Fermi National Accelerator Laboratory

FERMILAB-Pub-93/147

Development of the Sloan Digital Sky Survey Online Systems

B. MacKinnon, E. Berman, P. Constanta-Fanourakis, S.M. Kent, T. McKay, T. Nicinski, D. Petravick, C. Pluquet, R. Pordes, R. Rechenmacher, G. Sergey and N. Wilcer

Fermi National Accelerator Laboratory
P.O. Box 500, Batavia, Illinois 60510

June 1993

Submitted to *IEEE Transactions on Nuclear Science*

Operated by Universities Research Association Inc. under Contract No. DE-AC02-76CHO3000 with the United States Department of Energy

Disclaimer

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DEVELOPMENT OF THE SLOAN DIGITAL SKY SURVEY ONLINE SYSTEMS

Bryan MacKinnon, Eileen Berman, Penelope Constanta-Fanourakis, Tom Nicinski, Donald Petravick, Catherine Pluquet, Ruth Pordes, Ron Rechenmacher, Gary Sergey, Neal Wilcer
Online Support Department

Steven Kent, Timothy McKay
Experimental Astrophysics Group

Computing Division
Fermi National Accelerator Laboratory
Batavia, IL 60510, USA

Abstract

The Sloan Digital Sky Survey (SDSS)[1] is an astrophysics project to produce a three dimensional map of a quarter of the Universe. We present a description of the SDSS prototype, the Drift Scan Camera (DSC) online system,[2] which is operational and currently being deployed. We also present the status of the development of the SDSS online system. The hardware and software architectures for both systems is provided

I. INTRODUCTION

The SDSS is a collaborative effort between Fermi National Accelerator Laboratory (Fermilab), the University of Chicago, Princeton University, the Institute of Advanced Study (at Princeton), Johns Hopkins University, and the Japanese Promotion Group. Its main results will be a photometric imaging survey and a red shift spectroscopic survey of galaxies and color selected quasars across a quarter of the sky. Some 10^8 galaxies and 10^6 quasars will be mapped producing approximately 1.2×10^{13} bytes of data over its five year running period (1995-2000). The result will be a three dimensional map of the Universe. It will cover a volume that is many times larger than the structures predicted by current theories of structure formation or observed in existing redshift surveys [1]. It, along with the DSC, will be deployed at the Apache Point Observatory (APO) in New Mexico. The site is in the Sacramento Mountains at the elevation of 2800 meters. The DSC and SDSS requirements bear significant similarities to high energy physics experiment support as well as some requirements unique to astrophysics.

I.A Fermilab's Role

The SDSS application marks a departure for us at Fermilab. It is our first venture into experimental astrophysics. We bring to the project our High Energy Physics (HEP) experimental experience and the large data sets acquired at high rates that is required. Although the application is different, there are enough similarities between experimental astrophysics and HEP that we are sharing many of the technologies.

II. TECHNOLOGIES EMPLOYED

The SDSS and the DSC systems integrate high speed Charged Coupled Device (CCD) readout over fiber optics into embedded VME single board computers with backend UNIX[3] systems. In doing so, we incorporate VMEbus, fiber optics, Motorola MVME167 CPU modules, Wind River Systems' VxWorks operating system, and Silicon Graphics computers. They are also heavily dependent on Fermilab developed hardware and software as well as hardware and software already in use at APO.

II.A CCDs

CCDs are electronic devices that digitize photon generated charge. They are preferred in optical astronomy over photographic plates due to their superior sensitivity, linearity, and dynamic range. They consist of a two dimensional array of light sensitive pixels. For the SDSS and the DSC, we are employing Tektronix TK2048D CCD imagers.[4] Each CCD is a 4.9 cm by 4.9 cm array of 2048 by 2048 pixels (4.19×10^6 pixels total). Each pixel datum represents 16 bits of digitized light intensity. The image from a telescope is focused onto one or more CCDs (the number depending on the system).

[1] This work was partially supported by the Alfred P. Sloan Foundation.

[2] Sponsored by DOE Contract number DE-AC02-76CH03000.

[3] UNIX is a registered trademark of UNIX Systems Laboratory.

[4] Tektronix is a registered trademark of Tektronix, Inc.

For comparison, the human eye is sensitive to a resolution of approximately 50 pixels per cm (at a 25 cm distance) compared to 417 pixels per centimeter for these CCDs. High quality photographic film can have an equivalent resolution from 500 to over 3000 pixels per centimeter.

III. DSC

In addition to being a conceptual and technological proving ground for the SDSS online systems, the DSC has science objectives independent of the SDSS and will operate independently. Unlike the SDSS, it is not a dedicated system, but will be employed for multiple purposes as determined by the observers (i.e., astronomers).

The DSC is a single CCD imaging camera system to be mounted upon a 3.5 meter optical telescope;[5] the DSC CCD has an effective 4.8 arcminutes field of view (the angular size of the moon is approximately 30 arcminutes). The system acquires and digitizes astronomical image data at approximately one megabyte per second from the CCD, displays it in real-time, writes it to disk, archives it to 8mm tape, and makes it available in near real-time to the scientist observer on a UNIX Host. An important aspect of the DSC is that control over the data acquisition and monitoring is provided to the observer via a public domain "C and Lisp like" interpreter. Data acquisition and control are VME based with no critical dependence upon the Host.

III.A DSC Goals

The DSC system has a number of attributes, some of which are unique to astronomy, others not.

III.A.1 Astronomy related attributes

1. Acquires the data in one of two modes: *drift scanning* and *staring*. Drift scanning is where the sky moves across the telescope. Staring is where the telescope stares at a specific point in the sky
2. Acquires the data at rates of up to twice *sidereal rate*. Sidereal rate is defined is the speed at which the earth rotates. A telescope acquiring at sidereal rate will be in a fixed position relative to the earth.
3. An emphasis was placed upon reliability. Since useful operational time is not predictable (e.g., it may be cloudy), reliability is very important. An observer's allocated time on the telescope may be only several days per year so down time due to system failure could mean crucial time lost. This is underscored since the system is deployed high upon a mountain top (at APO) 1800 kilometers from the development staff (at Fermilab).
4. Provides remote observation via Ethernet and the Internet.
5. Provides near real-time analysis of the data upon the Host.
6. Data is organized into a complete readout of the CCD, called *frames.* Depending on the CCD readout dimensions, a frame can be up to eight megabytes in size.

III.A.2 Other attributes

1. Provides a user extensible command language.
2. As the data is acquired, it is displayed in real-time upon a monitor to provide real-time feedback.
3. Frames are pooled on disks local to the Front End system. Storage for at least two hours of data is provided.
4. Can archive pooled frames to 8mm tape and provide it to the Host.
5. Front End independence from the Host. If the Host becomes unavailable, the Front End will continue to function. This implies that all items necessary for the functioning of the data acquisition must be under control of the Front End.
6. Data is acquired from the CCD at rates of up to one megabyte per second.

III.B Data Flow Architecture

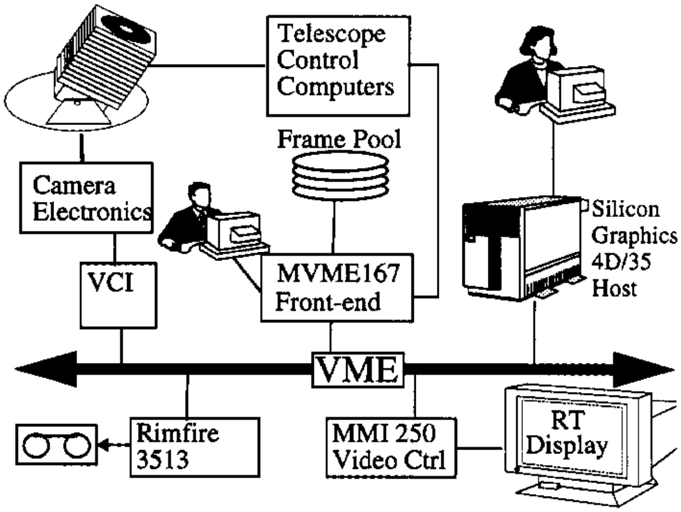


Figure 1: DSC System Block Diagram

Figure 1 shows the major components of the DSC data acquisition system. The data flows through the system as follows:

1. Image pixel data focused by the telescope upon the CCD is acquired by the Camera Electronics and transmitted via fiber optics to the VME Camera Interface (VCI).
2. The VCI buffers the data into segments that correspond to rows on the CCD.
3. The Front End MVME167 reads out the VCI and periodically writes out the data to the frame pool; simulta-

[5] The 3.5 meter telescope belongs to the Astrophysical Research Consortium (ARC).

neously, the Front End displays the image on the real-time Display.

4. Once a whole CCD's worth of data has been acquired, the Front End completes the frame by writing out relevant information in the frame's header (e.g., time of observation, name of the frame, observer's comments, etc.).
5. Under observer control, any frame that has been written to pool can be either written to 8mm tape, sent to the Host for viewing and analysis, or simply deleted.

III.C Design and Implementation

We executed an object oriented design for the DSC. We divided the system into major objects and defined the messages that pass between them. Figure 2 shows the object diagram produced for the DSC. The data flow can be traced through the system from one object to the next. For example, in the diagram, node 1 is the interface to the camera electronics, node 2 assembles the data into blocks, node 4 is the real-time display, and node 6 writes the data blocks to disk. During design, this was extremely helpful in modularizing the system.

Since no object oriented languages were readily available for VxWorks at the time, we had to develop in ANSI C. During implementation the object oriented modularization began to break down as a result. Nonetheless, the basic design was sufficient and the methodology proved as adequate as previous ones we employed (e.g., Structured Analysis and Structured Design).

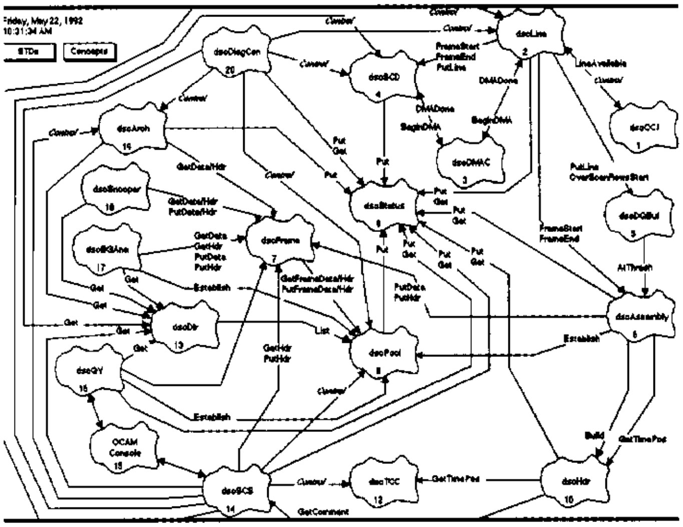


Figure 2: DSC object diagram

The Front End CPU engine for the DSC is the *Motorola MVME167*, a 25MHz 68040 VME CPU module.[6] The operating system employed is *Wind River Systems' VxWorks*.[7] The supported development kit for VxWorks is GNU C. The Front End handles all critical aspects of the data acquisition from data readout to writing to disk and tape.

The Host is a Silicon Graphics 4D/35 running IRIX[8]. To reduce the number of single points of failure, the Host is not in the critical path for data acquisition. Its purpose is to provide the user near-real-time analysis tools and general computing support.

III.D Camera Electronics

The electronics for controlling and reading out the CCD are a combination of devices all constructed at Fermilab by the Detector Electronics Systems Group. The CCD and preamp electronics are contained within a Dewar (a cryogenic container that permits light to pass in one side). This cools the CCD and some electronics to -100° C to reduce electrical noise. The Dewar contents are connected to the Camera Electronics board which digitizes the CCD output signals. A portion of the Camera Electronics interfaces with the VCI via fiber optics to provide CCD data to the data acquisition System. The Control and Status Link interfaces with the Camera Electronics and the Front End to control the CCD Camera and obtain its status. The Camera Electronics are electrically isolated from the VCI and the rest of the system by using a fiber optic cable.

III.E Data Acquisition Control

The DSC run control is centered around a status database and the command interpreter. We present here a summary of the more important features.

III.E.1 Status database

A critical component for the DSC system is the *status database*. In a memory region resident on the Front End MVME167 but accessible from the Host, approximately 150 critical parameters are maintained. When each software component initializes, it creates in the database any status entries required for its operation that may be of interest to other components or the user. Likewise during initialization, a component may lookup any entries that it requires. Implementing accesses to the database via alphanumeric lookups as opposed to simply sharing C data structures alleviates recompilation each time the data structures change during development or installation of new versions.

A status entry consists of an alphanumeric name, a type (integer, string, floating point, etc.), the current and minimum and maximum allowable values, protection (when it can be changed), current validity, and a description. When a com-

[6] Motorola is a registered trademark of Motorola, Inc.

[7] VxWorks is a registered trademark of Wind River Systems, Inc.

[8] Silicon Graphics and IRIX are registered trademarks of Silicon Graphics, Inc.

ponent needs to access an entry, it first locates it by name and is given a handle for later, more efficient access. It is then possible for the component to reference the current value or even change it, the latter depending on its protection.

For example, the status entry *cciRowSize* specifies the number of columns that are currently being read from the CCD. This entry's value is changeable, but only when the data acquisition is idle. Attempting to change it when the data acquisition is active will result in an error.

The status database entries come in eight different classes:

1. Front End CCD and electronics
2. Frame Pool
3. Telescope Control
4. Real time display
5. General data acquisition parameters
6. Frame construction
7. 8mm tape archiving
8. User Defined

III.E.2 User interface

The primary user interface to the DSC system is the interpreted command language, TCL (developed at the University of California, Berkeley by John Ousterhaut, et. al.) [4]. TCL is a C and Lisp like user-extensible command interpreter. We wrote command primitives in C and incorporated them into TCL (i.e., commands were defined to invoke them). The user then uses these commands directly or constructs TCL procedures to perform a sequence of commands.

For example, three of the command primitives defined by the DSC system are *readoutBegin* (begin reading out data from the CCD), *readoutEnd* (end readout), and *statusGet* (query the status database). A command sequence to begin the data acquisition, wait for 5 frames to be acquired, and then end the data acquisition could be:

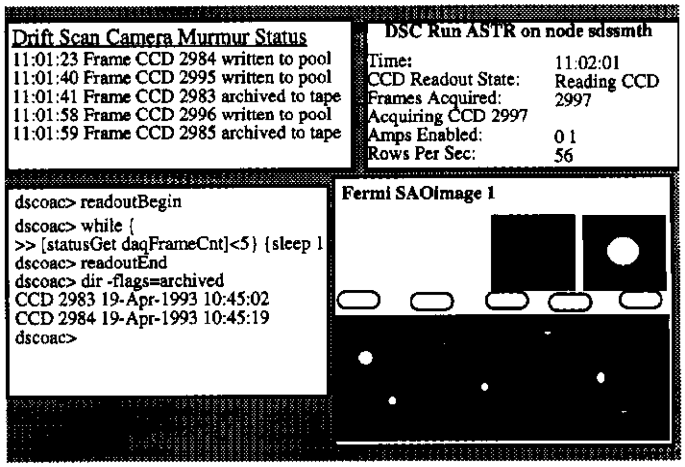
```
readoutBegin
while {[statusGet daqFrameCnt]<5} {sleep 1}
readoutEnd
```

The observer can construct any number of TCL procedures to control any aspect of the data acquisition and analysis. One TCL procedure may refer to another giving the observer considerable power. Other examples of TCL scripts are control of the tape archiving and near real-time analysis.

TCL commands may be issued by the observer either on the Host or directly to the Front End. The commands come in three executing classes: only on the Front End, only on the Host, or on both. Respective examples are data acquisition control commands, graphical data analysis commands, and status query commands. All types may be issued on the Host; however, commands that must be executed on Front End are sent verbatim to it and the execution status is returned to the Host. This approach enables the user to use the same set of data acquisition control commands on either system.

Two Graphical User Interface (GUIs) builders are used: Visual Edge's UIM/X[9] and Tk [4]. UIM/X is a Motif based interface builder that permits one to interactively design a GUI and supplement it with C callbacks. Tk is an X Windows interface package integrated with TCL and developed at Berkeley. They both are used to provide a graphical display of the current state of the system. Since UIM/X produces compiled code, it has the advantage of being more efficient. Tk gives any user with TCL competence the capability to build his or her own interface.

Figure 3 shows some of the tools available to the observer. Starting at the top left and going clock-wise: a Murmur status window [3], a system status display written in Tk, a SAO-Image display window for analysis [5], and a TCL command window for control.



```
Drift Scan Camera Murmur Status
11:01:23 Frame CCD 2984 written to pool
11:01:40 Frame CCD 2995 written to pool
11:01:41 Frame CCD 2983 archived to tape
11:01:58 Frame CCD 2996 written to pool
11:01:59 Frame CCD 2985 archived to tape
```

```
DSC Run ASTR on node sdssmth
Time:                  11:02:01
CCD Readout State:     Reading CCD
Frames Acquired:       2997
Acquiring CCD 2997
Amps Enabled:          0 1
Rows Per Sec:          56
```

```
dscoac> readoutBegin
dscoac> while {
>> [statusGet daqFrameCnt]<5} {sleep 1
dscoac> readoutEnd
dscoac> dir -flags=archived
CCD 2983 19-Apr-1993 10:45:02
CCD 2984 19-Apr-1993 10:45:19
dscoac>
```

Figure 3: Observer's Control Console

III.E.3 Data Acquisition Configuration

Run control configuration is provided via TCL scripts by building upon the supplied command primitives and utilizing the status database. The user is able to execute any number of scripts that will manipulate all aspects of the data acquisition including setting readout parameters, archiving frames to 8mm tape, and performing online data analysis.

For example, the *statusSet* DSC TCL command sets the value of a status entry. The following sequence sets the CCD readout dimensions to 1024 rows by 2048 columns, opens the shutter that covers the CCD for 1000 milliseconds, then begins the data acquisition:

```
statusSet cciRowCnt 1024
```

[9] UIM/X and Visual Edge are trademarks of Visual Edge Software, Ltd.

```
statusSet cciRowSize 2048
shutterOpen 1000
readoutBegin
```

III.F Telescope Control

The DSC system interfaces with the existing telescope control computers at APO. Telescope control consists of focus, positioning, and motion commands. It's actual operation is beyond the scope of this paper.

III.G DSC Status

The DSC system is operational here at Fermilab. Final deployment at APO is anticipated for September, 1993 with the completion of the 3.5 meter telescope. It continues to provide valuable information into the design and implementation of the more ambitious SDSS system. Specifically, we better understand the integration and use of VxWorks, the VME resident hardware (e.g., the MVME167), the VME to Silicon Graphics interconnect, the front-end electronics, and the value of employing a general purpose, user-extensible command interpreter. We also gained from significant software reuse, especially in the analysis tools (See section V.).

IV. SDSS

The SDSS online system expands upon the software and hardware developed and employed by the DSC. The SDSS system is currently under development. We provide a summary of its three major subsystems to show them in comparison to the DSC. The Monitor Telescope subsystem is complete. The Imaging and Spectroscopic subsystems are under development.

IV.A Monitor Telescope Subsystem

The Monitor Telescope is a single CCD and MVME167 system. Its primary purpose is to monitor atmospheric conditions during imaging. The host system is a Silicon Graphics 4D/35. The data rate is one megabyte per second.

The Monitor Telescope system is, in effect, a special purpose DSC system. The main difference being that it has no disk or 8mm tape local to the MVME167. Data as it's acquired is sent directly to the Host machine.

IV.B Imaging Subsystem

The purpose of the Imaging system is to find astronomical objects to be supplied to the Spectroscopic system for analysis. Imaging data that is acquired is analyzed offline to locate the objects.

The Imaging subsystem provides for the readout of 47 CCDs mounted on a 2.5 meter optical telescope. The field of view is three degrees. It consists of 30 photometric and 17 astrometric CCDs. Data acquired from the CCDs is analyzed in real-time to monitor the data quality. The photometric data is used to in the later analysis to locate the objects. The astrometric data is used to determine the precise position of the objects.

This subsystem contains twelve MVME167s, associated Front End hardware, disk and 8mm tape hardware, and the SDSS Host Computer system (a Silicon Graphics Crimson). The acquired aggregate data rate is seven megabytes per second.

Figure 4 shows the system block diagram for one half of the photometric data acquisition system. Another identical system will serve the other half. Photometric data is acquired and written to the frame pool disks. The pooled data is then sent to other parts of the system as needed (e.g., for tape archiving and analysis).

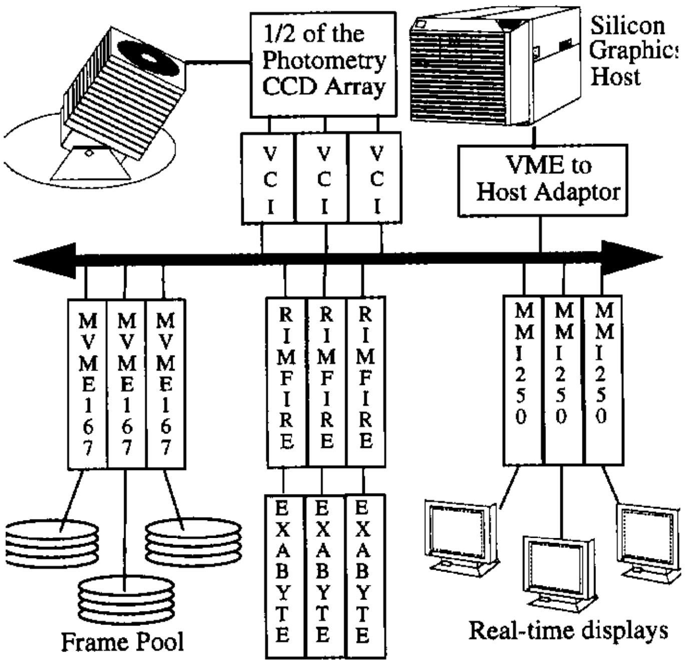


Figure 4: Photometric Online System for 15 CCDs

IV.C Spectroscopic Subsystem

The Spectroscopic System employs two spectrographs each with two CCDs on the 2.5 meter telescope. This single MVME167 system's purpose is to provide spectral information on objects found by the imaging system. The acquired aggregate data rate is four megabytes pre second. As in the Monitor Telescope system, it is a modified DSC system.

V. ANALYSIS TOOLS

Provided with DSC and the SDSS project as a whole is a suite of user support and analysis tools.

SHIVA (Survey Human Interface and Visualization Environment) is the tool kit used for supporting the analysis of data acquired [8]. As discussed, data acquired is assembled into frames. For viewing and analysis, frames are copied in to Shiva *regions*. Shiva provides C and TCL support for performing analysis on these regions. Shiva was developed at Fermilab integrating together:

- PGPLOT: a package for drawing simple scientific graphs on various displays and was developed at Caltech.

- FSAOIMAGE: an X11 window based, interactive, color or halftone image display program for astronomical images[5]. Developed at the Smithsonian Astrophysical Observatory, it has been tailored here at Fermilab for use in SDSS.

- FTCL: A Fermilab value added version of TCL [10]. We have added command line help, VxWorks support,[10] UNIX style command aliasing, and VMS style command line editing. We also integrated Neosofts's Extended TCL package.

- LIBFITS: a routine package for reading, writing, and manipulating FITS frames. Frames are organized using the FITS (Flexible Image Transport System) standard developed at NASA. Roughly speaking, a FITS frame consists of a ASCII header describing that binary data that follows. LIBFITS was developed by Alan Uomoto of Johns Hopkins University [11].

- WWW: the wide-area hypermedia information retrieval system developed at CERN [6]. Shiva online help is provided via FTCL help and WWW. Information browsing is supplemented by the *XMosaic* browser developed at National Center of Supercomputer Applications (NCSA) [7].

VI. SUMMARY OF EXPERIENCES

We have had positive experiences with a number of the software technologies we employed in the DSC system.

We have found TCL to be very useful and will continue to use it; the value of a extensible, command interpreter with the power that TCL provides cannot be overstated. Overall, Wind River's VxWorks (and the GNU based development kit they support) proved an adequate and reliable operating system and was immediately usable. Of the graphical user interfaces we used, both Tk and UIM/X proved useful; Tk's value is that it's integrated with TCL; UIM/X provides an easy to use Motif interface builder.

For the hardware, we found it quite valuable to have full time hardware technical support personnel working closely with the software development staff. On previous projects, we did not have this access and suffered as a result. A summary of our hardware experiences are:

- The Motorola MVME167 proved a solid general purpose VME CPU module; small problems that we had were addressed by Motorola.

- The HVE VME crate extender for connecting the Silicon Graphics to the main VME system has performed smoothly.

- The Silicon Graphics 4D/35 was adequate in most cases; it is not as good as VME neighbor as one would like since it makes unfortunate assumptions regarding VME addressing: on the VMEbus, the Silicon Graphics presumed that it was the only slave being addressed and would interfere with other slaves. We fixed this problem by modifying the HVE link to only respond to addresses within a certain range. Other small problems that we had were addressed by Silicon Graphics.

- The Nanao monitors and Vigra controllers were sufficient for their purpose for providing VME based realtime displays.

- At Fermilab, we have had extensive experience with Ciprico Rimfire 3513 8mm tape controllers and Exabyte 8200 tape drives; this is one of our first experiences with the Exabyte 8500 in embedded VME based data acquisition systems.

- The Micropolis 1921 5400 RPM disks used in the frame pool also have performed well.

Employing UNIX based RISC CPUs gave us access to the high speed hardware necessary for the graphics and near real-time analysis required. However, since we removed any dependence on UNIX from the critical data acquisition path, we avoided some of the common pitfalls that come with UNIX; some examples are NFS failures, dynamic resource depletion, mysterious crashes, file permission problems (this problem sometimes prevents the VxWorks from booting), and others. Some problems are more common due to our heterogeneous UNIX development environment and others are intrinsic to UNIX.

VII. References

[1] "A Digital Sky Survey of the Northern Galactic Cap," Proposal, December 20, 1990.

[10] The actual TCL port to VxWorks as done by D'Anne Thompson, et. al. at the National Optical Astronomical Observatory.

[2] Kent, S., Kron, R., and Stoughton, C.,"A Proposal for the Drift Scan Camera (DSC)," Fermilab Internal Document, March 13, 1992.

[3] Oleynik, G., et. al., "Murmur - A Message Generator and Reporter for UNIX, VMS, and VxWorks," IEEE Real Time '93 Conference Record.

[4] Ousterhaut, J., "TCL: An Embeddable Command Language," Proceedings of the winter 1990 USENIX conference.

[5] VanHilst, H., "User Manual for SAOImage," Smithsonian Astrophysical Observatory.

[6] Berners-Lee, T. J., Cailliau, R., Groff, J., Pollermann, B., CERN, "World-Wide Web: The Information Universe", Electronic Networking: Research, Applications and Policy, Vol. 2 No 1, pp. 52-58 Spring 1992, Meckler Publishing, Westport, CT, USA.

[7] "NCSA Mosaic Documentation," National Center for Supercomputer Applications.

[8] Berman, E., et. al., "The Shiva Book." Fermilab Computing Division Internal Document.

[9] "PGPLOT User's Manual," California Institute of Technology.

[10] Vittone, M., et. al., "FTCL - TCL At Fermilab," Fermilab Computing Division Document PN 464, 1993.

[11] Uomoto, A., "LIBFITS Reference Manual," Johns Hopkins University.

Please refer questions to:
Bryan MacKinnon
Mail Station 234, Fermilab
Batavia, IL 60510, USA
(708) 840-4790, email: mackinnon@fnal.fnal.gov