JP0150576

JAERI-Conf 2001-002

ICANS-XV
15th Meeting of the International Collaboration on Advanced Neutron Sources
November 6-9, 2000
Tsukuba, Japan

16.3
New Data Acquisition System for the Lujan Center

R. Nelson[1*], P. S. Bowling[1], G. M. Cooper[1], and T. Kozlowski[1]
[1]Los Alamos National Laboratory, Los Alamos, NM 87544, USA
*E-mail: ron@lanl.gov

Abstract

To meet the data acquisition requirements for six new neutron scattering instruments at the Los Alamos Neutron Science Center (LANSCE), we are building systems using Web tools, commercial hardware and software, software developed by the controls community, and custom hardware developed by the neutron scattering community. To service these new instruments as well as seven existing instruments, our data acquisition system needs common software and hardware core capabilities and the means to flexibly integrate them while differentiating the needs of the diverse instrument suite. Neutron events are captured and processed in VXI modules while controls for sample environment and beam line setup are processed with PCs. Typically users access the system through web browsers.

1. Introduction

At LANSCE we are currently designing and building six new neutron scattering instruments. While each has common requirements for detector support, histogram building, archival data storage, integrated event counting, real-time data analysis, and slow controls, these instruments also simultaneously exhibit widely different detector and sample environments. Accordingly the data acquisition system needs common software and hardware capabilities while retaining sufficient flexibility to accommodate different requirements. For example the numbers of neutron detectors will vary from a single position-sensitive area detector to 1500 tube detectors. Depending upon the instrument characteristics average count rates for a bank of detectors will range from 10^2 to 10^6 events/s and measurement times will vary from minutes to days.

Generally we select the most demanding requirements as the common standards for all instruments. Thus measured parameters will include time-of-flight, position, and pulse-height information. Storage for histograms for a single instrument will initially be as large as 1 GB and will increase as the capabilities of processor modules expand. Experiments may continuously monitor portions of the histogram space and at completion of counting will record the full histogram in a NeXus file format [1] in no more than 15 s.

The system will have three major functional or core components: the user interface for monitoring and control (web server and browser), the real-time processes (histogramming, control of the sample environment, etc.), and the run manager for sequencing controls and measurements. To maximize software reuse and minimize cost we want to exploit commercial products and the work of physics collaborations. Thus we have chosen generic Web browsers, VxWorks plus the Experimental Physics and Industrial Control System (EPICS) [2], and WindowsNT for the software foundations to implement these three functional components, respectively. The data acquisition hardware configuration, shown in Figure 1, consists of custom VXI modules to capture data from detectors that view the pulse of neutrons, one or more microprocessors to handle the real-time event processing and

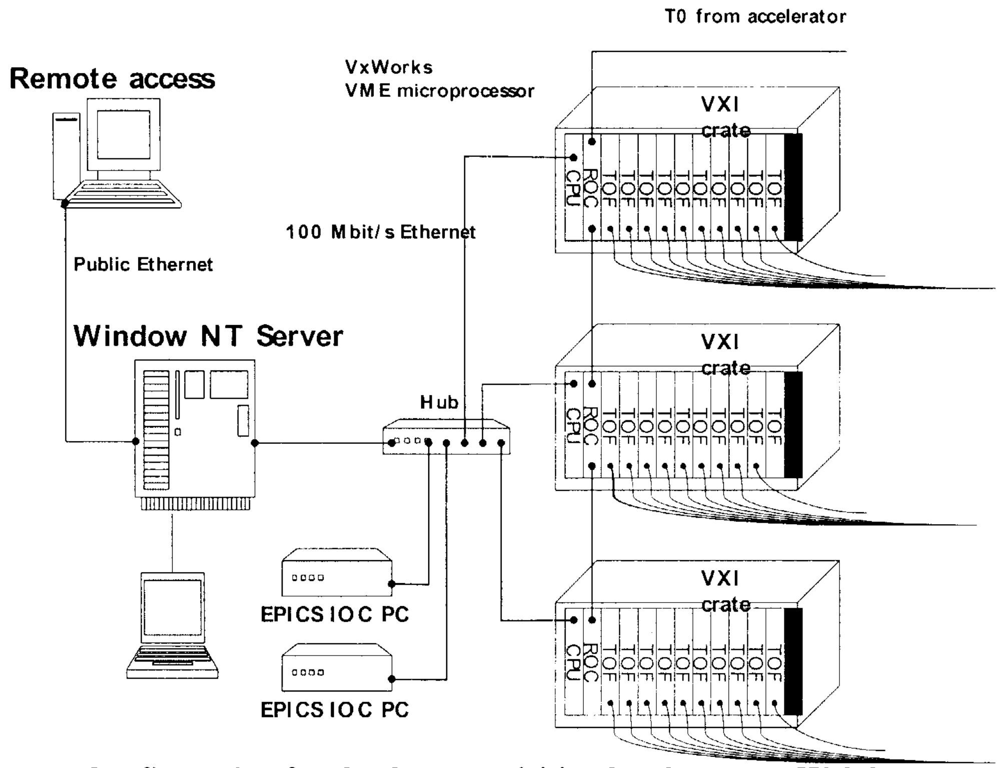

Figure 1. Configuration for the data acquisition hardware. A Web browser on the remote access machine provides the user interface. The real-time system components include the VXI crates and the EPICS IOCs. The NT server offers run control, monitoring, and Web publishing services.

histogram construction, one PC to handle the user interface, and one PC to function as a Web server and run controller. Following collaborative efforts with the Intense Pulsed Neutron Source, we have programmed a flexible Argonne VXI module [3] to time stamp and buffer events. A private network connects the readout controllers to the data acquisition server. A second public network, the Internet, connects users from any remote location to the data acquisition server.

2. Functions of the core components

The real-time system interfaces to signals from the detectors, time stamps each neutron event, collects all the events from a single pulse of neutrons in a "frame," and runs a user-specified software routine that builds histograms in memory. Once initialized, the real-time system operates autonomously and continues to accumulate events in its histograms until the user manually intervenes or the data acquisition server automatically signals the real-time system that sufficient data have been accumulated. For monitoring data, the real-time system provides snapshots of requested histograms. It also controls the sample environment and neutron beam delivery.

The GUI provides a friendly interface for control of the experiment. Although the appearance of the GUI will vary for each instrument, the overall "look and feel" will be designed to be similar. The experimenter uses the GUI to create a new run configuration, edit a run configuration, initialize hardware and software, control the sample environment, start data collection, end data collection, and monitor environment status and histograms of interest.

The data acquisition server, which is the Web server and run controller, provides services that control execution of the experiment. At the start of an experiment the experimenter prepares a database that defines all the parameters required to configure the instrument and sample environments prior to data collection. The data acquisition server controls all "run" sequencing. When a run is complete, the data acquisition server constructs an Hierarchical Data File (HDF) [4], the public domain file format from the National Center for Supercomputing Applications, and transfers the file to a data archive file server. The HDF is laid out according to the NeXus standard as defined by the neutron scattering community.

3. Implementation

A. Real-time System

The real-time system is highly configurable and scaleable. As shown in Figure 1, each VXI crate holds a VME CPU processor connected to the VXI backplane through a 180-mm extender. Although the basic building blocks for each VXI crate are the same, each instrument can be configured to meet its individual requirements. Plug-in daughter boards on the TOF modules provide variations to service either single-ended or linear position-sensitive ^{3}He tubes. The TOFA, a variant of the TOF, serves two-dimensional area-sensitive ^{3}He detectors. Electronics at the front end of these three models of TOF modules will accommodate the different detector technologies, but the back end including time stamping, buffering, vetoing, and readout are essentially identical. The TOF modules service 16, 8, and 1 of the single-ended, position-sensitive, and area-sensitive ^{3}He detectors, respectively. The number of crates and TOF modules per crate are determined by the requirements of the instrument. Note that the TOF modules are implemented in a VXI form factor but use the VME bus protocols.

Each VXI crate will normally be configured with one VME processor module to serve as the readout controller. If additional processing power is required, a second processor may be added to the VME module. For even more processing additional processor modules may be added to the crate at the expense of the TOF modules.

Each crate also has one read-out controller (ROC) module. The ROC performs such ancillary functions as distribute T0 pulses to the various TOF modules and optionally to other crates, flow control, and test pulses for detector-electronics calibration or verification.

Starting with the TOF module as designed and built at Argonne, we have modified only the field programmable gate arrays (FPGAs) to accommodate our requirements. We record 64-bit events in the FIFO where each event consists of 24 bits of time, 16 bits of channel identification, and pulse-height information, either 8 bits for singled-ended tubes or 16 bits for position sensitive tubes (8 bits for each end). Unused bits are set to zero.

The performance of the network that links the front-end VME processors with the data acquisition server will be determined by the count rate and readout requirements of the instrument. Often this network need not be better than the 100 Mb/s (megabits/s) Ethernet in use today. Future performance can be boosted with the addition of a PMC with a 1 Gb/s Ethernet controller on the processor board.

Software for the real-time processors will be based on the commercial operating system, VxWorks, from Wind River [5], which is widely used through the nuclear physics, high-energy physics, and instrumentation communities. Using different board support packages, VxWorks is available for virtually every commercial VME processor module in the market thus providing yet another dimension of flexibility. Early implementations of the data acquisition system may be easily upgraded as more powerful microprocessors are designed into VME boards.

In each processor the experimenter may choose from a set of parameterized histogramming algorithms or may provide a custom algorithm. The programming language for the algorithm may be FORTRAN or C. The data acquisition system will provide a standard environment and a library of functions to facilitate development of custom histogramming algorithms.

Most LANSCE instruments will use PCs running LabView [6] to control external equipment necessary to regulate environmental parameters such as temperature, goniometer position, slit position, *etc*. Through an interface to channel access, these PCs will act as EPICS IOCs (input/output controllers) [2].

B. Graphical User Interface

Generally the GUI for the data acquisition system is simply a commercial Java-enabled Web browser available from several vendors. Should the network graphics performance be inadequate for some applications, then experimenters may use the data acquisition server directly. Of course the advantage of the browser interface is its extreme accessibility, i.e. experimenters may control the measurement from anywhere on the Internet. The benefits of this approach include experiment control from home or while on travel and, with some coordination from the Lujan Center personnel, remote control and monitoring from facilities worldwide.

Web access must be regulated. We cannot allow malicious interference from hackers on the Internet or even inadvertent interference from other experimenters to jeopardize the integrity of measurements or the safety of an instrument. For example, SMARTS is a $5M instrument with moving parts capable of crushing a person. Therefore to protect our investment and to operate safely, security measures must be an integral part of our Web interface. Currently the technology for secure Web transactions is rapidly evolving, and we believe it will offer a mature solution before the new data acquisition system goes into production.

We are using the Java language to implement the GUI interfaces that we develop for each instrument. Numerous "drag-and-drop" GUI-builders are now available to construct a significant portion of the Java code. Since the Java code executes in the Java Virtual Machine within the Web browser, the code is automatically portable to all hardware platforms and operating systems offering a Java-enabled browser.

C. Data Acquisition Server

The data acquisition server is a commercial PC that runs WindowsNT. As shown in Figure 1, the hardware must be configured with two network adapters, one for the private high-performance network for the real-time systems and the other for the public Internet. The file archive, where the data are stored in NeXus files for all instruments in our facility, is accessible through the Internet.

The data acquisition server is the center of the software system that provides run control and monitoring for an instrument. Through software interfaces the experimenter defines a database that contains all the information required to configure the instrument for the desired measurement. Some of this information specifies how the histograms are organized for this measurement, how the time is compressed, how channel numbers relate to physical parameters, what are the sample parameters, etc. Browser tools facilitate copying, editing, and viewing such databases.

Through a browser a user can also initiate a measurement defined by a specified database and stored on the data acquisition server. At this point the data acquisition server must initialize the real-time hardware and environmental parameters. Thereafter the work is performed by the real-time system. To monitor the experiment a user may view histograms that are accumulating in the processor's memory. Read requests from the user for 'live' data are communicated through the browser to the data acquisition server, and the data acquisition server reads the content of the relevant memory from the VME processor and develops the image of the display. This image is then returned to the browser.

As data accumulation progresses, either the user manually or the real-time system in an automated mode recognizes that sufficient data have been counted and signals the data acquisition server that data collection should be terminated. At the discretion of the experimenter the histograms and contents of the database are formatted into a NeXus file and written to the local disk. A duplicate of the NeXus file is also exported to the data archive located elsewhere on the network. For near-real-time analysis, the experimenter may choose to write an analysis input file that is a subset of all the information included in the NeXus file.

Through use of scripts, the data acquisition server may be configured for automatic sequencing of measurements. With this capability a series of measurements can be set up to run in the absence of error conditions without manual intervention.

4. Software

The new data acquisition system software for neutron scattering instruments is being developed using formal object oriented analysis and design methods (Shlaer-Mellor OOA/RD) in conjunction with automatic code generation. The use of rigorous models with code generation guarantees that requirements and design models are in step with the code and allows many development and maintenance activities to take place at a relatively high level of abstraction free of concern for the implementation details of a particular coding language.

The Shlaer-Mellor OOA and Recursive Design method [7], [8] partitions a system into independent subject matter domains, some of which can be pre-existing, some can be created by normal hand coding, and some implemented by code generated from formal models. A system typically has one Application Domain, a set of Service Domains, a set of pre-existing Implementation Domains, and a special Architecture Domain that defines the mapping of domains to implementation artifacts. A Domain Chart and an accompanying description document are created to precisely define the domains that constitute a system and their relations.

Figure 2 shows the Domain Chart for the LANSCE data acquisition system. The relation between domains is defined in terms of "bridges", which are represented on a Domain Chart

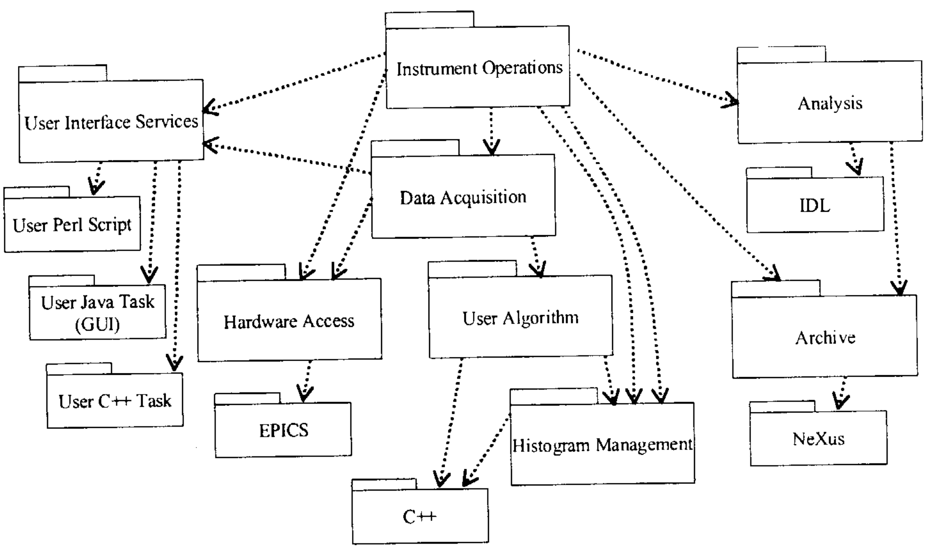

Figure 2. Data acquisition system Domain Chart.

as arrows between domains. The arrows do not depict data flow. In fact, data typically flows in both directions across a bridge. The arrow represents a dependency or "requirements flow". It can be read as: “the (client) domain assumes a (service) domain exists in order to ...". A Domain is the scope of a Shlaer-Mellor model (if we chose to build a formal model).

Since most real-world systems incorporate pre-existing subsystems or subsystems that are implemented with "traditional" coding methods, the method must accommodate them. The domain concept provides an easy solution, since by definition one domain is completely unaware of the internal structure of the other domains it communicates with. Bridges provide the framework for integrating non-modeled domains. It is a convention that the implementation of the service domain incorporates the bridge implementation for all client domains.

The Domain Chart in Figure 2 shows the major components of the first release of the LANSCE data acquisition system. The major functional components, organized as Application and Service Domains are:

- Instrument Operations (Application Domain): neutron scattering instrument configuration, initialization and run control from the point of view of the scientist/operator.
- Data Acquisition: readout of instrument data acquisition hardware and invocation of user provided “user algorithm” software for processing and histogramming (typically distributed over multiple VXI crates).
- User Interface Services: support for user interaction with the system, including handling of requests and responses for multiple user sessions, including scripts, guis, and spawning of user applications in response to system occurrences. The first release supports only a single GUI application.
- User Algorithm: user supplied event processing software that runs in the VXI crates, but must obtain instrument parameters from the Instrument Operations domain.
- Histogram Management: provides services for allocating, incrementing, and retrieving histograms defined in the Instrument Operations domain and distributed across all VXI crates.
- Archive: service for archiving histograms and all relevant instrument parameters during and at the end of a run into a permanent archive.
- Analysis: instrument specific algorithms for reducing the data and extracting measured parameters.
- Hardware Access: services for accessing system hardware, including data acquisition hardware and ancillary system hardware (choppers, high voltage, sample changers, etc.). Much of the latter will be accessed via EPICS; however, the first release does not support ancillary systems.

So far we have implemented the first three domains for formal OOA modeling, since it was especially useful to capture and maintain the complex requirements of the instrument and data acquisition components using the formalism. We used “traditional” code development techniques for the other domains, since we had experience implementing similar subsystems and also we wanted to confine the formal modeling to a subset of the entire system since this was our first experience with the formal techniques.

The DAQ software is now in use on three systems. Release 2.0 integrates Windows NT and VxWorks systems with simple GUI, user algorithm, and NeXus file generation capabilities. While the performance of 10^4 events/s/crate is not adequate for production, most of the essential functions are present and working. As yet we have not optimized the code for performance or integrated slow controls.

5. Conclusions

The architecture identified in this paper provides modular software and hardware for data acquisition that enables construction of extensible systems for all neutron scattering instruments at LANSCE. Using commercial and public domain software and hardware components, our implementation extensively exploits the work done by others. Furthermore the novel use of Web-based servers and browsers permits unparalleled flexibility for the remote conduct of experiments perhaps even eliminating the necessity of travel for some experiments.

We have used object oriented analysis methods to automate generation of source code. We find this approach clarifies the function and relationships of system components. We have successfully applied the methodology to both the Windows NT server and the processors in the VXI crates running VxWorks.

References

[1] Web URL http://www.neutron.anl.gov/NeXus
[2] Web URL http://www.aps.anl.gov/asd/controls/epics/ EpicsDocumentation/WWWPages/
[3] Kent Crawford, Joe Haumann, John Weizeorick, Argonne National Laboratory, private communication.
[4] Web URL http://hdf.ncsa.uiuc.edu
[5] Web URL http://www.wrs.com
[6] Web URL http://www.ni.com
[7] Sally Shlaer and Stephen J. Mellor, Object Oriented Systems Analysis, Modeling the World in Data, Englewood Cliffs, New Jersey: Yourdon Press, 1988.
[8] Sally Shlaer and Stephen J. Mellor, Object Oriented Systems Analysis, Modeling the World in States, Englewood Cliffs, New Jersey: Yourdon Press, 1992.