Data Acquisition and Analysis at the Structural Biology Center

M. L. Westbrook, T. A. Coleman, R. T. Daly
ECT-EE, Argonne National Laboratory, Argonne, Illinois 60439, USA
mwestbrook@anl.gov

J. W. Pflugrath
Molecular Structure Corporation
The Woodlands, Texas 77381, USA
jwp@msc.com

NOTE: Visit the SBC Web Site http://www.sbc.anl.gov/ for an introduction to the SBC. This document is http://www.sbc.anl.gov/IUCrpresent.html.

1. Introduction

The Structural Biology Center (SBC), a national user facility for macromolecular crystallography located at Argonne National Laboratory's Advanced Photon Source, is currently being built and commissioned. SBC facilities include a bending-magnet beamline, an insertion-device beamline, laboratory and office space adjacent to the beamlines, and associated instrumentation, experimental apparatus, and facilities. SBC technical facilities will support anomalous dispersion phasing experiments, data collection from microcrystals, data collection from crystals with large molecular structures (such as viruses and ribosomes), and rapid data collection from multiple related crystal structures for protein engineering and drug design.

The two major goals of the Structural Biology Center are to provide extremely brilliant beamlines for studies that cannot be performed elsewhere and to maintain a high user throughput in order to serve the largest number of crystallographers possible (Smith and Watenpaugh, 1991). To this end, the SBC facility has been designed and built with an emphasis on both speed and brightness. Each SBC beamline is equipped with x-ray optics and a control system designed to deliver a stable, intense, highly focused, physically small x-ray beam with low angular divergence onto protein crystal samples. Diffraction patterns generated on SBC beamlines are recorded on physically large, highly efficient electronic area detectors and are stored by a control system capable of fast data transfer and analysis. To achieve these goals, SBC construction has been conducted by coordinated groups specializing in x-ray optics, detectors, real-time data acquisition and controls, mechanical engineering, and conventional facilities. All SBC components — optics, detector, and controls — are explicitly designed to work together as an integrated system.

The SBC Computing Systems and Software Engineering Group is tasked with developing the SBC Control System, which includes computing systems, network, and software. The emphasis of SBC Control System development has been to provide efficient and convenient beamline control, data acquisition, and data analysis for maximal facility and experimenter productivity (Ealick and Walter, 1993). This paper describes the SBC Control System development, specifically data acquisition and analysis at the SBC, and the development methods used to meet this goal.

2. The SBC Control System

Figure 1 outlines the software engineering methodology followed in SBC Control System development (Structural Biology Center, 1994c; Humphrey, 1989).

2.1. *Control System Development*

2.1.1. *Requirements*

In the first phase of control system development, the SBC Control System developers acquired a complete understanding of the facility specifications and user needs. This information was gained through interviews of experimenters, beamline operators, and beamline optics and detector designers, as well as from the experience gained through the operation of a user program at SBC beamline X8C at the National Synchrotron Light Source (NSLS). The SBC Control System Requirements (Structural Biology Center, 1994a) represents a compilation of facility and user specifications and has formed the basis of subsequent hardware and software design.

For the control system, the technical challenges have been in the areas of rapid data acquisition and ease of facility use. Users specified a requirement for rapid data

DISCLAIMER

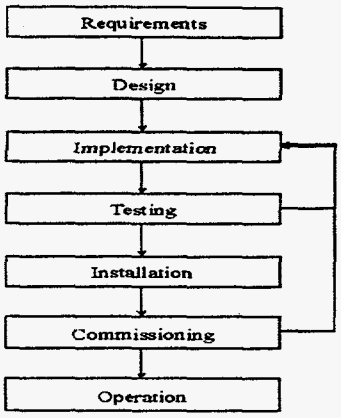

Figure 1 Software engineering methodology applied to SBC Control System development

acquisition rates in order to minimize the exposure of labile protein crystals to x-rays; time-dependent radiation damage ultimately limits the quality and quantity of data that can be measured from each crystal. Users also stressed the need for an ability to easily and correctly control the beamline, collect data, and analyze the data.

Another requirement of the SBC program is that the facility must remain technically modern for years so that it will remain productive. It has been a challenge for the control system developers to produce a design that will accomplish this.

2.1.2. *Design*

A design phase followed control system specification. In this phase, the methods for implementing the requirements were prescribed, and commercial computers, network components, and commercial software products were identified. The tools needed to complete the task of control system development were identified and acquired. During the design phase, tasks were identified, assigned to individual developers, and scheduled.

By design, the SBC Control System architecture (Structural Biology Center, 1995g) comprises the major hardware and software components shown in Figure 2. Because our experience has shown that software development is driven by the hardware, the SBC systems and network were designed first. Following this, the control system software was designed and implemented.

2.1.3. *Implementation and testing*

Control system implementation followed the design phase of the project. Initially, the implementation phase

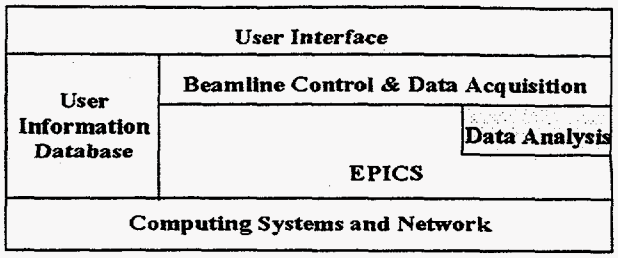

Figure 2 SBC Control System architecture

involved the installation of software development computers in the laboratory, the installation of Experimental Physics and Industrial Control Software (EPICS) and the installation of commercial and other software (such as the GNU compilers and other tools) on these development systems. Implementation of the SBC Control System has been conducted in a bottom-up fashion, as shown in Figure 3.

Individual device control and monitoring was addressed first, starting with motors, due to the large number of motors (approximately 70 per beamline) used on each beamline. The SBC developers demonstrated the driving of a dc servo motor for the first time, in October 1995. Since that time, motor support has been extended to motor movements involving multiple axes, as found in beamline optics components such as the double-crystal monochromator, which was controlled to deliver the first monochromatic x-rays into the insertion-device beamline end-station on May 7, 1996.

Data analysis software (d*TREK) has been developed by Molecular Structure Corporation (MSC) to SBC specifications (Structural Biology Center, 1994b). The interface between the SBC Control System and d*TREK was produced as a collaboration between MSC and the SBC Computing Systems and Software Engineering Group. SBC Control System software development followed a formal change management system in order to coordinate the work of multiple software developers and to

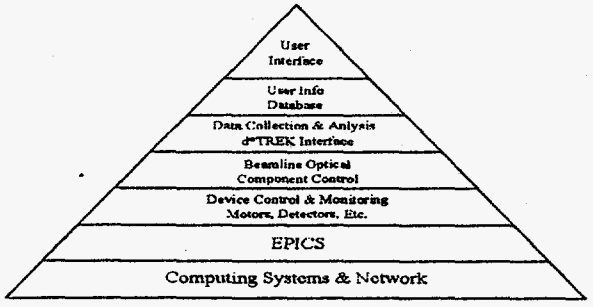

Figure 3 SBC Control System bottom-up implementation

control the software development change process (Structural Biology Center, 1994b). SBC software configuration management (Structural Biology Center, 1995j) follows the software configuration management developed for the APS Control System, which also uses EPICS.

Each phase of control system development has been reviewed internally. Unit and integration testing of control system software is being conducted by the developers during the implementation phase. The Technical Advisory Committee, which is an external review committee appointed by the SBC Principal User's Group, conducted periodic technical reviews of SBC Control System development progress as well.

2.1.4. *Installation*

The facility network, computing systems, control system software, and data analysis software required to support SBC insertion-device beamline operation have been installed. The SBC bending-magnet beamline is currently being installed.

2.1.5. *Commissioning*

The insertion-device beamline, the SBC Control System, and the d*TREK data analysis software are now being exercised as an integrated system. Testing of control system functionality against requirements is being conducted; this testing is referred to as "beamline commissioning." The bending-magnet beamline will begin commissioning in the fall of 1996.

2.1.6. *Future operations*

The SBC facility, the insertion-device beamline, and the bending-magnet beamline are scheduled to be operational in April 1997. Users may gain access to SBC facilities through a peer-review process. SBC policies regarding production-mode operation are currently being determined.

2.2. *Computer Systems and Network*

The SBC computer systems and network were designed (Structural Biology Center, 1995k) to provide a distributed real-time computing and data acquisition environment that meets or exceeds user requirements. SBC computers and network were designed to support the aggregate tasks of beamline control, data acquisition, data analysis, data archive, and data visualization. SBC users are encouraged to take advantage of the high-performance computing and networking capabilities by performing data analysis tasks on-site.

No single network architecture was found to be capable of supporting all beamline functions equally well. Three network architectures are used at the SBC: a high-performance parallel interface, 800 Mbps as a dedicated data transfer pathway for the data acquisition; Ethernet, 10 Mbps for distributed control of beamline components and end-station instrumentation; and Asynchronous Transfer Mode (ATM) OC-3c, 155 Mbps (with the ability for future upgrade to OC-12c at 622 Mbps), for data visualization and analysis. SBC developers have measured 7 MB/s data transfer rates via ATM with ftp (file transfer protocol) network protocol overhead, a factor of nine times the performance over Ethernet. SBC ATM network components, switches and adapters, are from Fore Systems. The complete system and network diagram is shown in Figure 4.

One file and compute server — an SGI Challenge L server with large RAID (redundant array of inexpensive disks) arrays — is provided on each beamline. The SGI Challenge was selected based upon its support of many features shown in Figure 5: symmetric multiprocessing (SMP) capability (2-12 CPUs), system memory expandable to 6 GB, 1.2 GB system bus, multiple SCSI-2 and SCSI-2 fast wide differential controllers, and multiple network interfaces (ATM OC-3c, Ethernet, HIPPI). SBC beamline SGI Challenge servers are running an IRIX V5.3 operating system.

The SGI logical volume disk driver (XLV) provides access to disk storage as a logical volume. A logical volume behaves like a disk partition while the disk storage may span several physical disks. With XLV, the SBC has combined three Fujitsu DynaRAID Arrays to create a larger logical volume (see Figure 5) and stripes the data across the three RAID arrays, each on its own SCSI-2 FWD controller, to create a logical volume with higher I/O performance. The SBC uses an eXtended File System (XFS) file system on this large logical volume. XFS is a 64-bit file system that supports file and file system sizes up to one terabyte. With a standard 32-bit file system, which is limited to an 8-GB file system size, one file system would not accommodate a data scan of up to one thousand 18-MB images.

The SBC user workstations are SGI Indy R4000 systems running IRIX V5.3. The SBC control workstations are HP 9000 J200 workstations running the HP-UX 9.0.7 operating system. The HP workstations were selected based upon their complete compatibility with the EPICS software and their support of multiple (2) CPUs. Modular hardware implementations are used wherever possible to permit the control system to expand and upgrade smoothly.

When working at any of the SBC computers, users will have ATM connectivity to the beamline file and compute server for enhanced data visualization and analysis performance. SBC workstations are both Ethernet- and

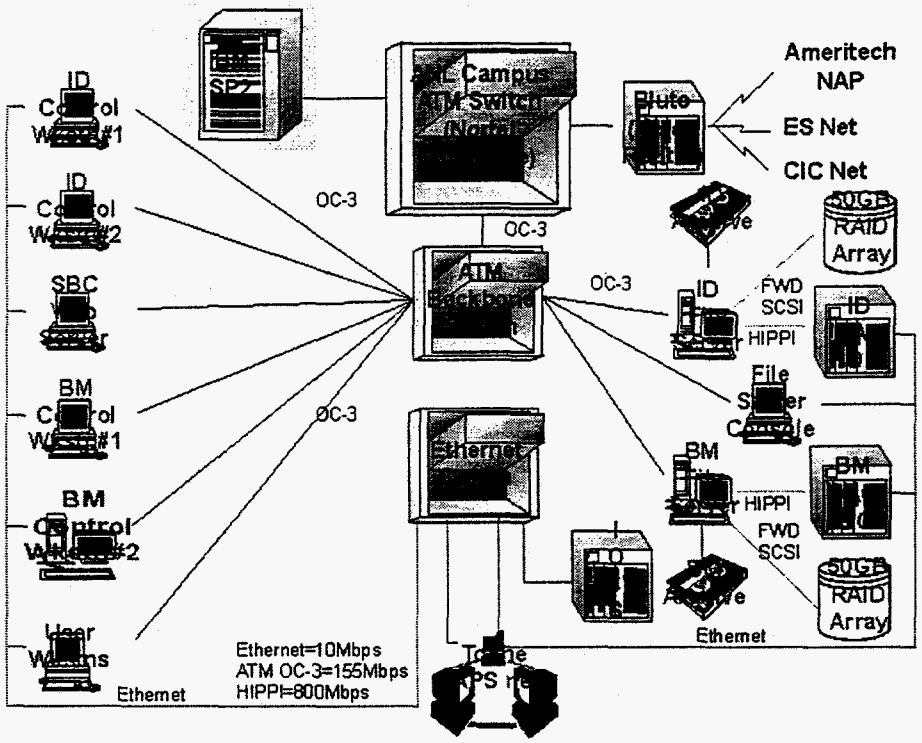

Figure 4 SBC systems and network

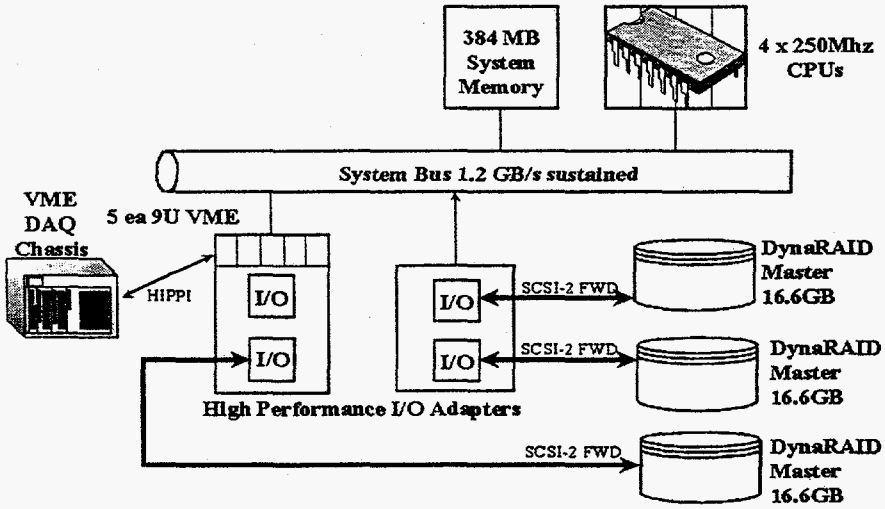

Figure 5 SBC File and Compute Server architecture

ATM-attached for redundancy. The SBC workstations are configured to use the ATM path unless traffic is bound for the local attached Ethernet LAN, or if the ATM network fails. SBC users may transfer reduced data to their home institutions over the Internet.

The SBC maintains the capability to archive data for beamline archival purposes and for user data interchange. Beamline data archiving is provided to facilitate any data reprocessing that may be required and is conducted on two digital linear tape (DLT) libraries, each consisting of a DLT tape drive and robotics to accommodate a seven-cartridge magazine. DLT technology was selected based upon media capacity (20 GB native and up to 40 GB if 2:1 compression is achieved) and average data transfer rate (1.5 MB/s native and up to 3.0 MB/s if 2:1 compression is achieved). SBC developers have observed a 2.3-MB/s sustained data transfer rate for 18-MB images. The SBC intends to archive data for up to 100 days, following the example set by the

European Synchrotron Research Facility (ESRF) Computing Services Group. Both 4-mm and 8-mm 8500 Exabyte tape drives are provided on each beamline file and compute server for user data interchange. Users with access to the DLT drives at their home institutions may also archive to DLT libraries using the Unix "tar" command.

2.3. *Data Acquisition with the APS 1 CCD Detector*

The experiment to be conducted at the SBC is a macromolecular crystallographic x-ray diffraction experiment in which a protein crystal is exposed to x-rays and the x-rays diffracted from the protein crystal are recorded by an electronic area detector. Because the SBC utilizes synchrotron x-rays, extremely short exposure times (down to 10 milliseconds) are expected. For each electronic image, the sample is rotated at constant angular speed through a small rotation angle while the diffraction data are recorded by the detector. A data scan consists of a number of images taken sequentially in angular position. At the SBC, the sample is rotated by a high-precision rotary stage with three arc-seconds of angular reproducibility.

The electronic area detector utilized is the APS 1 CCD detector (Westbrook, 1996), which has a large front face area of 21 x 21 cm. The APS 1 CCD detector is composed of a 3 x 3 matrix of 1024 x 1024-pixel CCD chips bonded to reducing fiber optic tapers. Each APS 1 CCD detector image contains a square 3072 x 3072-pixel array of 16-bit integers. The readout electronics have been designed modularly, with an independent readout module for each CCD chip. The time required for full-resolution readout, which results in an 18-MB image, is 1.8 seconds. The APS 1 CCD detector may also be readout in a 2 x 2 binned mode, resulting in a 4.5-MB image readout in 0.45 seconds. Binning reduces the spatial resolution of the detector and therefore is not possible for experiments where resolution is critical, such as with large-unit-cell crystals. However, binning reduces the load on disk storage while improving the detector's dynamic range and reducing noise.

If the SBC Control System were to add no time overhead to an experiment, it must perform all data manipulation tasks (such as writing the image to disk) during the time it takes to record and read out the next image. In this case, the control system would realize a 10-MB/s sustained data throughput. Based upon this sustained data rate, users will be able to collect data very quickly compared with a conventional x-ray source (see Table 1).

In this form of experiment, it is critical that the sample be moved at constant speed and that the angle through which the sample is rotated while exposed to x-rays (i.e., for each image) be consistent and contiguous. To achieve this, the Delta Tau Systems PMAC (Programmable Multi-Axis Controller)-VME motor controller is used to control the position and speed of the crystal orientation axis (also called the goniometer Omega axis). The PMAC-VME programmable feature is also used to calculate and maintain a constant speed of rotation during image acquisition.

SBC developers use the PMAC-VME programmable feature to define the start and end positions for an image using real-time angular readback data from a shaft encoder attached to the Omega axis motor and to output control pulses at each position. These pulses are used to precisely control an x-ray shutter which defines the sample x-ray exposure interval time, the gating of the APS 1 CCD detector, and any detector used to measure the actual dose of x-rays to the sample (i.e., a beam intensity monitor). To assure that the gating of all devices is done accurately and with minimal latencies (which rules out software control) a hardware solution was devised. A Data Acquisition (DAQ) Synchronization (Sync) Module (Figure 6) was designed with an on-board general-purpose VME interface. The DAQ Sync Module has the ability to delay the gating of any device by a predefined time; in this way the delay in opening of the x-ray shutter can be accommodated. The goniometer PMAC-VME motor controller and DAQ Sync Module represent key data acquisition subsystem hardware components (see Figure 6).

Table 1. *APS 1 CCD detector model data scans at 10 MB/s sustained data rate*

Image type	Image size (MB)	No. of images	Capacity (GB)	Collection time (min)
High-symmetry crystal (50° scan of 0.2°/image)				
Full resolution	18	250	4.5	7.5
2 x 2 binned	4.5	250	1.125	1.8
Low-symmetry crystal (185° scan of 0.2°/image)				
Full resolution	18	925	16.7	27.75
2 x 2 binned	4.5	925	4.16	6.5

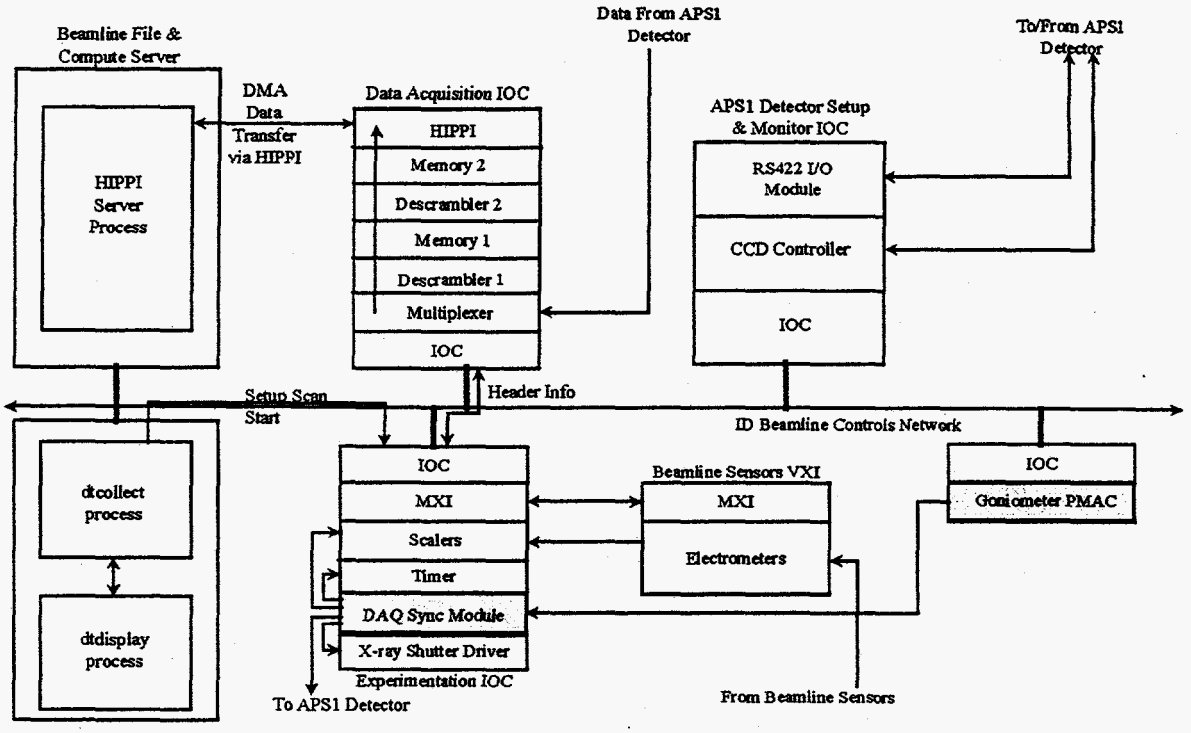

Figure 6 SBC beamline data acquisition subsystem

The sequence of events during data acquisition for a single image is as follows:

- The goniometer Omega axis PMAC-VME controller calculates the speed required based upon the image exposure time and image rotation angle specified by the user.
- The goniometer PMAC-VME sends a "sync" hardware pulse, which represents a "get ready" signal, to the DAQ Sync Module and starts to accelerate the goniometer Omega axis drive.
- The goniometer PMAC-VME sends a "start" pulse to the DAQ Sync module when it has reached a predefined starting position for the image.
- Upon receipt of the "start" pulse, the DAQ Sync module then gates the x-ray shutter, APS 1 CCD detector, and beam intensity monitor "on".
- Once the goniometer Omega axis is in the end-of-image position, it sends a "stop" pulse to the DAQ Sync module, decelerates the motor, rewinds the motor, and repositions the motor for acquisition of the next image.
- Upon receipt of the "stop" pulse, the DAQ Sync module then gates the x-ray shutter, APS 1 CCD detector (which initiates detector readout), and beam intensity monitor "off".

SBC developers have confirmed that the x-ray shutter opens and closes at the proper motor encoder counts (i.e., position) and for the correct exposure time (i.e., speed) for slow motor speed scans. PMAC internal data gathering capabilities can be used to confirm the data acquisition synchronization accuracy at fast motor speeds.

When the APS 1 CCD detector is read out, the data are passed serially in a fixed order from 18 pixels at a time (each of the nine CCDs has two readout amplifiers) and transferred to a VME Multiplexer module (see Figure 6). The 18 pixels are scattered across the detector face. The Multiplexer module rearranges the data into 16-bit parallel words, which represent the signal measured by each pixel where the signal is proportional to the number of x-rays detected, and passes the 16-bit words to a VME Descrambler module. The VME Descrambler reorders the pixels assembling a sequential image, sequential in x- and y-positions, in VME memory. While in development, the VME Descrambler module is being used in pass-through

mode and the image descrambling task is being done in software. The task of moving the data from VME memory to disk on the beamline file and compute server is described in Section 2.4.1.4, Data Acquisition.

In this mode of operation, the APS 1 detector is read out into one VME memory module, and while the image data is written to disk, the next image is taken and transferred to a second memory module. By "ping-ponging" between memory modules, the data may be transferred to disk without any additional overhead.

The use of on-line compression of image data was explored during the design phase of the project. Many factors contributed to the decision not to compress image data. Among them:

- Developers observed varying compression ratios of 1:1 to 5:1 for test image data. It is difficult to produce an optimal design for compression done in hardware in this case.
- Compression of the image data would burden the memory and CPU use of the file and compute server, because uncompressing would be done in software. Data analysis and visualization are already using these resources intensively.
- The SBC Control System is already fast enough to handle uncompressed (18 MB) images, using commercially available hardware at a 10-MB/s sustained data transfer rate.
- For data archiving, the facility DLT Archive Libraries already use built-in compression methods.

The control system design presented in this paper represents the most cost-effective solution.

2.4. *Control System Software Development*

SBC Control System software is made up of the five major components:

- EPICS-controlled device drivers
- data analysis software, d*TREK
- the interface between d*TREK and EPICS-controlled drivers
- user information database (planned)
- user interface

2.4.1. *EPICS*

The SBC Control System software is implemented with EPICS for many reasons, among them, compatibility with the APS Accelerator Complex Control System (a wealth of devices for which support has already been developed), and the distributed and scaleable environment that EPICS provides (Dalesio et al., 1991).

EPICS is a collection of code and documentation—a software toolkit—for building control systems in the 1990's. EPICS was originally developed by Argonne National Laboratory and Los Alamos National Laboratory for the purpose of building distributed control systems to operate accelerators and large experiments. Current development is a collaboration between Argonne, Los Alamos, Lawrence Berkeley National Laboratory, Brookhaven National Laboratory, the Continuous Electron Beam Facility (CEBAF), DESY, and the Telescope community.

EPICS architecture

- follows the "standard model" for current control system architectures
- follows a client/server model
- is "standards based" (UNIX, X Windows, Ethernet and FDDI, TCP/IP, and VME)
- uses distributed real-time process variable databases, with no central database

The physical architecture of an EPICS control system consists of a Local Area Network (LAN) that will support the distributed real-time computing and data acquisition tasks conducted at the beamline. A computer workstation, or operator interface (OPI), will exist on the LAN to support user access to the SBC Control System. The SBC Control System will interact with network-based single-board computer(s), called input-output controllers (IOC), installed in a VME chassis for access to beamline and end-station hardware (Figure 7). EPICS requires that Wind River Systems' vxWorks real-time kernel be running in the IOC. VME device-specific control modules (such as VME motor control modules) are provided by the control system to support user access to the hardware.

EPICS software tools (Figure 8) provide:

- the ability to interface data acquisition and control to instrumentation through use of a database.
- a tool to build such databases.
- an operator interface to the control system through interactive displays.
- a tool to build such interactive displays.

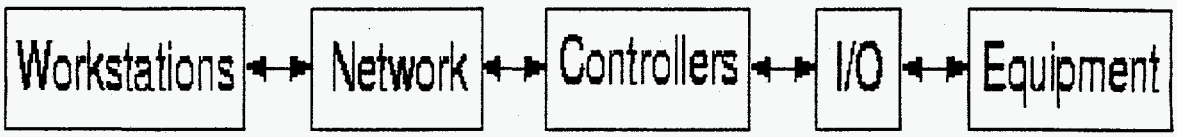

Figure 7 EPICS: control system architecture for the 90's

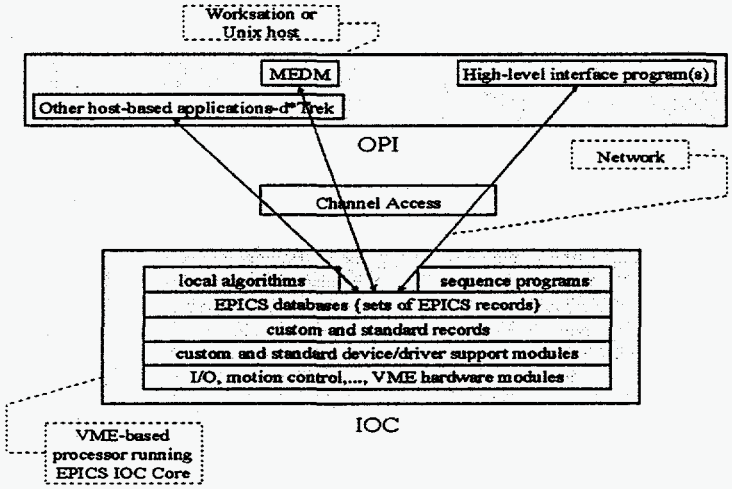

Figure 8 EPICS features

- data logging and alarm management.
- sequential control through a state notation language. EPICS provides a state notation compiler that is a C preprocessor which converts state notation language source code into C code. Through state notation language programming, SBC developers can write programs as a sequence of related states (typically, hardware states) within a database(s). State notation language also provides easy access to EPICS database information.
- network access (Channel Access) to the databases for interfacing the control system to data analysis, third-party software, and any other function not provided by the control system (d*TREK, IDL, and others).

By adopting EPICS, the developers were able to make use of the collaborative work of others over the past 10 years, in this way minimizing the amount of custom coding required to implement the SBC Control System. The task of control system development largely became one of system integration rather than new software development.

2.4.1.1. *Beamline dc servo motor controller*

The SBC developers were required to write device and/or driver support for any SBC beamline or end-station devices not currently supported by EPICS. One important contribution that the developers have made to the EPICS collaboration has been in the area of device and driver support for a beamline dc servo motor controller, specifically EPICS support for the Delta Tau Systems PMAC-VME. The use of the PMAC-VME programmable feature is described in Section 2.3.

2.4.1.2. *APS 1 CCD detector*

SBC developers have also implemented custom EPICS device support for the APS 1 CCD detector. Software modules have been developed to set up and monitor individual CCD gains, offsets, and temperatures via an RS-422 Interface. Software modules have also been provided to set up and monitor detector enclosure temperature and humidity and individual CCD Thermo-electric Cooler (TEC) Controller currents and detector power supply voltage analog signals via a XYCOM Analog-to-Digital VME Module. Software modules used to monitor and control approximately 200 APS 1 CCD detector parameters (such as readout mode and detector

state via an RS-422 Interface) have been implemented. This software serves to aid detector engineers with routine detector setup, monitoring, and control tasks. This software has been implemented via EPICS device support and EPICS databases, which are downloaded to the APS 1 CCD Detector Setup & Monitor IOC (see Figure 6).

2.4.1.3. *Data transfer*

Image data is transferred from VME memory to memory on the beamline file and compute server, an SGI Challenge, via High-Performance Parallel Interface (HIPPI) protocol. On the Challenge, a HIPPI Module and software driver are provided by SGI. On the Data Acquisition IOC shown in Figure 6, a commercial CHI Systems HIPPI-VME interface is used. A custom HIPPI VxWorks driver was developed by the SBC to support this interface. A raw-character-mode protocol, rather than a TCP/IP protocol, was used to eliminate the software overhead needed to implement the TCP/IP protocol. A 22-MB/s data transfer rate has been sustained for transfers between VME and SGI memory. At this rate, only 15% of the IOC CPU capacity was used. Once the image data is in SGI memory, the data can be transferred to the RAID disk array on the SGI Challenge at a sustained rate of 28 MB/s.

2.4.1.4. *Data acquisition*

The **dtcollect** module (see Section 2.4.2.4) represents the user's primary interface to data acquisition at the SBC. To implement the data collection scan specified by the user using the **dtcollect** graphical user interface (GUI) requires the coordination of a number of programs running in the Data Acquisition IOC, the Experiment IOC, and the SGI Challenge (see Figure 6).

The scan information (i.e., exposure time, start angle, end angle, etc.) for a particular data acquisition scan specified within **dtcollect** is sent to the Experiment IOC. The Experiment IOC coordinates most of the work involved in acquiring images. It controls the motors used to rotate the crystal mounted on the goniostat, the opening of the beamline shutter used to expose the crystal to x-rays, and the gating of the APS 1 CCD detector. It also gathers ancillary experimental data (such as scalar data) which will be inserted into the header of each image. Once the APS 1 CCD detector has been exposed to x-rays and the image-specific header data has been collected by the Experiment IOC, the Data Acquisition IOC is notified to prepare to transfer an image and image-specific header data to the SGI. Once notified, the Data Acquisition IOC transfers the image-specific header data into an internal buffer, notifies the Experiment IOC to prepare to take the next image, and waits approximately 1.8 seconds for a full 18-MB image from the APS 1 CCD detector to be read into VME memory. When the image is in VME memory, the Data Acquisition IOC starts a HIPPI transfer of the image-specific header data and the image data to a "HIPPI server" program running on the SGI Challenge (see Figure 6). Each HIPPI transfer to the Challenge is followed by a reply from the Challenge indicating the success of the HIPPI transfer and the success of any outstanding disk IO. If any errors are found, the Data Acquisition IOC flags the Experiment IOC to repeat the collection of the bad image.

The "HIPPI Server" program running on the Challenge is responsible for verifying the integrity of the data received over the HIPPI interface, assembling a complete ASCII header, writing the ASCII header and image data to the RAID disk array, and replying to the Data Acquisition IOC for each HIPPI packet received. The complete ASCII header information written to disk by the HIPPI Server program is composed of information in a scan-specific disk file generated by **dtcollect** and the image-specific header data received by the HIPPI Server program over HIPPI. The **dtcollect** module uses site-specific information stored in a site-specific header file, together with information specified by the user when defining the data acquisition scan via the **dtcollect** GUI, to generate the scan-specific header data file. The HIPPI Server program verifies that no HIPPI data transfer errors have occurred, verifies that the fixed fiducial data inserted into each image by the APS 1 CCD detector controller is correct, starts an asynchronous disk write, and starts a HIPPI read for the next image. Using an asynchronous disk write allows the HIPPI Server program to overlap the disk write of the previous image with the HIPPI read of the current image.

During data acquisition, the Data Acquisition IOC keeps track of the last file successfully written to disk by the HIPPI Server program. **dtcollect**, during data acquisition, displays current information on the progress of image collection, progress of the overall data acquisition scan, and the current goniostat position. In addition, **dtcollect** obtains the last image file written to disk information from the Data Acquisition IOC and passes this information via an X-windows property to the **dtdisplay** application program (see Section 2.4.2.6). In this way, **dtdisplay** can display images immediately after they are written to disk.

This discussion assumes the use of hardware descrambling. However, during development, data are descrambled in the "HIPPI server" program, which contributes an additional overhead associated with each image. The overhead for each image has contributions from

the following tasks, which are currently conducted in sequence:

- Detector readout (1.8 s)
- HIPPI transfer of image to the Challenge (1.2 s)
- HIPPI server assembles image header (1.25 s)
- Image fiducial verification (0.007 s)
- Software descrambling of image data (4.6 s)
- Rewind, backlash correction, and ramp of omega motor to speed for next image (5 s). During this time, the image is written to disk.

Therefore, during development, the time between exposures is approximately 14 s. SBC developers are currently developing a hardware descrambler module. Also, the double buffer "ping-ponging" is still being implemented. SBC developers will be overlapping the above tasks to allow data acquisition to operate concurrently rather than consecutively. When these changes are fully implemented, overhead in the final system will be decreased significantly.

2.4.2. *d*TREK software development*

d*TREK software, a third-generation area detector data acquisition and analysis software, is the primary means by which users interact with SBC beamlines to collect single-crystal x-ray diffraction images and produce estimated intensities and associated standard deviations for the Bragg reflections which appear on those images.

Some notable d*TREK features include the following:

- d*TREK was designed and implemented to be device-independent to accommodate future hardware changes.
- d*TREK design (Structural Biology Center, 1995c) is object-oriented and consists of the abstract entities: source, crystal, goniometer, detector, images, and reflection lists. These entities are application-oriented abstractions that model a single-crystal x-ray diffraction experiment (Pflugrath, 1996) and therefore are easily extensible to a range of hardware. d*TREK object entities have been implemented as classes and objects in C++.
- d*TREK device clients consist of classes or objects in a tool kit (see Section 2.4.2.10, d*TREK - SBC Control System Interface).
- The d*TREK software can function on UNIX operating systems, OpenVMS, and Windows NT.
- d*TREK software development is moving toward single-button or no-button control.

2.4.2.1 *Memory requirements*

At the SBC beamlines, d*TREK will process a large amount of data (i.e., two-dimensional diffraction images). For the APS 1 CCD detector, each image is 18 MB in size. A 1000-image data scan will consume 18 GB of disk space. These images will be reduced to a large list (>50,000).of Bragg reflections. Therefore, sufficient resources are required to run d*TREK. While d*TREK requires a minimum system configuration of 64 MB core memory and 300 MB swap space, it performs better when the data analysis systems are configured above the minimum. To reduce the memory requirements, d*TREK divides data analysis into separate processes, in contrast to a monolithic program such as MADNES, the Munich Area Detector NE System (Messerschmidt and Pflugrath, 1987; Pflugrath, 1996), which uses the operating system to swap parts of a process.

2.4.2.2 *Relationship to MADNES*

d*TREK implements many of the algorithms and ideas which have roots in previous work with the EEC Cooperative Workshop on Position-Sensitive Detector Software (Bricogne, 1986) and with MADNES. These ideas and algorithms have been tested and proven through years of use at the Structural Biology Center facility at the National Synchrotron Light Source beamline X8C (Barford, 1994; Lewis et al., 1996) and elsewhere. d*TREK goes beyond MADNES by providing the capability to merge, scale, and average reflection lists.

2.4.2.3 *Distribution of data processing*

d*TREK, by design, allows for the distribution of data processing tasks to enhance overall data analysis throughput. Images on disk, which are available to the SBC beamline SMP file and compute server, can be processed in a distributed manner in the following ways:

- One CPU could integrate reflections in one-half or one-ninth of the image, while other CPUs can integrate reflections in other areas of the image, with each CPU generating a reflection list.
- Each CPU can process a batch of data that consists of some portion of the data scan.

In each case, all reflection lists can then be merged to create a final result.

2.4.2.4 *Data acquisition with d*TREK dtcollect*

SBC beamline users conduct single-crystal x-ray diffraction experiments with **dtcollect** (Structural Biology Center, 1996a). Users may specify the data acquisition parameters for a single image or a data scan using the **dtcollect** GUI shown in Figure 9. Users may specify the data acquisition scan parameters for a single data scan or a series of data scans using the **dtcollect** Scan GUI screen shown in Figure 10. The **dtcollect** GUI has been implemented in X/Motif. The Dtcollect X Resource file allows customization of the **dtcollect** screens.

With **dtcollect**, a user can control and monitor SBC beamline end-station instrumentation (Figure 11) to achieve the following:

- collect images in a variety of scan modes: shutter-closed (i.e., dark images), shutter-open still images, and shutter-open rotation images
- control and monitor the x-ray shutter
- control and monitor the APS 1 CCD detector
- control and monitor the beam intensity monitor and fluorescence detector (such as the PIN Diode detector)
- control and monitor the crystal goniometer position
- control and monitor the detector goniometer position (at a future date when one becomes available)
- control and monitor source wavelength automatically through use of a wavelength chooser
- monitor experiment progress through instantaneous feedback of device status
- create, load, edit, and save scan tables
- collect images which may be processed with other d*TREK modules
- communicate with d*TREK dtdisplay to display images as they are collected

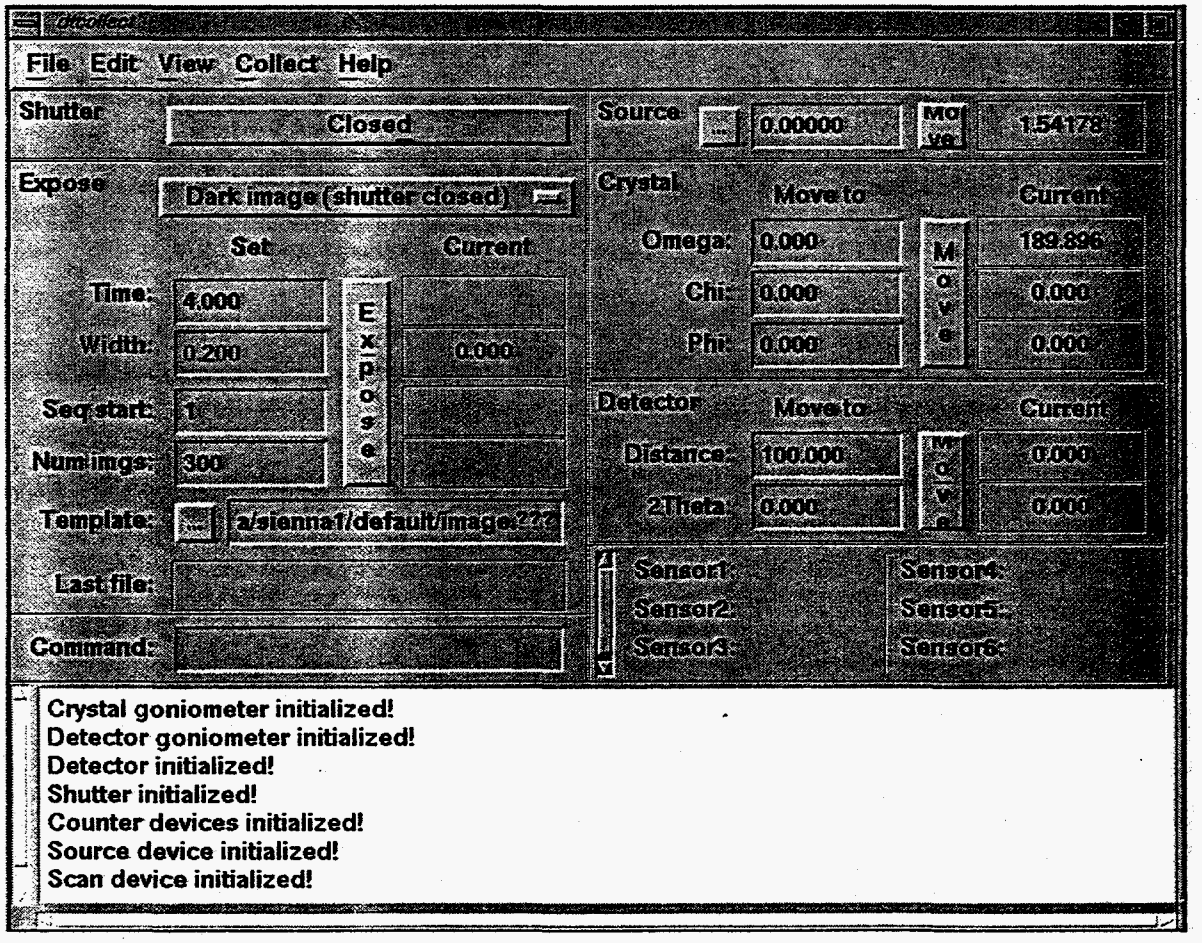

Figure 9 dtcollect graphical user interface (GUI)

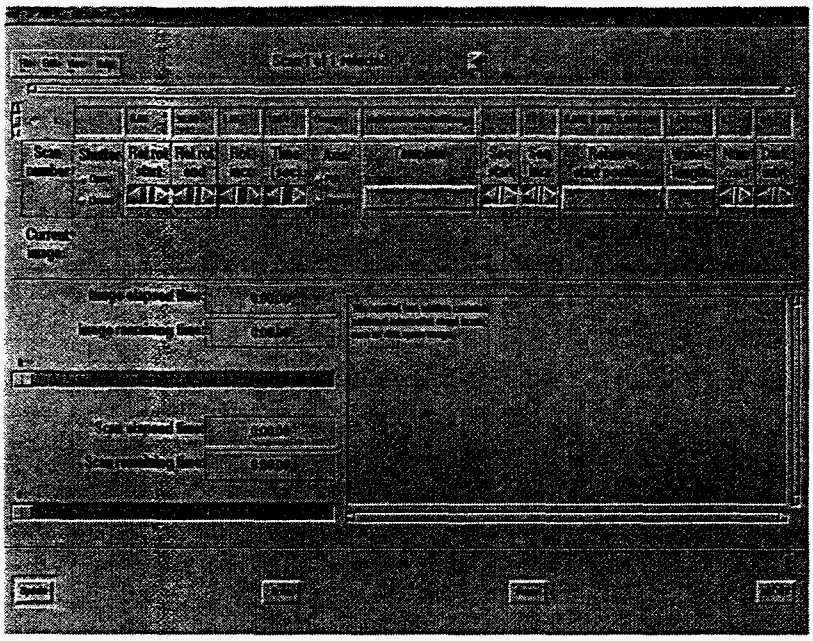

Figure 10 dtcollect scan GUI

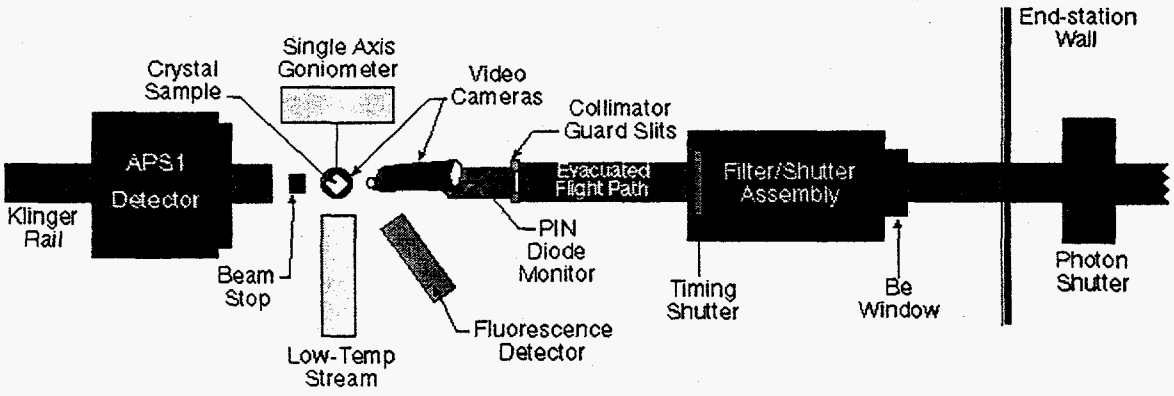

Figure 11 SBC beamline end-station instrumentation

The **dtcollect** GUI provides both context-sensitive help and hypertext markup language (HTML) help features. Whenever a user arms a button, enters a text widget, or maps a menu, **dtcollect** places a context-sensitive text string in the window title, located at the top of the **dtcollect** screen. Whenever a user activates a help command, **dtcollect** runs an external HTML viewer, such as NCSA Mosaic or Netscape, and displays HTML-formatted UNIX manual page-style help files.

2.4.2.5 *Data analysis with d*TREK*

Images collected with **dtcollect** may be analyzed with other d*TREK software modules. The data analysis tasks provided by d*TREK are performed as separate processes in the order shown in Figure 12.

*Find peaks with **dtfind**.* One of the steps performed in the data analysis of single-crystal diffraction images is the determination of the crystal properties so that the pattern of diffracted reflections can be accurately predicted. The observed centroids of a few dozen reflections with unknown Miller indices can be used to calculate the crystal cell parameters and crystal orientation, provided that the detector properties, the source properties, and the image properties are accurately known. The **dtfind** module finds reflection centroids in one or more images and writes the pixel coordinates and angular position of

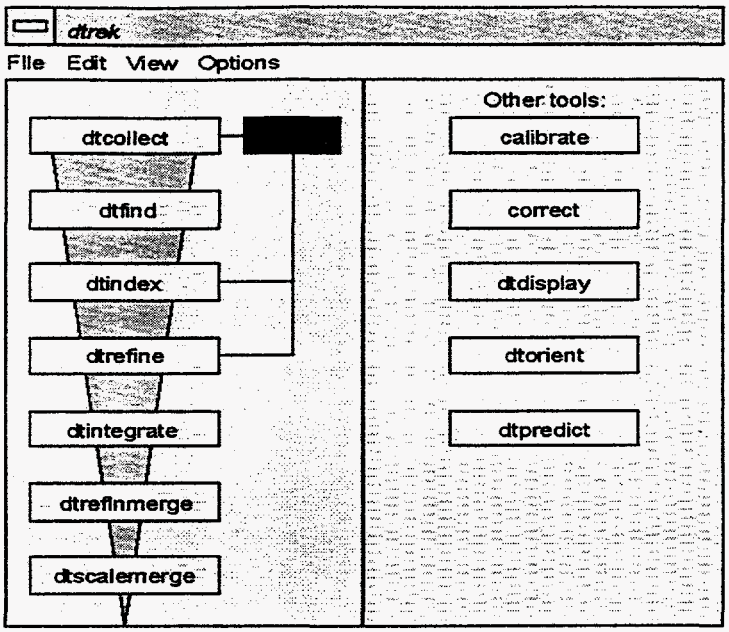

Figure 12 Early GUI showing d*TREK modules

each reflection to an output file, called a reflection file. The basic criteria used for finding peaks, is that a peak is represented by a set of contiguous pixels that are significantly above the local background. **dtfind** (Structural Biology Center, 1995b) requires an image or images to be searched for peaks. Since the spatial distortion of the detector does not vary drastically in a region as small as a reflection, the spatial distortion information is not required nor used by **dtfind.**

*Indexing with **dtindex**.* In order to integrate the intensities of Bragg reflections in a single-crystal experiment, reflection positions within the images must be predicted accurately. This can be done provided information about the source, crystal, crystal goniometer, detector, detector goniometer, and crystal rotation are known. For the predictions to be accurate, the crystal, source, and detector properties must be refined by minimizing the differences between the observed and predicted reflection positions. The radius of convergence in this refinement is limited, so approximate values of the crystal and detector properties must be obtained first. While source, goniometer, and detector properties may be physically measured, crystal properties (i.e., crystal unit cell parameters and crystal rotation) are determined by autoindexing. The **dtindex** module (Structural Biology Center, 1995c) is used to perform this autoindexing and produces unit cell parameters and crystal rotation values that may be refined with the **dtrefine** program. A reflection file (i.e., the output of **dtfind**) and an image header which describes the experiment that created the reflection file are required inputs to **dtindex.**

Developers are striving to make the **dtindex** autoindexing algorithm robust enough to successfully autoindex using a reflection list from a single diffraction image. Internally, **dtindex** uses the vectorial description (EEC, 1986; Helliwell, Machin, and Papiz, 1987) of the source, crystal, goniometers and detector objects to index the reflection list and output crystal properties. Such a vectorial description assures device independence and future upgradability by accommodating new detectors and goniometers.

*Refinement with **dtrefine**.* **dtrefine** (Structural Biology Center, 1995f) minimizes the differences between the observed and calculated reflection positions through an eigenvalue-filtered least-squares procedure. **dtrefine** refines the crystal, source, goniometer and detector parameters. Because the radius of convergence of the refinement algorithm is limited, good initial values are required for the crystal (obtained from **dtindex**), source, goniometer, and detector parameters to be refined. A reflection list with observed reflection centroids and an image header that describes the experiment that created the observed reflection list are input to **dtrefine.** If the detector spatial distortion and non-uniformity of response information in an image header requires these additional images to be read in, **dtrefine** does so automatically. The user may specify which parameters are to be refined. Final parameter values are written to a save file (see Section 2.4.2.7, d*TREK File Formats).

*Intensity Integration with **dtintegrate**.* Together **dtintegrate** and **dtprofit** (Structural Biology Center, 1996b) integrate a data scan created by **dtcollect** to produce a reflection list that contains the Miller index (*hkl*), estimated intensity, and standard deviation for the Bragg reflections that appear in the data scan. The **dtintegrate** module uses **dtpredict** to predict reflections and the same procedures as **dtrefine** to refine crystal, detector, and source properties. **dtintegrate** gets its initial information about the detector, source, and crystal properties from an image header file, which is usually the output of the **dtrefine** module. The **dtintegrate** module reads the individual images of a scan and collects three-dimensional profiles from predicted reflection positions. Every few images (defined as a batch and set by a command-line argument), the differences between observed and calculated reflection positions are minimized. The reflection profiles are written to a file which is later read by **dtprofit**. The **dtprofit** module performs three-dimensional profile analysis with the method of Kabsch (1988) and writes a reflection list file.

In more detail, integration proceeds by looping through the images in a scan. For each image, the process is as follows.

A. If this image is the last one in a "batch" of images, the crystal, detector, and source properties are refined based upon the observed positions of reflections.

B. The reflections that appear in this image are predicted and the positions of the full reflections are re-predicted.

C. For each predicted reflection:

- If this is the first time this reflection is predicted, a three-dimensional array is allocated to hold a set of pixels that would completely encompass the reflection; this 3D set of pixels is referred to as a "shoebox". The shoebox should be large enough to include pixels for a local background determination and allow for small shifts of the predicted reflection position due to crystal slippage and inaccurate prediction.
- Next, copy a window of pixels around the reflection from the image to the shoebox and check whether the 3D array is completed.
- If the array is full, then pre-process the reflection. Correct the reflection for non-uniformity of response and if the peak is strong enough, determine the observed reflection centroid and mark the reflection as full.
- Proceed to the next reflection.

D. If the image is the last one in a batch of images, then process the full reflections. Determine the local background; decide in a preliminary analysis which pixels are in background regions, which are in the peak region, and which are in neither region (i.e., belong to the peak of another reflection). Process the shoebox into 3D profiles. Write these profiles to a file for subsequent profile analysis.

E. Proceed to the next image.

When all images have been read and processed, the result is a reflection list with preliminary integrated intensities and a file containing reflection profiles. **dtprofit** is then run to do the actual profile fitting. With **dtintegrate** and **dtprofit** observed intensities are corrected for the Lorentz and polarization factors calculated by the prediction algorithm. They are not otherwise scaled. **dtreflnmerge** is used to merge the reflection lists and **dtscalemerge** is used to scale and average the reflections before they are used in any crystallographic calculations.

*Merge and filter reflection lists with **dtreflnmerge**.* Bragg reflection lists are produced in a variety of ways. For example, reflection lists are generated when **dtfind** finds peaks in diffraction images and when **dtpredict** is used to predict reflections in an image. A reflection list is also produced when diffraction images are integrated with **dtintegrate**. In each separate reflection list, different properties can be associated with each reflection (i.e., Miller index, intensity, observed position, calculated position, etc.), however, all reflections in a single list have the same properties or fields. The fields of a reflection list depend upon the purpose for which the list was created. The d*TREK reflection list format (see Section 2.4.2.7 d*TREK File Formats, Reflection Files) was developed to encapsulate all possible reflection lists that might be used in crystallographic applications.

A d*TREK reflection list is an m by n array, where m reflections have n fields, with both m and n are limited by the virtual memory of the processing engine. Many crystallographic applications manipulate reflection lists. Each different application may require the input reflection list to contain a different set of properties. Because d*TREK reflection lists can contain widely different information, the **dtreflnmerge** utility (Structural Biology Center, 1996c) was developed to manipulate (i.e., filter) reflection lists outside of the standard crystallographic applications. While the main use of **dtreflnmerge** is to merge reflection lists that result from integrating (with

dtintegrate) different data acquisition scans and to prepare a merged list for input to **dtscalemerge**, another use is in the manipulation of reflection lists. With **dtreflnmerge**, any filtering operation can be combined with merging to produce a reflection list with only the reflections and fields desired in the output reflection list. If necessary, **dtreflnmerge** reads an image header specified on the command-line to get the crystal unit cell dimensions and spacegroup, the prerequisite for certain filtering operations, such as reducing Miller indices to an asymmetric unit or calculating the resolution of reflections.

*Scale and average reflection lists with **dtscalemerge**.* In a single-crystal x-ray diffraction experiment, the intensities of many hundreds of Bragg reflections are measured. As with any scientific experiment, a single measurement for each reflection is inadequate. It is preferable to make multiple measurements of all reflections in order to assign the best estimate of intensity and standard deviation to each Bragg reflection.

When datasets of reflections collected at different exposure times, on different instruments, or from different crystals or crystal volumes are scaled together, the scaling algorithm must correct for a variety of effects:

- different crystal volumes
- radiation damage
- different absorption due to different paths through the crystal
- different detectors
- wavelength-dependent factors
- different or fluctuating source intensities

Hamilton, Rollett and Sparks (1965) published a least-squares method for determining scale factors between datasets. This method was made more robust by Fox and Holmes (1966) and serves as the basis for nearly all scaling algorithms used by crystallographers today, including d*TREK dtscalemerge. **dtscalemerge** (Structural Biology Center, 1996d) may be used to calculate and apply scale factors to a d*TREK reflection list that contains multiple batches. In the process, various statistics are calculated to help the user evaluate the quality of the input reflections and the output averaged results. With **dtscalemerge**, users can calculate and display:

- the multiplicity of observed reflections
- the number of overlapping reflections among the scaling batches
- Rmerge vs. batch
- Rmerge vs. intensity/sigma I
- Rmerge vs. resolution
- completeness vs. resolution

2.4.2.6 *Other d*TREK Tools*

*Detector calibration with **calibrate**.* Electronic position-sensitive detectors, such as the APS 1 CCD detector, generally geometrically distort the detected signal, i.e., diffraction pattern. The components of a detector, such as the phosphor, the fiber-optic tapers, and the CCD, may introduce potential problems. Detector calibration software is used to correct the deficiencies in the images introduced by the hardware. These deficiencies are spatial distortion, non-uniformity of response, defective pixels, and dark current.

The detector calibration software used at the SBC was written by M. Stanton (1992) of Brandeis University. Consult this work for more detailed information regarding detector calibration with **calibrate**.

*Image correction with **correct**.* While the d*TREK modules, such as **dtintegrate**, can perform the image corrections identified above automatically, the **correct** module converts "raw" images so that they may be processed with other data analysis software packages that require corrected images. d*TREK can process previously corrected images, as well.

*Image display with **dtdisplay**.* When users collect and process single-crystal diffraction images, it is very useful to visualize the images. In this way, a user can check that the crystal diffracts adequately and that the Bragg peaks can be integrated, i.e., the crystal is not twinned, the spot shape does not exhibit split or satellite peaks, and the spots are separated. Often by viewing the images, a user can judge whether or not to collect diffraction data from the crystal. **dtdisplay** (Structural Biology Center, 1995a) displays d*TREK-compatible images on a color X Windows display. **dtdisplay** offers a variety of features that help the user analyze images within the context of a single-crystal diffraction experiment. With **dtdisplay**, users can:

- display d*TREK images in any of eight orientations with a variety of color scales
- zoom, pan, and resize the image display
- pick spots and display pixel value, resolution, integrated intensity, and background average
- measure the d-spacing between selected spots
- display saturated pixels in selected colors

- view image headers
- plot reflection lists in selected colors
- plot resolution arcs
- tile a series of images to create a new image
- overlay a series of images to create a pseudo-oscillation image
- average a series of images to create a new image
- display images as they are collected by dtcollect
- create a PostScript™ file of the image and print it

In order to perform certain calculations, such as resolution, **dtdisplay** requires knowledge of the source, detector, and detector goniometer, which it obtains from the image header. For this reason, it is important that the image headers be accurate. The **dtdisplay** GUI shown in Figure 13 has been implemented with the X/Motif widget set and style. The Dtdisplay X Resource file allows customization of the **dtdisplay** screen.

The **dtdisplay** GUI provides the same context-sensitive help and HTML help features as **dtcollect** (see Section 2.4.2.4, Data Acquisition with d*TREK dtcollect).

*Crystal orientation with **dtorient**.* In a single-crystal diffraction experiment, the crystal is often mounted randomly on the crystal goniometer. Only after finding the centroids of a few Bragg reflections with **dtfind** and indexing them with **dtindex** is the crystal orientation known. A random crystal orientation may not be best for data collection. In combination with a three-circle goniometer, **dtorient** (Structural Biology Center, 1995d) allows the crystal to be oriented for a specific data collection strategy. At this time, SBC beamlines are equipped with single-axis goniometers and the d*TREK **dtstrategy** has not as yet been implemented (see Section 2.5). In the future, the **dtorient** module will serve as an aid in orienting a crystal for strategic data acquisition.

*Reflection prediction with **dtpredict**.* The **dtpredict** (Structural Biology Center, 1995e) module produces a list of reflections which appear on a two-dimensional position-sensitive detector for a given rotation range. The reflection list contains Miller indices (*hkl*), the detector pixel coordinates, the detector millimeter coordinates, the starting, central, and ending rotation angle values, the polarization factor, the Lorentz correction factor, the oblique incidence correction factor, and the resolution in angstroms. In order to predict a reflection list, **dtpredict** requires knowledge of the source, the crystal properties, the crystal rotation axis, the crystal rotation range, the detector, and the detector goniometer, which it obtains from an image header. Accurate prediction of reflection positions are required when integrating intensities of Bragg reflections in a single crystal diffraction experiment. **dtpredict** differs from many other prediction algorithms in that it can predict reflections for any rotation axis and detector position.

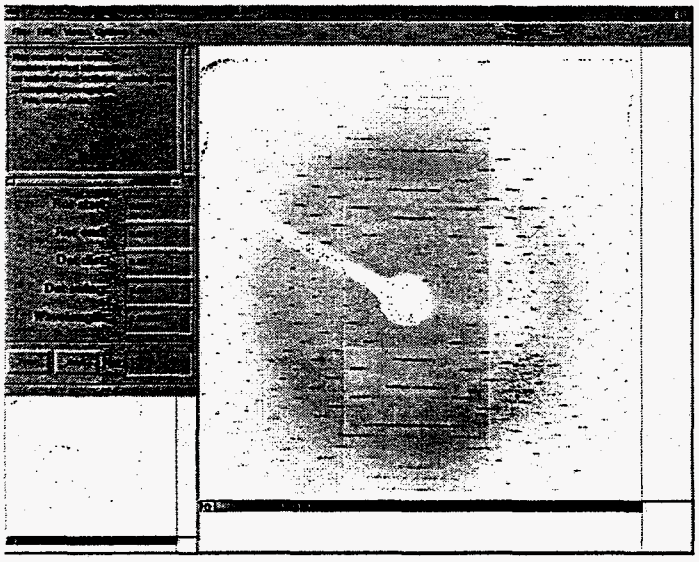

Figure 13 dtdisplay GUI

2.4.2.7 *d*TREK file formats*

Here are three types of files that d*TREK will use: image files, image header (or save) files, and reflection files.

Image Files. At the SBC, raw (i.e., uncorrected) images are written to disk. SBC image files are formatted when acquired in the MIFF format (machine-independent file format), first developed by Jon Cristy of Dupont and later modified by Jim Pflugrath and Marty Stanton for area detector images. Image files consist of an ASCII image header followed by the binary data.

The advantage of the MIFF format is that the data format can be easily determined and therefore, an image read routine used to process these images with other data analysis programs may be easily implemented. Also, the UNIX "more" command may be used to view image headers and avoid viewing binary data.

Once the crystallography community adopts a standard image format (for example, a Crystallographic Binary File format, or CBF, for image data), SBC images will be formatted in this standard format.

Image Headers (Save Files). The image header format consists of an ASCII string that has a length that is a multiple of 512 characters. The string consists of human readable text of the form KEYWORD=VALUE; where, KEYWORD is a case-sensitive string of up to 32 characters. VALUE is the value of the KEYWORD, which may be a number (integer or decimal), a string, or an array of numbers. There are a few required KEYWORDS that are used to aid in reading the data:

HEADER_BYTES, length of the header in bytes,

DIM, the number of dimensions in the image (usually two),

SIZE1, the number of pixels along the first direction,

SIZE2, the number of pixels along the second direction,

TYPE, image format identifier (MAD for APS 1 CCD detector images),

BYTE_ORDER, the byte order of the data,

DATA_TYPE, the data type identifier.

Save files are exactly equivalent to image header files and to image files without binary data.

Reflection Files. Reflection files are ASCII files (i.e., they contain no binary data) and may be edited with a text editor. The first line of a reflection file consists of three numbers designating the number of integer, decimal, and string fields for each reflection. Field labels as character strings, one per line, follow the first line of a reflection file. The first three fields of the reflection file are *h*, *k*, and *l*. The first two decimal fields are Intensity and Sigma I. All other fields are optional. The reflection data, one reflection per line in free format, follows the field labels. This format is a prelude to using a CBF reflection file format.

2.4.2.8 *d*TREK licensing*

The Department of Energy (DOE), specifically Argonne's Structural Biology Center, has contracted with Molecular Structure Corporation for the technical services required to develop the d*TREK software package. While the SBC owns the source code developed under this contract, the SBC has agreed not to distribute the source code. The SBC will distribute executables to users for the purpose of processing data collected at SBC beamlines at their home institutions. MSC is scheduled to begin commercial distribution d*TREK at the end of calendar year 1996.

2.4.3. *d*TREK - SBC Control System interface*

The d*TREK was developed outside of the Structural Biology Center, in parallel with the control system software. To facilitate this, complete specifications (Structural Biology Center, 1994b) for d*TREK development were defined and used as the basis for the contract between the SBC and MSC. Responsibility for the device-independent C++ programs for analyzing and collecting data were assigned to MSC, while responsibility for support of the device-specific monitor and control functions were assigned to SBC developers who had ready access to the hardware and instrumentation (Figure 14).

d*TREK and the SBC Control System interface in the data acquisition program, **dtcollect**. To interface the device-independent **dtcollect** to the device-specific SBC Control System, MSC and SBC developers collaborated in the specification of an application programming interface (API) that consists of six C++ classes: CDevScan, CDevGoniom, CDevDetector, CDevCounter, CDevShutter, and CDevSource (Structural Biology Center, 1995i).

At the SBC, the API was implemented by providing classes to interface with the EPICS control system. Methods were provided to control and monitor the APS 1 CCD detector, the SBC goniometer, ion chamber, PIN Diode detector, double crystal monochromator, and fast x-ray shutter. The methods in the SBC API implementation

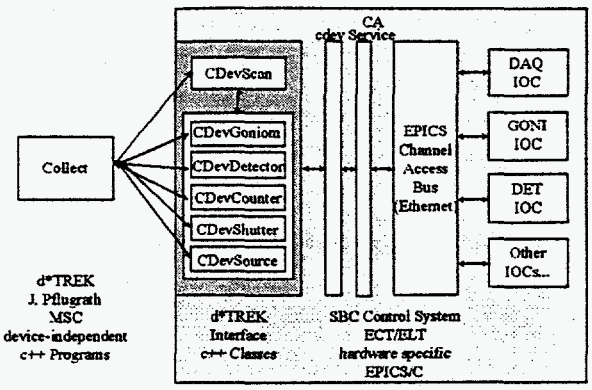

Figure 14 d*TREK interface to the SBC Control System

used the cdev (Common DEVice) C++ class library developed at CEBAF. Cdev provides a standard interface between the class methods (or any other application) and the EPICS-based control system.

Data acquisition of processible x-ray diffraction images demonstrates the successful integration of d*TREK and the SBC Control System. This API has been successfully used at the SBC and has since been successfully applied to a Windows NT version of the software by MSC for use with a commercial CCD detector system.

2.4.4. *SBC Control System user interface*

The SBC user interface allows users and developers to control and monitor beamline and end-station instrumentation through a variety of control system screens, or control screens. These control screens have been built using the EPICS Motif Editor and Display Manager (MEDM) tool.

One example control screen is shown in Figure 15, which shows the user interface to the fast x-ray shutter and attenuator assembly. With this control screen, the user can easily monitor shutter status and with one-button can control the shutter (one-button control of the attenuators as well).

With over 500 individual control screens developed for beamline commissioning, developers needed a convenient way to gain access to each screen. For this reason, SBC developers implemented a toolbox-style interface, called the SBC Panel (Figure 16) with tcl/tk. The SBC Panel provides a cascading menu selection organized according to function, which allows direct access to every control screen. This toolbox-style interface provides a versatile and easily modified user interface suited to the needs of beamline commissioning.

As users have begun to participate in beamline commissioning, SBC developers recognize the need for a high level interface tailored explicitly to user tasks at a synchrotron beamline. With the SBC Panel, users find that multiple control system screens are required to conduct routine tasks and that the level of detail on these screens can be distracting and confusing. While the SBC Panel interface may not be the answer to conducting routine user tasks, it will continue to facilitate software development, any required troubleshooting, and will provide "expert-mode" beamline operation capability.

To address the need for a high-level user interface, SBC developers have followed a user task-oriented iterative design process (Springmeyer, Blattner, & Max, 1992) shown in Figure 17. User surveys have been conducted on the Internet via the Structural Biology Center Web Site (http://www.sbc.anl.gov/intro.html). The information gained from these surveys, combined with SBC facility requirements, has lead to prototype interfaces (see Figure 18) that are being evaluated by users via the Internet and at SBC beamlines as well. These results are driving further user interface development and enhancements.

2.5. *Future Control System Research and Development*

The many control system research and development initiatives that can be undertaken at the SBC fall into two major categories: automation and SBC operations.

Further research and development in the area of overall automation may be conducted at the SBC:

- use of supercomputers for real-time data analysis and visualization in a transparent manner (see Figure 4)
- development of fast (parallel) algorithms for data analysis
- development of direct methods for solving the phase problem for direct phase solution of atomic resolution macromolecular crystallographic data

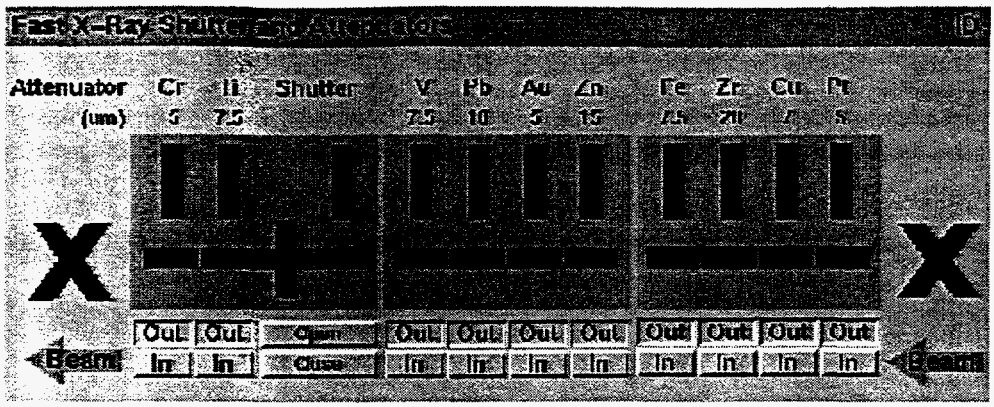

Figure 15 SBC x-ray shutter user interface

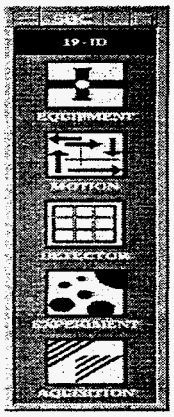

Figure 16 SBC panel

- beamline automation (task-specific push-button control)
- development of expert systems (Anonymous, 1966a; Anonymous, 1996b) which will provide for automatic data acquisition, auto-data analysis, auto-phasing, auto-electron density map fitting, auto-refinement, and auto-difference patterson map interpolation

To facilitate SBC operations, further work is required in the following areas:

- user help and training documentation and programs
- implementation of a User Information Database
- continued development and enhancements of the SBC user interface
- provision for user access to data via high-speed network links from their home institutions
- installation of other available data analysis programs on-site
- installation of MAD data analysis software on-site.
- implementation of collision detection and prevention
- upgrade of d*TREK programs to read and write images in the CBF format
- implementation of a d*TREK dtstrategy program to aid in strategic data acquisition
- implementation of an electronic notebook feature for tracking of beamline operation and experimentation
- automatic detector parameter adjustment program (offset, gain, etc.)

3. Summary and Conclusion

The SBC Control System was not developed in a vacuum; rather, it is based upon user input and has been subject to external review. SBC Control System development has also benefited from the implementation techniques used and the wealth of device support built into EPICS.

The SBC Control System is designed to be at the state of the art, providing convenient and effective beamline control, data acquisition, and data analysis; it should enhance the

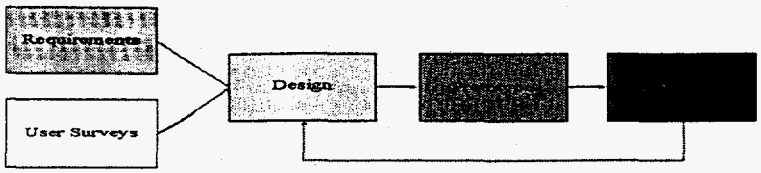

Figure 17 SBC Control System GUI Design

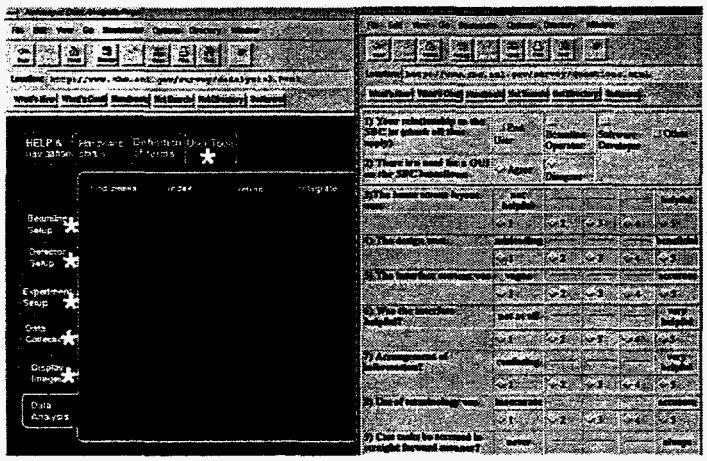

Figure 18 Prototype interface and survey on the WWW

productivity of both SBC staff and users. The APS promises user beam availability 250 to 300 days per year, 24 hours per day in its mature operation. At the SBC, users should be able to obtain high-resolution datasets (100 micrometer nominal crystal size) in 0.5 to 1 h. Therefore, the implication is that the SBC will be extraordinarily productive.

The SBC is the first user facility of its kind to apply high-performance networking along with symmetric multi-processing computing to macromolecular crystallography. The challenge of rapid data acquisition has been met. User intervention has become a rate-limiting step: how quickly can the user get the sample mounted? In addition, data processing may limit the speed of data acquisition, and we intend to focus attention on this problem in our future work.

Acknowledgments

Structural Biology Center construction, which includes all the work presented in this paper, has been funded by the U. S. Department of Energy, Office of Health and Environmental Research.

SBC Control System hardware and software development has been conducted by the SBC Computing Systems and Software Engineering Group at Argonne: Tom Coleman, Bob Daly, John Weizeorick, and Mary Westbrook. Control electronics, such as the data acquisition synchronization module, have been designed and fabricated by Joe Haumann. The APS 1 CCD Detector was designed and built by Istvan Naday.

Special thanks to Marty Kroll for his help with computing and network installations and to Mike Mohtsky for his help with control electronics installations.

The development of the d*TREK Data Analysis software has been carried out by Jim Pflugrath at Molecular Structure Corporation.

The SBC user interface has been designed and implemented in collaboration with Christina Vasilakis, Electronic Visualization Laboratory, Department of Electrical Engineering and Computer Science, University of Illinois at Chicago.

References

Anonymous, 1996. Web site at "http://pasture.ecn.purdue.edu/~agenhtml/agen565/es.htm".

Anonymous, 1996. Web site at "http://winnie.cs.nott.ac.uk/aim/expert/expert.htm".

Barford, D., Flint, A. L. & Tonks, N. K. (1994). Science 263, 1397-1404.

Bricogne, G. (1986). In: Proceedings of the EEC Cooperative Workshop on Position-Sensitive Detector Software III, 65.

Dalesio, L. R., Kozubal, A. J. & Kraimer, A. J. (1991). In: EPICS Architecture, Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems, edited by C. O. Pak, S. Kurokawa, & T. Katch, pp. 278-282. Tsukuba, Japan: National Laboratory for High Energy Physics, KEK Proceedings 92-15.

Ealick, S. E. & Walter, R. L. (1993). Current Opinion in Structural Biology 3, 725-736.

EEC (1986). Proceedings of the EEC Cooperative Workshop on Position-Sensitive Detector Software, Phases I, II, and III.

Fox, G. C. & Holmes, K. C. (1966). Acta Cryst. 20, 886-891.

Hamilton, W. C., Rollett, J. S. & Sparks, R. A. (1965). Acta Cryst. 18, 129.

Helliwell, J. R., Machin, P. A. & Papiz, M. Z., eds. (1987). Computational Aspects of Protein Crystal Data Analysis, SERC Daresbury Laboratory, U.K.

Humphrey, W. S. (1989). Managing the Software Process, Redding, MA: Addison-Wesley.

Kabsch, W. (1988). J. Appl. Cryst. 21, 916-924.

Lewis, M., Chang, G., Horton, N. C., Kercher, M. A., Pace, H. C., Schumacher, M. A., Brennan, R. G. & Lu, P. (1996). Science 271, 1247-1254.

Messerschmidt, A. & Pflugrath, J.W. (1987). J. Appl. Cryst. 20, 306.

Pflugrath, J. (1996). In: Methods in Enzymology, Vol. 276, edited by C. W. Carter, Jr. & R. M. Sweet, San Diego: Academic Press. In the press.

Smith, J. L. & Watenpaugh, K. D. (1991). The Biosync Workshop Report, Department of Biological Sciences, Purdue University, West Layfayette, IN 47907.

Springmeyer, R. R., Blattner, M. M. & Max, N. L. (1992). In: Proceedings IEEE Visualization '92, pp. 235-242.

Stanton, M. (1992). J. Appl. Cryst. 25, 549-558.

Structural Biology Center (1994a). Structural Biology Center Control System Requirements, Argonne National Laboratory document SBC01-0007.

Structural Biology Center (1994b). Structural Biology Center Macromolecular Crystallographic Data Analysis Software Statement of Work, Argonne National Laboratory document SBC01-0004.

Structural Biology Center (1994c). Structural Biology Center Software Development Plan, Argonne National Laboratory document SBC01-0008.

Structural Biology Center (1995a). d*TREK dtdisplay User Documentation, Argonne National Laboratory document SBC01-0200.

Structural Biology Center (1995b). d*TREK dtfind User Documentation, Argonne National Laboratory document SBC01-0202.

Structural Biology Center (1995c). d*TREK dtindex User Documentation, Argonne National Laboratory document SBC01-0197.

Structural Biology Center (1995d). d*TREK dtorient User Documentation, Argonne National Laboratory document SBC01-0199.

Structural Biology Center (1995e). d*TREK dtpredict User Documentation, Argonne National Laboratory document SBC01-0201.

Structural Biology Center (1995f). d*TREK dtrefine User Documentation, Argonne National Laboratory document SBC01-0198.

Structural Biology Center (1995g). Structural Biology Center Control System Architecture, Argonne National Laboratory document SBC01-0116.

Structural Biology Center (1995h). Structural Biology Center Data Analysis Software Design, Argonne National Laboratory document SBC01-0131.

Structural Biology Center (1995i). Structural Biology Center d*TREK EPICS Interface Summary, Argonne National Laboratory document SBC01-0216.

Structural Biology Center (1995j). Structural Biology Center Software Configuration Management Plan, Argonne National Laboratory document SBC01-0153.

Structural Biology Center (1995k). Structural Biology Center Systems and Network Design, Argonne National Laboratory document SBC01-0117.

Structural Biology Center (1996a). d*TREK dtcollect User Documentation, Argonne National Laboratory document SBC01-0256.

Structural Biology Center (1996b). d*TREK dtintegrate/ dtprofit User Documentation, Argonne National Laboratory document SBC01-0261.

Structural Biology Center (1996c). d*TREK dtreflnmerge User Documentation, Argonne National Laboratory document SBC01-0270.

Structural Biology Center (1996d). d*TREK dtscalemerge User Documentation, Argonne National Laboratory document SBC01-0271.

Westbrook, E. (1996). In Methods in Enzymology, Vol. 276, edited by C. W. Carter, Jr. & R. M. Sweet, San Diego: Academic Press. In the press.