Fermi National Accelerator Laboratory

FERMILAB-Conf-93/339

Distributing Functionality in the Drift Scan Camera System

T. Nicinski, P. Constanta-Fanourakis, B. MacKinnon, D. Petravick, C. Pluquet,
R. Rechenmacher and G. Sergey

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

November 1993

Presented at the *3rd Annual Conference on Astronomical Data Analysis Software and Systems*,
Victoria, British Columbia, Canada, October 13-15, 1993

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

Disclaimer

Distributing Functionality in the Drift Scan Camera System[1]

Tom Nicinski, Penelope Constanta-Fanourakis, Bryan MacKinnon, Don Petravick, Catherine Pluquet, Ron Rechenmacher, and Gary Sergey

Fermi National Accelerator Laboratory, PO Box 500, Batavia, IL 60510

Abstract. The Drift Scan Camera (DSC) System acquires image data from a CCD camera. The DSC is divided physically into two subsystems which are tightly coupled to each other. Functionality is split between these two subsystems: the front-end performs data acquisition while the host subsystem performs near real-time data analysis and control. Yet, through the use of backplane-based Remote Procedure Calls, the feel of one coherent system is preserved. Observers can control data acquisition, archiving to tape, and other functions from the host, but, the front-end can accept these same commands and operate independently. The DSC meets the needs for such robustness and cost-effective computing.

1. Introduction

The Drift Scan Camera System (DSC) was developed at Fermilab for two major reasons: to be used as a science instrument on the ARC 3.5m telescope at Apache Point Observatory, New Mexico, and to provide experience with using large CCDs (Charge Coupled Devices) for the Sloan Digital Sky Survey (SDSS). Because Fermilab is a High Energy Physics laboratory, its experience in producing high speed data acquisition systems was beneficial in the development of the DSC.

The DSC is split into two major functional units:

- Front-end data acquisition (DA). Its duties include:
    - acquiring data from the CCD Camera
    - displaying acquired data in real-time
    - storing acquired data on disk in the Frame Pool
    - archiving acquired data to 8mm tape.
- Host processing. Its duties include:
    - controlling front-end operations
    - analyzing Frame Pool data in near real-time.

---

[1] Sponsored by DOE Contract number DE-AC02-76CHO3000.

Some quality goals of the DSC include:

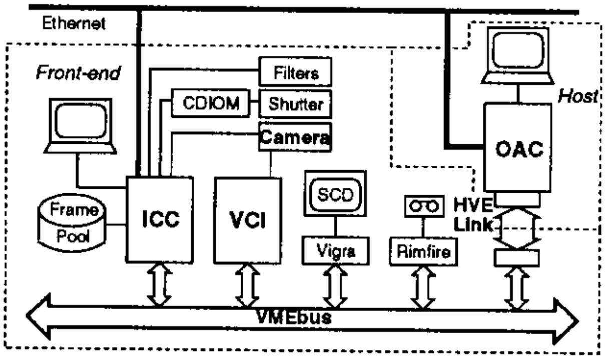
- Responsiveness. The front-end must acquire data at high rates without loss. In addition, it must process host requests for data during acquisition. The host must be able to retrieve and analyze acquired data in near real-time. This is necessary to allow the host to determine, from acquired data, that problems need to be fixed (such as a misaligned instrument rotator).
- Robustness. The front-end data acquisition must be capable of operating without the presence of the host, including an unexpected loss of the host.
- User-friendliness. Graphical User Interfaces (GUIs) may be used. But, DSC must have a command language allowing operation without GUIs.

These goals also apply to the SDSS System being developed. Since SDSS data rates are considerably higher, off-loading operations to the host is even more vital.

2. Isolating Functionality within DSC Components

The two major DSC functional units are isolated physically in two subsystems (see Figure 1).The choice of machines was dictated by cost, computing power, availability of commercial hardware, and the ability to use in-house built hardware. VMEbus is an ideal backplane with the necessary bandwidth. One machine to handle all DSC needs could not be found, necessitating a separate host.

The Instrument Control Computer (ICC) contains the front-end DA while host analysis is done on the Online Analysis Computer (OAC). The ICC is a 20 MIP Motorola MVME167b Single Board Computer. The OAC is a 30 MIP Silicon Graphics 4D/35. These machines' VMEbuses are tightly coupled with an HVE Engineering VME Repeater (link). ICC and OAC processes communicate with each other via shared memory.



Front-end:

VMEbus Camera Interface (VCI): acquires data from the CCD camera across fiber optics.

CDIOM: provides signals to control .the shutter.

Vigra and SCD: provide real-time display of acquired data.

Rimfire: provides archiving of data to 8mm tape.

Figure 1. DSC Block Diagram

3. The Need for Backplane Communications

The data rates of the DSC are not excessive, but the need for the OAC to analyze large data sets in near real-time necessitated backplane communications across shared memory. The DSC operates in one of two modes: drift scanning or staring. With a CCD size of 2048 × 2048 pixels (2 bytes each), the DA acquires data at 461 KBytes/sec. at the maximum drift rate. This data is packaged into Frames and stored on disk in the Frame Pool. The nominal Frame size is the CCD size, approximately 8 MBytes. At the maximum drift rate, an 8 MByte Frame is generated every 18.1 seconds. Buffering Frames to disk allows clients (including the archiver) to access data asynchronous to data acquisition.

For the host to analyze Frames in near real-time, Frames must be moved from the Frame Pool to host tasks quickly. Using the backplane interconnect, Frame transfer times are comparable to the time required to retrieve Frames from disk, about 10 seconds for an 8 MByte Frame; minimal time is spent on the actual transfer. However, coaxial Ethernet involves additional overhead along with transfer times an order of magnitude slower. Thus, using shared memory across the backplane provides the host considerably more analysis time between Frames (without resorting to double buffering in the science code). Besides moving Frames quickly from the ICC Frame Pool to the OAC, with tight coupling, other data can be shared transparently:

- Frame Pool directory. Maintaining, in shared memory, the list of all existing Frames permits OAC tasks to search the directory.
- Status entries. They provide information about the states, capacities, etc., of different DSC components. Host user interfaces can effectively inform users without repeatedly querying the ICC. Control parameter entries are also used to affect many system operations.

The shared data resides in ICC memory. This permits the ICC to continue operations in case the OAC is unavailable. The sharing is relatively efficient, compared to socket-based protocols, since only one indirect memory access is needed. If tasks outside the ICC and OAC need access to the shared data, a server on the OAC can be written to provide that access.

Sharing data across memory also improves system efficiency and reliability:

- Communication overheads are greatly reduced by using DMAs rather than, for example, socket-based protocols.
- Some processing tasks are localized to the "requester." For example, as mentioned above, OAC tasks perform Frame Pool directory searches, freeing up additional cycles for the ICC.
- Fewer points of failure exist (backplanes are more reliable than networks).

4. Distributing Control

Besides distributing functionality, control of the ICC is also distributed. A client/server model is used, where the clients can be on any machine, including the ICC, and the servers are on the ICC. ICC servers are theoretically not service-specific; they can handle all requests from all clients. In practice, servers handle specific jobs. Thus, by using varying server task priorities, client access to the ICC is prioritized. For example, a lower priority server task handles client requests for Frames; these requests cannot then block data acquisition.

Through more shared memory, the ICC can be controlled by the OAC. Remote Procedure Calls (RPC) across the backplane were chosen over a socket-based RPC because of better robustness. Although the hardware independent backplane RPC is somewhat complex, ICC servers and OAC clients can "connect" and "disconnect" from it without affecting their respective peers. The verification of safe backplane critical sections was considered simpler than handling connection requests and abnormal disconnects in a socket-based protocol.

4.1. Modularizing Commands

Having ICC servers handle all client requests is greatly simplified by using one common interpretive language throughout the DSC. Tcl (Ousterhout 1993) is a C and LISP-like extensible command interpreter. Commands can be added as Tcl procedures or by easily interfacing C routines to the interpreter. DSC uses low-level C routines with Tcl routines layered above to provide a user interface. As Tcl behaves the same on all platforms, users see no distinctions between issuing commands from the OAC or the ICC.

OAC clients issue Tcl commands to ICC servers via the backplane RPC while ICC clients execute Tcl commands directly. The GUIs on the OAC, acting as clients, emit Tcl commands. Not only is a user-friendly environment provided, but the system can be run with only line oriented commands, as the GUIs use the same Tcl interface to ICC and OAC functions. GUIs can be brought up quickly by using Tk (Ousterhout 1993), the Tcl toolkit for X Windows.

4.2. Using Existing Apache Point Facilities

DSC also interfaces to existing Apache Point machines on the local network:

- Telescope Control Computer. It controls the 3.5m telescope and enclosure. It broadcasts UDP packets containing telescope position/time pairs.
- Master Computer. Through a TCP/IP socket-based protocol, the DSC system issues telescope and enclosure movement commands and obtains information about the telescope, weather, etc.

5. Results

The DSC System has met its goals successfully. It's quite responsive and robust. Users have quickly written considerable code (much in Tcl) to perform analysis on the OAC and to control the ICC.

The DSC DA and communications architecture will be used as the base for the more ambitious SDSS System. Due to modularity, many DSC software components can be reused with little or no changes. With transparent access to shared data, such as the Frame Pool directory, the OAC off-loads work from the ICC. With SDSS' higher data rates (7 MBytes/sec. to 9 ICCs), such off-loading is necessary.

References

Ousterhout, John K. 1993, *An Introduction to Tcl and Tk*, Addison-Wesley.