Software Design for a Database Driven System for Accelerator Magnet Measurements

B. C. Brown, M. E. Bleadon, H. D. Glass, R. Glosson, R. W. Hanft, D. J. Harding, P. O. Mazur, J. E. Pachnik, J. W. Sim, K. Trombly-Freytag, and D. G. Walbridge
Fermi National Accelerator Laboratory *
P.O. Box 500
Batavia, Illinois 60510

Abstract

Measurements of more than 1000 new magnets are needed for the Main Injector Project at Fermilab. In order to achieve efficiency and accuracy in measurements, we chose a database driven design for control of the measurement system. We will use a relational database to describe the measurement subjects and equipment. A logbook system defined in the database will provide for prescription of measurements to be carried out, description of measurements as they are carried out, and a comment database for less structured information. The operator interface will be built on X-windows. This paper will describe our system design.

1 Introduction

The Magnet Test Facility (MTF) at Fermilab must do production quality control testing involving many magnets of a given series, occasional measurements of small numbers of other magnets, and R&D measurements. The facility can test both conventional and superconducting magnets[1]. The conventional magnets for the new Main Injector will require a variety of different magnetic field measurements. The following is a partial list of the required techniques:

- Using a Hall or NMR probe to measure static magnetic fields as a function of position within the field.
- Ramping the magnet current while measuring the voltage induced in a stationary coil.
- Rotating or translating a coil in a static magnetic field while measuring the voltage induced in the coil.

Development is underway for a new measurement system using VME-VXI hardware in a UNIX software environment. It will achieve improved electronic sensitivity and will accommodate additional mechanical and probe systems. Software for an existing system [2] has individual software programs to perform each type of measurement. It is a semi-automatic system with measurers choosing the measurement sequences in ways specific to each program. For the new system, we are designing a unified software approach for all of the above measurements.

2 Requirements

Attributes specified for the design include the ability to

- record the test conditions precisely
- identify which test hardware is used for measurements and record what calibration values are used for analysis
- change hardware modules easily, while maintaining required calibration information records
- specify and record production test sequences easily
- record results of testing in easily accessed databases
- perform non-standard measurements and/or record special situations
- record measurement assessments and commentary from measurement, supervisory and analysis personnel on any aspect of the measurement process in an easily correlated form

The measurements to be performed will be selected by a Test Manager and executed by a Measurer who can order selections from the prescribed measurements. The measurer will establish the mechanical test configuration, initiate the testing, carry out any manual operations, and provide initial data evaluation.

3 Computing Environment

The data acquisition hardware will consist of a Concurrent 6400 Computer[1] with external VME and VXI crates. This data acquisition computer will be networked with Sun SPARC computers[2] on which display and database software will be operated. Most control hardware, including motor control and power supply control modules, will reside in the VME crate. Most data acquisition hardware, including DVM's, ADC's, probe interfaces, and flux readout systems, will reside in the VXI crate. Commercially

*Operated by the Universities Research Association under contract with the U. S. Department of Energy

[1] Concurrent Computer Corporation, 106 Apple Street, Tinton Falls, New Jersey, 07724.

[2] Sun Microsystems, Inc. 2550 Garcia Avenue, Mountain View, CA, 94043

available modules will be supplemented with a few special modules built for our use.

Several commercial software products form the basis of the measurement system software environment. Data storage, for both measurement control and data recording, will be accomplished through the use of the Sybase relational database management system[3]. Sybase also provides the tools and libraries which facilitate database access. The measurement system relies on a network to distribute resources. Remote Procedure Calls (RPC's) will tie the system together for data access and perhaps eventually for distributed measurement control. Sun RPC's will be used on our two computer platforms.

Most program execution will take place within the X Window environment using the Motif graphical user interface.[4]

4 Measurement System Database Structure

The measurement system database is constructed of a set of databases in which separate functions are maintained. These fall into three broad categories:

- measurement **facilities**
- test **subjects** and results
- measurement **logbook**

This paper focuses on the implementation of the measurement logbook and programs using it.

In the logbook, measurements are prescribed using checklists while measurement progress is recorded in an activities log. Together these allow us to prescribe and describe all routine successful measurements. Comments are recorded in the logbook to allow personnel to record non-standard measurements, unusual measurement activities, or descriptions of unexpected measurement results. The **logbook** database is intended to provide easy access both for casual queries and reports and for measurement and analysis programs.

The activities log is implemented as a pair of database tables. The primary table (**activities_log**) provides a time-stamped record of measurement activities. Textual information is stored which is suitable for unstructured queries or preparing management reports. Detailed information, including the measurement activity, the measurement program and version, and checklist information, is also provided. When combined with the **activity_data** table, the information will allow analysis programs to locate necessary data. The **activity_data** table stores pointers to the data.

Checklists are implemented as a master-detail structure in which arbitrary levels of nesting (without recursion) are provided. The list contents are prescribed in the **checklist_contents** table which links specific items into a list. Both the label for the list and the specific items in the list are described in the **checkitems** table. A **checkitem** is specified using a **name**, an **application** which carries it out, and one of four **types**. A **checkitem** which has **type** 'item' is an action to be performed. The remaining three checkitem types identify checklists. A 'list' type checkitem identifies a set of action items to be performed once. An 'incrementing_loop' or 'table_loop' type item identifies sets of repeating actions. Associated checkitem parameter tables store named parameters to allow further specification of the checkitems including the looping parameters for lists.

Supplemental tables in the **logbook** database contain lists of valid applications, procedure documentation, and status and assessment values. By associating these with the **checkitems** and **activities_log** tables, standard information for reports and queries is easily maintained.

A serial number is stored in each row in the measurements database. It is indexed in a table which associates its value with the table for which it was issued. Comments are stored using a **comments** table which records an author and a time. The comment contents are stored in 'lines' of up to 255 characters in an associated table. A serial number is used to allow comments to be linked to items of interest. Using this tool, a **comments_connection** table can associate any comment with any item in the entire measurements database.

5 Checklist Execution

A program which accepts instructions for measurement from a database may utilize the above structure to create a measurement system. While the techniques differ for each category of measurement, any measurement can be subdivided into a list of fundamental steps. A partial list of these steps includes:

- Change a measurement parameter (e.g., magnet current, or probe position).
- Measure using one or more instruments.
- Convert the instrument reading(s) to engineering units.
- Store selected results of the above two steps.
- Repeat these steps in a loop, varying the measurement parameter with each iteration of the loop.

The checklist-driven magnet testing program executes a sequence (checklist) of steps (checkitems), each of which specifies an action to perform using a set of parameters. The parameters vary with the type of action performed in the checkitem. The following is a list of definitions used by various checkitems.

[3] Sybase, Inc. 6475 Christie Ave., Emeryville, CA, 94608

[4] X Windows operates under a client/server model. The MIT X Consortium, 545 Technology Square - Room 217, Cambridge, MA 02139) supplies X servers for various architectures, X clients, Xlib for X windows development, and a window manager. The Open Software Foundation 11 Cambridge Center, Cambridge, MA, 02142 makes OSF Motif, a graphical user interface built on top of X. This includes their own library of functions for X program development and the Motif window manager.

Instrument definitions which instruments are used to take a reading or to change a measurement variable, and how each instrument is set up for use.

Calculation definitions which calculations are performed to convert instrument readings into engineering units, and what the parameters of the calculation's equation(s) are.

Loop definitions what list of actions are to be repeated over a specified range of values.

5.1 Checkitem Parameter Definition and Use

Checkitem parameters are defined by giving them a name and associating them with the checkitem. The following sections describe various checkitems and associated parameter definitions.

5.1.1 Instrument Definition and Use

Instruments are defined by giving each a unique name or serial number, and by providing the parameters that are appropriate for the particular instrument. For example, for an input to a digital voltmeter, the required parameters include:

- the name of the digital voltmeter used to read the input channel.
- the name of the multiplexer used to connect the channel to the digital voltmeter.
- the channel number on the multiplexer.
- the IEEE 488.2 command string used to perform the measurement.

This example is just a sample of the instrument definitions supported. New instruments can be added at any time.

Once an instrument is defined, it can be manipulated in a checkitem in one of two ways:

1. *measure* instrument_name
2. *set_instrument* instrument_name value

A *measure* checkitem is used to obtain a reading from a measurement instrument, while a *set_instrument* checkitem is used to change the setting of a control instrument. The value in the *set_instrument* checkitem can be the output of a calculation, a constant, or the present value of the loop executing the checkitem.

5.1.2 Calculation Definition and Use

Calculations are defined by giving them a name, and associating with the checkitem a set of parameters, which include the following:

- the name of the independent variable(s).
- the name of the dependent variable, which is the same as the calculation name.
- the name of the function that performs the calculation.
- the parameters to use within that function (e.g., the coefficients of the polynomial equation)

Once a calculation is defined, it can be executed anywhere in a checklist by specifying:

- calculate calculation_name

5.1.3 Loop Definition and Use

Two types of loop definitions are supported, incrementing loops and table-driven loops. The incrementing loop varies the loop value from an initial value to a final value in discrete steps. The table-driven loop type varies the loop value according to a table of values.

A loop is defined by giving it a name and providing a set of parameters for the loop. For an incrementing loop, the parameters include:

- the initial loop value
- the final loop value
- the loop value increment (or decrement).

For a table-driven loop, the parameters include:

- the table of values that the loop value is assigned.

Once the loop is defined, it can be activated as a checkitem by specifying the following in a checklist:

- execute_loop loop_name

6 Summary

The need to control a wide variety of hardware in a manner which gives a permanent record of the control system operation and configuration is quite distinct from the usual needs for a control system in which previous activities are uninteresting. This database driven system for measurement control has given us a unique way to control measurements in a flexible way which captures the details of a measurement, and makes them available for future review.

References

[1] Bruce C. Brown. Fundamentals of Magnetic Measurements with Illustrations from Fermilab Experience. In P. F. Dahl, editor, *Proceedings of the ICFA Workshop on Superconducting Magnets and Cryogenics*, page 297. Brookhaven National Lab, 1986.

[2] B. C. Brown, D. J. Harding, M. F. Gormley, M. E. Johnson, A. J. Lennox, K. J. McGuire, J. E. Pachnik, J. K. Plymale, R. A. Shenk, and A. A. Wehmann. Data Acquisition System Design for Production Measurements of Magnets for the Fermilab Anti-Proton Source. *IEEE Trans. on Nuc. Sci*, NS-32:2050, 1985.