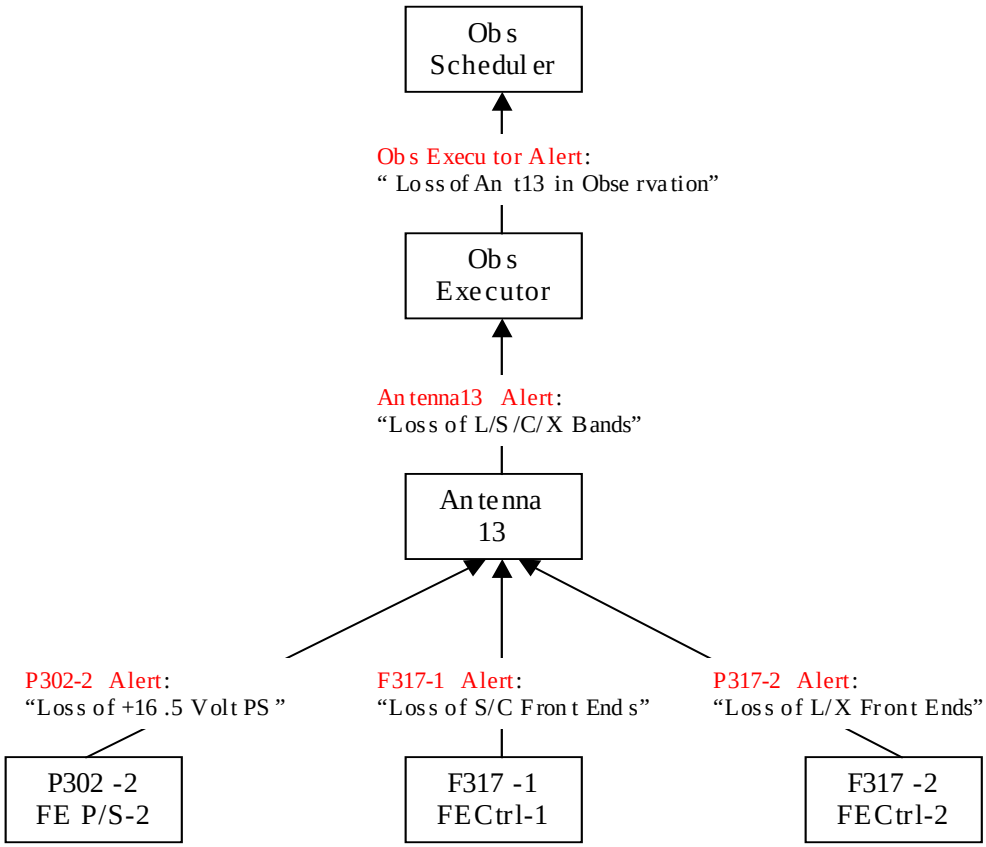


Figure 8. Diagram of an alert reporting use case.

5.6 Signals and Triggers

Signals and triggers represent a class of communications tied to particular events. A couple such situations in the EVLA System have already been identified; more will undoubtedly be discover during sub-system design.

Some standard signals (such as KILL which will be used during system shutdown and in failure and recovery procedures) will have to be propagated over the network to the intended recipient.

A trigger event will have to be delivered to the QLP at certain points during observation. This will most likely originate from the DCAF.

5.7 Time Synchronization

Time synchronization of components in the critical path(s) of the real-time system will be vital. Control systems will have to provide time stamped commands to various parts of the hardware for precisely coordinated initiation of functions. This will be accomplished in the antenna systems by time-deferred set commands that will contain a future absolute time at which a command is to become effective. Heartbeat interrupts and heartbeat interval times must be supplied via the communications system to the hardware MIBs in order to accomplish this.

Correlator data frames set to the Correlator Backend must be tagged with the absolute time reference for the correlated antenna signals, and Meta-data delivered to DCAF will also have to include the absolute time stamp of the associated command so output data can be properly organized.

6 System-wide Services

6.1 User Interface

A number of users will have access to various parts of the EVLA system; principal investigators, administrative staff, array operators, engineers, technicians, software developers, scientists and other authorized users. The method by which most of these users will interact with the system is through graphical user interface (GUI) applications. There are two primary types of GUI applications; Web-based applications (WebApps) and standalone applications. WebApps are accessed via a Web browser and are typically referred to as thin clients. The standalone applications will usually require executables to be installed on the user's computer and are often referred to as thick or fat clients. Each of these application types has advantages and disadvantages. The main differences between the two are ease of deployment and richness of the graphics. WebApps are by nature very simple to deploy and are geared more toward static form-based applications with little dynamic content. Standalone applications typically have more features and provide an environment that is more dynamic with richer graphics. The EVLA system will utilize both types of applications to handle tasks such as proposal management, scheduling, operator screens that allow monitor and control capabilities of the array, correlator monitor and control, and archive access tools.

Several applications that access the EVLA system are currently under development. The Proposal Submission Tool and the Monitor Data Archive Tool are being developed by the e2e project. The Proposal Submission Tool is responsible for proposal entry and submission. The Monitor Data Archive Tool provides the ability to query and display monitor data in either list format or as a plot.

Other applications under development include the Device Browser and the EVLA ™LabVIEW Server, both of which are written entirely in Java and are standalone applications. The Device Browser is a general purpose tool that enables users to connect to EVLA devices and view information about a particular hardware device. It also provides the ability to control the device. The EVLA LabVIEW Server is a bridge between LabVIEW and the EVLA monitor and control system. Its primary function is to provide a mechanism for monitor and control of EVLA devices from LabVIEW screens.

Due to its high portability and extensive libraries, Java will be the programming language used to develop most of the GUI-based applications. Although WebApps will not use Java for the GUI front-end, they will likely rely on Java Servlets in the middle tier of their applications.

System requirements identify the EVLA subsystems that require human interaction via GUI-based applications. General descriptions of input and output data flows for each subsystem have been provided in this high level design. However, details on these data

™ LabVIEW is a trademark of National Instruments Corporation.

flows necessary for complete GUI design will not be available until sub-system design work has been done.

6.2 Archiving

All data products generated by the fixed parts of the system (portable elements such as test equipment and engineering laptops are specifically excluded) shall be permanently stored in a manner that allows easy, flexible and quick retrieval of desired data. This includes, but is not limited to, spectra, lags, derived observational data products, auxiliary data, applied corrections, flagging information, configuration and observational parameters, monitor values, and error reports. The archive, as used here, may be comprised of one or more non co-located Databases on physically separate machines.

EVLA archiving will follow the NRAO E2E Archive Model.

6.2.1 Databases and Data Sources

6.2.1.1 Submitted proposals from PCS to the Proposal Database.

6.2.1.2 Proposal updates and content additions from PCS to the Proposal Database.

6.2.1.3 Projects of Program Blocks and their Scheduling Blocks from Observation Preparation to the Project Database.

6.2.1.4 Template, default, and test Scheduling Blocks from Observation Preparation or an interactive user to the Project Database.

6.2.1.5 Monitor data from M&C to the Monitor Database.

6.2.1.6 DCAF output in Export Data Format to the Science Archive.

6.2.1.7 Raw data products in the form of records of lags or spectra with attached headers from the Correlator Backend to the Science Archive.

6.2.1.8 Equipment state from M&C to the Telescope System Configuration Database

6.2.1.9 System Properties to the Site Properties Database.

6.2.1.10 TelCal output to the Calibration Database.

6.2.1.11 Weather Data from M&C to the Monitor Database.

6.2.1.12 Equipment test and maintenance information to the Maintenance History Database.

6.2.2 Requirements

Note: Requirements derived from documented sources are marked with the document number and requirement number. The document number is the same as the listing number in the references. These should be consulted for clarifications and expansions of the brief requirement descriptions given here.

6.2.2.1 Data and measurements shall be collected in real-time (3; 3.2.10).

6.2.2.2 The system shall store all observation parameters (4; 4.1-R1)

6.2.2.3 The system shall store all astronomical data obtained from EVLA antennas (4; 4.1-R1.1, 4.2-R1).

6.2.2.4 The system shall store all environmental data (4; 4.1-R1.2, 4.3) (5; 1.2.1-R1.1, 1.2.1.2) (7; 3.4.1).

6.2.2.5 The system shall store all monitor and control data (4; 4.1-R1.3, 4.4) (5; 1.1.1, 1.2.1-R1.2, 1.2.1.3) (7; 3.3-R1).

6.2.2.6 The system shall store all project data (4; 4.1-R1.4, 4.6).

6.2.2.7 The system shall store all derived data products (4; 4.1-R1.5, 4.5).

6.2.2.8 Stored data and information shall be protected from loss (4; 4.1-R3) (6; 3.6-R3).

6.2.2.9 The system shall store all maintenance data (4; 4.6-R5) (7; 4.6).

6.2.2.10 The stored data and information shall be searchable (4; 4.7) (5; 1.2.1-R3, 1.2.2) 6; 3.6-R1) (7: 4.6).

6.2.2.11 All stored data and information shall be retrievable (4; 4.8) (5; 1.2.1-R3, 1.1.2, 1.2.3) (6; 3.6-R1).

6.2.2.12 The archive shall be designed to meet the requirements for Virtual Observatories (4; 4.9).

6.2.2.13 Proprietary data protections shall be provided (4; 4.10).

6.2.2.14 All instrument configuration parameters shall be stored (5; 1.2.1.3-R2).

6.2.2.15 Import and export of stored data and information to and from offline processing system(s) shall be supported (6; 3.1, 3.2, 3.5).

6.2.2.16 Storage of data from a scheduled tasks or scripts (7; 4.4).

6.2.2.17 Report generation shall be provided (7; 4.7).

6.2.2.18 The system shall store all system descriptive parameters (7; 5).

6.2.3 Science Data Archive

The Science Data Archive is implemented as a two-tier system: queries are supported by a catalog of observational meta-data, stored in a relational database. Visibility data is stored as a file hierarchy on a file system served by a large file server. Querying the catalog generally results in a list of pointers to the appropriate data sets (that is, file names on the archive file server).

So far, the science data archive query patterns suggest a tabular, relational data model. Therefore, queries to the archive catalog are SQL queries supported by a relational database.

This may be contrasted to other archive designs, in which XML is the primary (or only) access pattern for the database: other archives may be implemented with "pure" XML databases.

Data Formats:

In general, users do not want to re-format the data, wishing to use the format that is native to the instrument for their observation... Currently, data is retrieved in the "native" format or each instrument. So for example, VLA data is stored and retrieved in the "VLA Export" format, GBT data is FITS-GBT, and VLBA data is CFITS.

Other formats may be desired; such re-formatting is currently delegated to the end-user. (For example, AIPS++ can read the formats currently supplied by the archive, and it may re-format the data into an AIPS++ measurement set.)

The archive has hooks for export re-formatting; data format publishers may be "plugged in" to the archive if other data formats are desired.

6.2.4 Monitor Data Archive

The "on-line" monitor data archive is implemented as a Java service that accepts monitor data points via an XML stream over an Ethernet IP connection. The Java service parses the XML and inserts the data into a table in a relational database, basically a "flat" table with time-stamped name/value pairs:

ANTENNA NAME

DEVICE NAME

MONITOR POINT NAME

TIMESTAMP

STATUS

MONITOR POINT VALUE

One refinement is that a new table is created at local midnight: tables cover a 24-hour period of time. This implementation detail is hidden from the rest of the system via a set of views which "stitch together" any input/output that spans a midnight boundary. Limiting tables to a 24-hour period significantly simplifies data management issues such as backup/restore, and also improves query performance.

XML and Monitor Data:

The current prototype monitor-data archive is implemented as an XML front-end to a relational database, which makes it easier to index the data points and to support structured queries (e.g., "All monitor data points for a particular device in a particular time-range").

Monitor Database query results are generally returned via XML InfoSets (VOTables) to a plotting tool (VOPlot).

Since this database is "XML-in, XML-out", it would be worthwhile to consider a "native" XML database implementation. But the current implementation works very well.

6.3 Start-up and Shutdown

6.3.1 Design Objectives

6.3.1.1 Start any major online system software component (Scheduler, Executor, M&C, DCAF, QLP, TelCal) on any processor or set of processors (for systems requiring significant compute resources) in the system compute complex.

6.3.1.2 Start all real-time system components (Executor, M&C, and DCAF) at the same time with a single command.

6.3.1.3 Completely shutdown all real-time system components at the same time with a single command.

6.3.1.4 Independently start any major system software component.

6.3.1.5 Independently shutdown any major system software component.

6.3.1.6 Include a clean-up function with any software shutdown that checks for and removes all hung, orphaned or otherwise leftover processes attached to the process or processes begin shutdown.

6.3.1.7 Include a general clean-up function that can be used to completely remove all hung, orphaned or otherwise leftover processes and return the system compute complex to a pristine condition.

6.4 Failure Detection and Recovery

6.4.1 Design Objectives

6.4.1.1 Incomplete Communication – the partial or total loss of a communication, whether through a system fault or deliberate cancellation should not permanently affect the operation of either party to the communication.

6.4.1.2 Floating Point Exceptions – all floating point exceptions should be trapped and, after corrective measures have been taken, return the process experiencing the exception to normal operation.

6.4.1.3 Not a Number (NaN) – generation of NaN values of any variety should be recognized and trapped immediately so they do not disrupt processing operations or get passed on to other processes.

6.4.1.4 Process Failure – Detect stopped, unresponsive, missing or otherwise non-performing processes and automatically restart them. A restarted process must be able to regain contact with all other processes with which it communicates and quickly return to a normal operating state. Healthy processes must not be disrupted by the failure of a process with which it communicates. At worst it may have to wait for the failed process to return to full operation and complete or cancel and resend a pending communication.

6.4.1.5 Processor Failure – The failure of a physical processor must be detectable from within the software system. All processes that were running on a processor that has failed must be quickly restarted on another processor and be able to resume normal operations. Healthy processors and the processes running on them must remain unaffected except for the possible need to wait for a pending communication to complete or be cancelled and resent once the failed system has been replaced or restored.

6.4.1.6 Network Failure – The failure of some or all of the paths in the communications network should not affect the function of the processors and the processes running on them. At worst, they may have to wait for a pending communication to be completed or cancelled.

6.4.1.7 Power Failure – Upon the restoration of power following a total outage to the online processing computer complex, the software systems should be able to automatically restart and, after verifying the health of all systems, come back to normal operations.

6.4.1.8 Data Loss – failure of any part of the offline or online systems should not result in the loss of any data. It should be possible to resume processing at a point where complete data integrity can be assured. For online processing, this may required other parts of the system to backup to a previous point once the failed component has been restored (e.g., the Scheduler may have to resend a Scheduling Block to a restored Executor). It is not expected that this will be possible for the real-time parts of the system. Quick resumption of services will minimize data loss, but failsafe data protection in real-time is not required.

7 Systems Engineering

7.1 Software

7.1.1 Programming Languages

To date Java and C have been used to implement early parts of the system.

7.1.2 Revision Management

CVS is currently being used for code management. Maven has been employed for Java builds and will be evaluated for other languages.

8 Transition Plans

8.1 Array

8.1.1 Operational Objectives

8.1.2 Science Objectives

8.1.3 Schedule

8.1.3.1 31-May-2004: Routine test observing using antenna 13.

8.1.3.2 Q4 2004: Routine inclusion of EVLA Antennas with VLA observing (Control of the EVLA antennas using EVLA hosted software).

8.1.3.3 Q1 2005: Routine control of VLA Antennas by EVLA hosted software.

8.1.3.4 Late Q1 2005: WIDAR Correlator PDR (29-Mar-2005 through 30-Mar-2005).

8.1.3.5 Q2 2005: Control of new VLA Correlator Controller by Modcomp hosted software.

8.1.3.6 Q3 2005: Control of new VLA Correlator Controller by EVLA hosted software.

8.1.3.7 Q4 2005: Specification of EVLA Science Data Archive format.

8.1.3.8 Q4 2005: Science data archiving via EVLA hosted software, VLA archive format.

8.1.3.9 End of Q4 2005: Modcomps retired.

8.1.3.10 Q1-Q2 2006: Assembly, configuration, hardware tests and on-sky tests of EVLA Correlator prototype at the VLA site (06-Mar-2006 thru 02-Jun-2006). (Note: 20 days have been allocated for on-the-sky tests.)

8.1.3.11 Late 2006: WIDAR Correlator CDR (19-Jun-2006 thru 20-Jun-2006).

8.1.3.12 Q2 2006: Data Capture and Format (DCAF) software ready for testing.

8.1.3.13 Q2 2006: TelCal software (new real-time calibrator analysis software) ready for testing.

8.1.3.14 Q3-Q4 2007: Installation, testing, and acceptance of EVLA Correlator subset.

8.1.3.15 Q4 2007: Science data archiving via EVLA hosted software, EVLA archive format.

8.1.3.16 Late Q4 2007: Start of shared-risk observing with WIDAR Correlator subset.

8.1.3.17 Late Q4 2007 thru Q2 2008: Installation, testing, and acceptance of full WIDAR Correlator.

8.1.3.18 Q2 2008 thru end of Q4 2008: Commissioning and turnover of full WIDAR Correlator.

8.1.3.19 End of Q4 2008: Earliest possible date to retire the VLA Correlator.

8.1.3.20 Q3 2010: Last VLA antenna retrofitted to the EVLA design (27-Aug-2010). End of Hybrid Array Operations.

8.1.3.21 Late Q2 2012: Last EVLA receiver installed (01-Jun-2012).

9 Appendix I - EVLA Architecture & Design Team Specification of Work/Charge to the Team.

9.1 Team Members

Tom Morgan
Kevin Ryan
Ken Sowinski
Boyd Waters

9.2 Key Dates

1. Date of team formation: 12/01/2003.
2. Charge to the team (original): 12/02/2003.
3. Charge to the team (latest version): 12/08/2003.
4. Anticipated lifespan of team: 4 to 6 months.

9.3 Overview

The basic responsibility of this team is to develop an overall architecture and design for the EVLA software, including both the e2e and M&C elements. The desired level of detail includes:

1. Review and certification or modification of the basic components of the EVLA software, i.e., Proposal Tool, Observing Tool, Scheduling Tool, Observing Layer, etc.

2. The assignment of functionality to each of the components of the EVLA software.

3. A reasonably detailed description of the dataflow among the components of the EVLA software. If possible, it would be desirable to specify the dataflow down to nearly the implementation level, but this goal must be reconciled with the lifespan of the team.

4. Specification of the interfaces for the components of the EVLA.

5. Identification of the basic objects for each component of the EVLA software.

6. Specification of the software infrastructure, especially the manner by which distributed objects will communicate with one another.

7. A path from the current VLA to deployment of the final overall architecture and design. In other words, a specification of the stages of development and deployment for control of the hybrid array that will take us from one EVLA antenna, 27 VLA antennas, and the current correlator to 28 EVLA antennas and a new correlator.

The level of detail achieved by this architecture and design effort must take into account the 4 to 6 month lifespan of the team. The maximum lifespan of 6 months is a hard deadline, and must not be exceeded. Four to five months would be preferable, but under no circumstances can the team extend its efforts beyond the 6 months time limit. If choices must be made, emphasis should be placed on that part of the e2e/M&C system that ingests the proposal tool output and ultimately delivers a data product suitable for post-processing. The EVLA architecture and design must be extensible, allowing incorporation of EVLA Phase II capabilities such as NMA antennas and VLBI correlation, and the incorporation of ancillary instrumentation such as an RFI monitor, the Atmospheric Phase Interferometer, and water vapor radiometers on the antennas.

9.4 Constraints

1. The use of the module interface board (MIB) must be accepted as a constraint upon the design. More specifically, the use of the MIB hardware and the MIB systems software must be accepted. MIB applications software is open to modification.
2. The current and evolving form of the test antenna software need NOT be taken as a constraint upon the overall architecture and design.
3. The architecture and design team must familiarize itself with the work of the current software teams and related efforts. That work need not constrain the design should new directions seem preferable, but the design effort should not ignore or remain ignorant of the work that has been done. The current software teams and related efforts are, as of 11/2003:

   a. the MIB software team and the related effort to develop a generic device browser.

   b. the control script language team

   c. the monitor data archive team

   d. the effort that has been made to adapt the CALC program to the needs of the EVLA

   e. the ongoing development effort working toward a visibility data archive

   f. the work that has been done on the Proposal tool

   g. the work that has been done on the Scheduling tool

9.5 Resources

9.5.1 ALMA

Some of the work that has been and is being done for the ALMA project may prove to be a valuable resource for the architecture and design team. In particular, the ALMA use case documents, the ALMA architecture documents, and ALMA development work on the path downstream from the conduct of the actual observation - correlator, correlator backend, the ALMA archive, and ALMA pipeline processing may be useful to the EVLA architecture and design effort.

9.5.2 Documents

1. The VLA Expansion Project: Construction Project Book, P. Napier and R. Perley, at:

http://www.aoc.nrao.edu/evla/pbook.shtml

2. System Requirements Specification EVLA Software System, Revision 1.0 (no project document number yet assigned), Tom Morgan, August 22, 2003. e2e project book, August 8, 2002, available at:

http://www.nrao.edu/e2e/documents/e2eprojectbook.pdf

3. EVLA Memo No 15, Scientific Requirements for the EVLA Real-Time System, J. Benson & F. Owen, editors, 9/26/2000, available on the EVLA Memos web page:

http://www.aoc.nrao.edu/evla/memolist.shtml

4. EVLA-SW-001, EVLA e2e Science Software Requirements, Revision 1.6, 2003-Apr-15, B. Butler and S. Myers, editors, available on the Computing Working Documents web page:

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

5. EVLA-SW-004, EVLA Engineering Software Requirements, Revision 1.4, 2003-Aug-06, S. Durand, available on the Computing Working Documents web page:

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

6. EVLA-SW-002, EVLA Data Post-Processing Software Requirements, Revision 1.2.1, 2003-Jul-03, S. Myers, available on the Computing Working Documents web page:

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

7. EVLA-SW-003, EVLA Array Operations Software Requirements, Revision 2.5, 2003-Jun-26, R. Moeser, P. Perley, available on the Computing Working Documents web page: http://www.aoc.nrao.edu/evla/techdocs/computer/workdocs/index.shtml

8. Antenna Monitor and Control System (AMCS) Antenna Monitor & Control System Design Philosophy, Kevin Ryan, May 6, 2003, available on the Computing Working Documents web page:

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

9. Antenna Monitor & Control Subsystem, Preliminary Requirements Specification, A24051N0001, Revision 2.0, April 4, 2002, Kevin Ryan, available on the Computing Working Documents web page:

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

10. Correlator Monitor and Control System (CMCS) NRC-EVLA Memo #15, EVLA Correlator Monitor and Control System, Test Software, and Backend Software Requirements and Design Concepts, Brent Carlson, Jan 23,2002, available on the Computing Working Documents web page:

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

11. System Requirements Specification, EVLA Correlator Monitor & Control, Revision 1.0, Bruce Rowen, December 5, 2002, available on the Computing Working Documents web page:

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

12. Protocol Specification, EVLA Correlator Monitor and Control, Virtual Correlator Interface - VCI, RFS Document A25201N0000, Draft 1.1 (or latest available), Sonja Vrcic, available at:

http://www.drao-ofr.hia-iha.nrc-cnrc.gc.ca/science/widar/private/Software

13. WIDAR Correlator Requirements and Functional Specification, The EVLA Correlator Software, Master Correlator Control Computer - MCCC, RFS Document A25202N0001, Revision 2.0, (or latest available), Sonja Vrcic, available at:

http://www.drao-ofr.hia-iha.nrc-cnrc.gc.ca/science/widar/private/Software

14. Correlator Backend (CBE) System Requirements Specification, EVLA Correlator Backend, A25251N0000, Revision 1.0, Tom Morgan, September 23, 2003, available on the Computing Working Documents web page:

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

15. Operational Interface, Software Requirements Specification, A24101N0001, April 4, 2002, Rich Moeser, available on the Computing Working Documents web page:

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

9.5.3 Technical

For UML design, there is a laptop computer, known as UMLtool, that currently resides in Bill Sahr's office (AOC Rm 348) that has been loaded with 4 different UML design tools: Rational Rose Enterprise Edition, Poseidon Professional Edition (Ver 1.6.1), Together Solo, and Visio Professional.

9.6 Milestones & Schedule

The team must meet with the team manager (and, perhaps, selected team consultants) within the first two weeks of the team's existence to discuss the charge to the team, and the organization of the team's efforts. This discussion should include or result in a plan or outline of the initial steps to be taken toward fulfillment of the charge. This discussion should also cover the topic of the fourth team member, and how to integrate that team member into the effort. A written report giving the state of the architecture and design, to date, shall be submitted at the end of the first two month's of the team's existence. Subsequent milestone reports, meetings, discussions will be agreed upon after the review of that report.

9.7 Deliverable

The core deliverable is a document describing the EVLA overall architecture and design.

10 Appendix II - Abbreviations and Acronyms

1. AMCS – EVLA Antenna Monitor and Control System
2. CBE – WIDAR Correlator Backend subsystem
3. CMCS – WIDAR Correlator Monitor and Control System
4. CMIB – WIDAR Correlator hardware Module Interface Board
5. CMP – Control and Monitor Processor. Processing system used to control mixed VLA and EVLA antennas during transition operations.
6. DCAF – Data Capture and Format subsystem
7. M&C – Monitor and Control system
8. MIB – (EVLA Antenna) hardware Module Interface Board
9. PCS – Proposal Construction and Submission System
10. QLP – Quick Look Pipeline
11. SB –Scheduling Block
12. TAC – Time Allocation Committee
13. TelCal – Telescope Calibration subsystem
14. WIDAR – Wideband Interferometric Digital Architecture, the Correlator design for EVLA
15. WVR – Water Vapor Radiometer

11 Appendix III - Definition of Terms

The Science Software Requirements (SSR – Resource Document 4) introduced its glossary with these words. “In this Section we introduce the main entities used to manage the whole observing process, which are constantly referred to in the requirements. These entities have some kind of hierarchical structure that will be further refined at the analysis stage of software development.” This list of terms, which is clearly based on SSR terminology, is the first step of the refinement process, an EVLA software glossary of terms which are important at the highest level. The purpose of this list is to provide the design group with a vocabulary to enable consistent discussions. The present version is limited to terms relevant to our initial efforts. It will be updated as meanings alter or new terms are needed. We prefer to think of it as a logical description rather than a dictionary. The glossary is written only with the EVLA in mind. It should sometime be read with the future NMA and VLBA operations in mind to see what changes are needed. Bits and pieces of various existing documents have been used to make a first pass at reconciling the disparate definitions. Terms in the definitions which themselves appear in the glossary are written in **bold face**. The term “observation” has been used to mean too many things in this discussion. It is suggest that it be left free as a generic word whose meaning is obvious (one hopes) in context. Usually it will mean something like “session”. There has been much confusion over the term used to describe the components of a “scan”. While we concur with the usage of the SSR which calls this a “record”, we will follow the recent ALMA change and call this a subscan. The terms defined here are intended to be consistent with the components of what the e2e effort is calling the ’project model’ and ’observing model’ and have been informed by the e2e discussions.

11.1 Project and Observing (Input Side)

Proposal: A request to NRAO for telescope time, including scientific justification, target sources, telescope setup, etc. A proposal is reviewed by referees and if approved is promoted to one or more projects.

Project: One or more projects are derived from an approved proposal.

Program: Within each project, observations in different configurations are each assigned to a separate program. Each program has associated with it an observing description (a program block), which describes (in greater detail than is available in the proposal) how the observations are to be completed for the program.

Program Block (PB): The set of scheduling blocks, derived from a single program, that are to be submitted to the Scheduling Tool for queuing at the same time. An observer would normally create these in a single run of the Observing Tool.

Scheduling Block (SB): The minimum scheduling unit. If a scheduling block cannot be completed in its entirety, it must be rescheduled. It may not be resumed in the middle. The scheduling block consists of an observe script and information for the scheduling tool to schedule it on the array in an appropriate fashion.

Observe Script: A script in the language of the real-time computer system directing the EVLA or a subarray thereof to collect data for the duration of a scheduling block. The EVLA is operated by having the Scheduling Tool, human operator, or observer pass an observe script to the real-time computer system. See subarray for more about observe scripts and subarrays.

Subarray: A collection of one or more antennas which are producing data which makes sense to correlate. We distinguish two kinds of subarrays: administrative and astronomical. Administrative subarrays are defined by the operations staff and apportion collections of antennas to independent observers. Astronomical subarrays are defined by the observer from the collection of antennas that were assigned. It is expected that subarrays can be independently configured with no arbitrary restrictions.

11.2 Data Capture, Archive and Reduction (Output Side)

Execution Block: A successful execution of the observe script associated with a scheduling block produces an execution block. There will be one for each execution of a given scheduling block.

Observing Session: The time contiguous execution of one or more **scheduling blocks** associated with a single **program**.

Observation: This term has been used both in the sense of **subscan** and **observing session**. We should not give any special meaning to this word and understand that when used its meaning is to be derived from context.

Scan: The scan is the lowest level output data object normally used by an observer. It is a sequence of one or more **subscans** that share a single goal. A scan is usually a single **subscan** of a target source or a calibrator, but, for instance, pointing and focus scans involve a pattern of **subscans**. This is usually the minimum unit seen by the observing tool.

Subscan: A Subscan is the minimal amount of data taking that can be commanded within an **observe script**. One or more **integrations** comprise a **subscan**. An example of a **subscan** is a single pointing within a holography **scan**.

Integration: An **Integration** is the basic unit of **astronomical data** presented to the **archive**. It is the average of a set of **correlator dumps**.

Correlator Dump: A Correlator Dump corresponds to the minimum available integration time output from the correlator hardware. One or more of these are averaged to form an **integration**. The minimum collection of data subject to software manipulation.

Pipeline: A sequence of data reduction operations performed according to a script present at the initiation of Pipeline operation.

Astronomy Data: Whatever is in the telescope's standard raw output data set. For the EVLA it is correlations, various calibration data (sys/cal, etc) flags, odds and ends, and **meta-data**.

Ancillary Data: Data other than **astronomy data** relevant to the time range of interest, such as operator logs, monitor data, open work orders and known software problems, slowly changing instrumental parameters (e.g. VLA baselines and pointing model parameters), ionospheric data extracted from the GPS network, etc.

Metadata: A description of a telescope setup under which **astronomy data** was obtained, containing, but not limited to, the relevant time range, source parameters, receiver setup parameters (especially LO frequencies, and the switch settings used to select bands and filters), the observing proposal in aid of which the observation is made, the observing procedures followed.

Archive: A permanent collection of all EVLA data and parameters available for access and retrieval. To be distinguished from the physical medium or media where the data resides.

Measurement Set: AIPS++ format for **astronomy data** from a telescope. *This is a candidate for the format of data communicated between the Correlator Backend and the* **archive**.

11.3 System Related Terms

Real-time System: The portion of the EVLA software that understands an **observe script** and actually controls the EVLA at the hardware level.

On-line System: The portion of the EVLA software which directly interfaces to the **real-time** system software, namely the Scheduling Tool, the Archive Tool, the Real-time Calibrator Analysis Tool, the quick-look Pipeline Tool, the Astronomer's What's Up Screen, and any other required software (for instance, that required to do the interfacing).