UPDATED Run II Computing and Software Plan for the DØ Experiment

DØ Run II Computing Planning Board

February 15, 1999

Abstract

This document is an update to the DØ Run II Computing and Software Plan from January 1997. We describe the updated (more specific) plans for computing and software systems, the current progress toward implementing them, and the future milestones to be met between now and the target date (which has shifted to January 1, 2000).

Contents

1 Overview

The DØ plan for developing its computing systems for Run II, including both hardware and software components, was described in a document [1] dated January 31, 1997. This plan included "organization, schedule, staffing, and budget requirements to achieve a set of milestones culminating in operational systems in July 1999." The organization described in that document, with some minor changes, has been in place for two years, and we now pause to describe where we are with respect to that original plan. One obvious change is that roll-in of the detector is currently scheduled for April 2000 (or later?), and therefore our target date for fully operational computing systems and software has become January 1, 2000. We describe the updated (and more specific) plans for computing and software systems, the current progress toward implementing them, and the future milestones to be met between now and the target date.

1.1 The Run II Computing and Software Organization

The formal organization to manage Run II computing and software was put in place in March 1996. The original co-leaders of the organization were Bruce Gibbard and Wyatt Merritt, and there were four groups in the organization, each with two co-leaders. Nick Hadley replaced Bruce Gibbard as co-leader in March 1997, and the four groups have changed into five, by elevating Monte Carlo from a subgroup of Algorithms. There has been some turnover in leaders of the groups, and the subgroup structure is evolving over time. But the basic structure is the same, and the group leaders still form the Computing Planning Board. This body oversees progress in general, and takes the decisions for overall strategy, which cross the group lines of responsibility. The five groups, with their current leaders and charges, are:

- Infrastructure Group (Herb Greenlee, Qizhong Li, Harrison Prosper). This group is responsible for the software development environment (e.g. compilers and debuggers), for software tools required in the system (e.g., design tools and GUIs), for graphics displays for the experiment, for the generic framework program(s), for the event model classes, and for the object persistency mechanism.

- Online Computing and Software Group (Stu Fuess, Paul Slattery). This group is responsible for the online computing system, which will handle the data acquisition, control and monitoring of the experiment, and the Level 3 trigger framework software. This group also handles the procurement of hardware for the data acquisition system; this activity is reviewed within the Lehman baselined part of the DØ Upgrade, under the management of Jim Christenson and Mike Tuts.

- Algorithms Group (Serban Protopopescu, John Womersley). This group is responsible for designing and coding the algorithms for the reconstruction program, the Level 3 trigger, and the offline calibration and alignment programs.

- Monte Carlo Group (Boaz Klima). This group is responsible for simulation tasks for the experiment. These include the GEANT-based full simulation of the experiment (DØGSTAR), the C++ wrapper which makes Monte Carlo information available in the Run II persistent data format, the development of a fast parametrized simulation of the experiment, and (possibly) a future migration to GEANT4.

- Production and Data Access Group (Mike Diesburg, Lee Lueking). This group is responsible for planning and operation of the reconstruction farm, for the design, procurement, and maintenance of the systems with which we access and analyze the reconstructed data, including the data handling software, and for database services.

In addition, we are receiving significant support from the Run II Joint Offline Computing Project within the Fermilab Computing Division. The major areas where we receive development support from this project are shareable C++ libraries for basic functions (ZOOM), data handling and tape access (SAM and ENSTORE), support for database implementations, and support for reconstruction farm software. We also receive support for visualization tools (Open Inventor), physics analysis tools, and simulation. The joint project handles the procurement of the farms, analysis systems, and mass storage system (robot and tape drives). The project office is coordinated by Ruth Pordes and Dane Skow of the Computing Division, and further information is available from its web page at http://runIIcomputing.fnal.gov .

Support for online computing is also being provided from the ODS department of the Computing Division for development of some major pieces of the online software, in particular the client server messaging system.

1.2 Current Status of Computing and Software

In January 1997, we had a plan for Run II computing and software: to build an all new C++ reconstruction system and software trigger, to use a standard sequential file format for primary data storage (thus requiring construction of an interface from C++ to the persistent file format), to build an all new online system based on Unix and Windows NT, to build a GEANT3-based simulation of the new DØ detector and make it function with the new reconstruction system, to procure analysis systems and develop software for them which could handle the petabyte-size dataset expected in Run II, and to move our VMS- and Fortran-based developer and user communities to Unix, Windows NT, and C++. We had, at that time, a partial GEANT3 simulation which wrote out data in the Run I Zebra format only, and no C++ reconstruction code except for the fiber tracker system. We also had a core group of about 30 people ready to start on these projects.

In January 1999, our status with regard to this plan is as follows:

- We have a complete GEANT3 simulation of the new detector, and a C++ wrapper which produces digitized data for each subsystem in the new Run II event model classes which are then written out in the new Run II file format (DSPACK). Verification of this simulation has been underway for several months, and the first production version of the program is being used now. It is now generating about 5K events/day (at Fermilab). The aim is to bring remote sites online and produce a sample of $\approx 140K$ events for Phase 1; these events will be used for several different purposes mentioned below. Phase 2 will begin in October 99 and produce events with "hidden" physics samples for a harder test of the reconstruction. Design work is underway for a fast parametrized Monte Carlo to supplement the full GEANT simulation.

- The infrastructure packages to support reconstruction are in place and in use. These packages include the framework, the DØ geometry system, the event model classes (edm), the parameter input classes (RCP) and the persistency package (DØ OM) with an implementation which stores the data in DSPACK files. There are also first versions of the trigger framework code. Two missing pieces of infrastructure are the interactive framework and the event display framework.

- We have chosen to restrict ourselves to the KAI C++ compiler on Unix platforms (including Linux) and to use the Microsoft Visual C++ compiler on Windows NT. Code is stored in the cvs code management system, and releases are made using SoftRelTools (adapted from the BaBar product) and CTEST/CTBUILD (provided by DØ). We have code releases to the IRIX platform weekly; starting with the current (t42) release, we'll also be making weekly releases to Linux and OSF1. We have a new Web-based system about to be advertised which will allow code developers to conveniently test their changes on the three systems above plus Windows NT, using the build machines at Fermilab.

- We have a reconstruction program which can read and unpack the digitized data for all subsystems, and perform the first order reconstruction tasks (make calorimeter and preshower clusters, find global tracks in the CFT and SMT central acceptance, find primary vertices). Work is ongoing to make the next major release of the reconstruction system (the January 12 release), which is scheduled to have in place a first version of particle ID (jets, missing ET, electrons and photons, taus, muons, and secondary vertices). This release should come close to satisfying the December 31 milestone goals. Once this version exists, we will work on making it into a production version which will be used to process the Monte Carlo data from Phase 1 of the challenge. A major goal of the Monte Carlo Challenge is to tune and verify the reconstruction code.

- We have prototypes of the data handling system (SAM and ENSTORE) which have been developed within the Computing Division with DØ participation. This includes a prototype of the event/file database using ORACLE. These systems have their own Linux-based test beds, and can read and write to a robot. We anticipate exercising these systems in the analysis of Phase 1 of the Monte Carlo Challenge.

- We have reviewed the plans for physics analysis systems and have a committee report detailing the architecture we want to see for analysis from central systems to desktops.[2]

- We have software to handle the data acquisition for the commissioning of the first two systems to come online in the control room in January and February, and are working on the monitoring software for these efforts. Test stand data acquistion has been proceeding for some time.

- We have in hand a new EMASS robot and a few tape drives for testing, as well as a new SGI Origin 2000 with 800 GB of disk, a prototype Linux reconstruction farm, Linux test bed systems for SAM and ENSTORE, and build machines for the four Run II operating systems. These resources should be sufficient to support software development and systems testing over the next few months. Further major acquisitions (another robot, analysis systems and farm systems) are scheduled for procurement by Oct. 1, 1999. The funding profile seems sufficient at present to have enough capacity on hand at the beginning of Run II to handle our reconstruction and analysis needs.

We have created a Microsoft Project file with specific names attached as resources to the tasks remaining between now and January 2000. This file shows ≈ 120 people working on the various projects. The ramp up over the past two years has represented a tremendous effort from the collaboration and the Computing Division. We still have missing pieces, but effort has been assigned to almost all of these. There is still a great deal of work to be done, but the good news is that the structure is in place to recruit contributions from a wider community, not required to be present at Fermilab or expert in C++ class design. The primary management tracking of the project comes from comparison to a list of major milestones maintained by each group. In general, we are running about 1-2 months behind the milestones as they were understood at the Bloomington workshop (July 97), except for Global Tracking and Level 3. A major concern is to recruit more effort for the Global Tracking group and for Level 3 algorithm development from the collaboration. Assuming we can do this, and that the delivery of SAM and ENSTORE remains on target, we should be able to start Run II with a functional computing and software system that will produce timely physics.

2 Infrastructure

2.1 Introduction

This document contains an overview of the main DØ offline infrastructure software packages: DØOM [3], EDM [4], the program framework [5], and RCP [6]. These are described below.

2.2 DØOM

DØOM, which stands for "DØ Object Model," is DØ's object persistence package [3]. Object persistence is the process by which a tree, or web, of interconnected C++ objects in memory is "flattened" into a machine-indendent external format, for example a disk file.

2.2.1 Historical Perspective

In April of 1995, the DØ Analysis Computing Committee recommended that a search be undertaken to identify a replacement for ZEBRA as D0's data persistency mechanism [7]. Later that year, the DØ Languages and Data Structures Committee recommended the DSPACK package [8], written by R. Zybert of CERN experiment NA49, be adopted as the ZEBRA replacement [9]. The Languages and Data Structures Committee also recommended, in order to retain flexibility, that DSPACK be "kept at arms length," meaning that a layer of DØ-written software should isolate most DØ programmers from direct DSPACK calls. The development of the DØOM software package by DØ has basically followed these recommendations. DØOM was first released as a working package, using DSPACK as its I/O mechanism, in about July, 1997.

2.2.2 DØOM Overview

DØOM consists of the following elements.

1. Object model classes.
2. Preprocessor and class dictionary.
3. Front end I/O interface.
4. Back end I/O mechanisms.

These are explained in greater detail below.

2.2.3 DØOM Object Model Classes

The object model classes are those classes that are allowed in persistent classes. The persistent class designer must familiarize himself with these classes and the requirements for using them. First of all, all persistent classes are required to inherit directly or indirectly from base class d0-Object. Secondly, only certain types or classes may be contained in persistent classes. In general, atomic types, ANSI strings, STL containers, bare pointers, and DØOM's smart pointer class, d0-Ref$< T >$, are allowed. Other user-defined classes may also be included by value or by reference.

2.2.4 Preprocessor and Class Dictionary

Persistent classes are made known to DØOM by having their headers preprocessed by the DØOM preprocessor, d0cint. The purpose of the preprocessing step is to generate dictionary information that is accessible by the persistence mechanism at run time. Makefile fragments are provided by the DØOM package to automate the preprocessing step within the DØ release procedure. The DØ preprocessor is based on ROOT's C++ interpreter, cint.

The preprocessor generates additional C++ source files, which must be linked into user programs. These additional source files cause dictionary information for persistent classes to be generated. The dictionary classes store the layout of C++ classes in memory by name and type, for use by the persistence mechanism. Normally, class dictionary information is only used internally by DØOM.

DØOM has a limited ability to handle classes for which dictionary information has not been linked in. Because of the self-describing nature of the DSPACK data format, DØOM can generate dictionary information for such "unknown objects" on the fly. Unknown objects can not normally be accessed as native C++ objects. The generated dictionary information is sufficient, however, to allow the copying of unknown objects from an input stream to an output stream. It can also be used by special purpose programs, such browsers, that are willing to access the class dictionary in order to see inside unknown objects.

2.2.5 Front End I/O Interface

The DØOM package includes a standard abstract I/O interface which can be used with various I/O back ends. The front end I/O interface is physically decoupled from the I/O back ends, allowing users to decide at link time which I/O back ends to use. Multiple back ends can be linked, with the choice between them made at run time. The main back end I/O mechanism for DØ event data is based on the DSPACK package from the NA49 experiment at CERN.

The main I/O interface class is called d0Stream. Users of the DØ program framework do not normally need to interact directly with d0Stream for event I/O, as utility packages that do this are available (see Sec. 2.4).

2.2.6 Back End I/O Mechanisms

The basic design of DØOM is that the object model classes (d0_Object, d0_Ref$< T >$, etc.) and the I/O interface classes (d0Stream, etc.) constitute a "thin layer" through which various I/O back ends are accessed. The purpose of the thin layer is to provide abstraction and physical decoupling from the back ends, which is where the hard work of persistence is actually done. The I/O back ends, along with the preprocessor and dictionary system, are the most complicated parts of DØOM. Note that one can compile and link a working program without use of the thick components (I/O back ends, preprocessor) of DØOM.

Four I/O back ends currently exist for the following I/O packages:

1. DSPACK — From NA49.
2. EVPACK — Based on DSPACK.
3. MSQL — Free relational database.
4. ORACLE — Commercial relational database.

The DSPACK I/O Mechanism The DSPACK software package can be thought of as an improved ZEBRA. DSPACK fixes most of ZEBRA's notorious problems. DSPACK objects (called "data sets") are dynamically allocated and deallocated, and they can contain pointers and arbitrary structures.

The DSPACK I/O back end operates by copying C++ objects in memory to or from DSPACK objects, which are then read or written using DSPACK's built in I/O. Thus, with DØOM's DSPACK I/O back end, the data ends up being copied one more time than would be the case in a more tightly integrated system. The performance penalty implied by the extra copy is not currently thought to be very significant in comparison with other factors. Should this turn out not to be the case, DØ retains the freedom, because of the physically decoupled nature of DØOM, to switch to another, more highly optimized, I/O mechanism in the future. The current plan is still to use the DSPACK (or possibly EVPACK) I/O mechanism for storing most DØ event data.

The EVPACK I/O Mechanism The EVPACK I/O back end is a variant of DSPACK that embeds DSPACK format data in a DØ-designed physical record structure [10]. EVPACK supports features not present in native DSPACK I/O, including random access reading of events and data compression.

Database I/O Mechanisms There are currently two database back end I/O mechanisms for DØOM based on the MSQL free relational database and the ORACLE commercial relational database. These are being developed with the idea of using DØOM to access calibration data stored in ORACLE. The way in which calibration data will be accessed is not yet a solved problem in DØ. Indeed, it is not yet certain that DØOM will be used, although that is the current plan.

The database I/O interface is a superset of the event-oriented I/O interface. Database I/O uses abstract class d0StreamDB, which is a subclass of d0Stream. Class d0StreamDB supports database-specific concepts such as queries and transactions.

Alternatives to DSPACK The Languages and Data Structures Committee considered various alternatives to DSPACK. The other main serious contender was the ADAMO package, which was also developed within the high energy physics community at CERN. ADAMO internally uses an entity-relationship data model (in this respect, ADAMO somewhat resembles a relational database). ADAMO was eventually rejected because it was thought to be less object-oriented than DSPACK. The committee also considered and rejected the idea of using a commercial object-oriented database because of concerns about cost, licensing, and fears of a commercial product or company becoming non-viable in the future.

The Languages and Data Structures Committee completed its work before the advent of ROOT in recent years within high energy physics. ROOT presents itself as an all-encompassing solution for many problems, including data persistence. As such it could replace either DSPACK as a back end I/O mechanism, or all of DØOM. However, we believe that the DØ's current DØOM with a DSPACK back end has two critical advantages over ROOT that continue to make it the better choice. The first advantage is the physically decoupled organization of DØOM that allows different I/O mechanisms, such as the database I/O back ends, to be linked into a DØOM program without changing any source code. ROOT has no such ability. ROOT has no physically-decoupled thin layer, but is more tightly coupled and monolithic. The second advantage is that DSPACK data is self-describing. The DSPACK data format includes embedded within itself a DSPACK data definition record. This allows DØOM to support data-browsing, copying of objects lacking linked in dictionary information, and reading old format data ("schema evolution"). ROOT data is not self-describing. There is some interest in writing a ROOT back end I/O mechanism for DØOM, but this is not currently considered to be a high priority.

2.2.7 Future Development of DØOM

Since its initial release, DØOM has added features (especially support for objects without dictionary information), and generally been improved. It has also added other back end I/O mechanisms. As far as reading and writing event data in DSPACK format is concerned, DØOM is now a mature product. It is now widely used in DØ. The database I/O back ends need further development before they will be ready for use by the reconstruction program. Other possibilities for future development are the development of network I/O back ends, to be used to ship events up from level 3, or to be used to communicate with a database server.

Figure 1 shows the tasks and schedule that were given to the Von Ruden Review Committee in October, 1997. According to the schedule, DØOM should be completely finished now. The actual status of the tasks in the list are as follows:

- Task II B 1 (MSQL interface) is finished. The first working version was released in February, 1998

- Task II A 1–2 is mostly finished. The EVPACK back end I/O mechanism supports random access and compression. Federated events (i.e. splitting events accross multiple files) has not been worked on, and there are no current plans to work on it. The first working version of EVPACK with random access but no compression was released in July, 1998. Compression was added in September, 1998.

- Task II B 2 (ORACLE direct interface) is partially finished. The first working version was released in December, 1998. However, as stated above, the current version of the ORACLE database interface is not considered to be adequate for use by the reconstruction program to access calibration databases generated online. The main thing that it is missing is the

```
Object Persistency Schedule

Tasks                                                  Start    Finish

I.  General.
    A.  Data model.                                               7/97
    B.  Class dictionary.                                         7/97
    C.  Preprocessor.                                             7/97
    D.  General I/O interface.                                    7/97

II.  I/O Mechanisms.
    A.  DSPACK.
        1.  Sequential access to monolithic events
            using standard DSPACK I/O.                            7/97
        2.  Alternative DSPACK file formats.
            a) Random access.                          10/97      1/98
            b) Compression.                            10/97      1/98
            c) Federated events.                        1/98      5/98
    B.  Random access/databases.
        1.  Lightweight/distributable database.
            a.  Analysis/choice of product(s)
                (DSPACK, mSQL, ASCII, etc.).            7/97     10/97
            b.  Design.                                 8/97     11/97
            c.  Implement.                             10/97      1/98
        2.  Commercial OO or relational database
            direct connect.
            a.  Analysis/choice of product
                (Oracle, Objectivity, other OODB).      7/97      1/98
            b.  Design.                                 1/98      2/98
            c.  Implement.                              2/98      5/98
        3.  Object server.
            a.  Analysis/choice of product
                (CORBA, local (CD) product, etc.).      7/97      5/98
            b.  Design.                                 5/98      8/98
            c.  Implement.                              8/98     11/98
```

Figure 1: Tasks and schedule for Von Ruden Review, 10/97.

ability to control the mapping from C++ to the database schema. The current ORACLE (and MSQL) interface uses a fixed mapping.

- Task II B 3 is currently being designed and prototyped.

The following areas are being worked on, or are in need of work, in DØOM.

- Calibration Data Model. This is a layer of software that will sit between DØOM and the Calibration Manager. This is somewhat analogous to what the Event Data Model (EDM) does for event data. The Calibration Data Model is currently in the design stage.

- Flexible C++ to database schema mapping. A way of giving programmers control of the mapping from C++ objects to their database representation is needed. Such flexible mapping scheme added to the current ORACLE interface would make it directly useful for accessing ORACLE calibration data. Any flexible mapping scheme will require something like a Database Definition Language (DDL). A DDL would be required even if the direct ORACLE interface is dropped in favor of a client-server ORACLE interface.

- Client-Server DØOM I/O back end. Mainly for performance reasons, a DØOM back end I/O mechanism that acts as the client which connects to a database server is highly desirable. Such a scheme can greatly improve performance when many farm nodes simultaneously access a calibration database by caching data read from the database. Database client-server architectures are currently being designed and prototyped. The current plan is to have the client and server talk to each other using CORBA. This will allow the server to be prototyped using Python, and then upgraded to C++, if necessary, for performance reasons.

- A memory buffer or network back end I/O mechanism. A way is needed to ship data events up from the level 3 trigger to the data acquisition system. To do this it will be necessary to transfer events to a flat memory buffer, a socket, or some similar arrangement. The DSPACK data format is quite suitable for this purpose. It is thought that the EVPACK I/O mechanism, which embeds DSPACK format data in DØ-specified physical records, could be adapted with a modest effort.

2.3 EDM

The purpose of the Event Data Model (EDM) [4] is to provide the means by which distinct components (called "packages") of the reconstruction or triggering program can communicate with each other. The major design goals are for these programs to be easily maintained and extended over the multi-year lifetime of the project. To this end, it is of great importance for the EDM to allow for *physical decoupling* [1] of the interacting packages.

2.3.1 Design Overview

The EDM consists of about 15 classes which form two libraries. These classes include Event, which is the container for all data resulting from a single collision, and AbsChunk, the abstract base class for all user-derived classes representing the results of reconstruction (called "chunks"). It also contains classes for extracting chunks from the Event in a type-safe manner, and to provide memory management. Also provided are the classes necessary for making associations between chunks, again in a manner that does not introduce physical coupling. These classes make use

[1] *Large-Scale C++ Software Design*, J. Lakos, Addison-Wesley Publishing Company.

of, but insulate the user from, the DØ object model (DØOM), which provides the persistency mechanism for event data.

Another part of the EDM is a set of guidelines to be followed by authors of reconstruction software. These rules are designed to enhance the maintainability of the reconstruction and triggering software, through limiting physical coupling.

2.3.2 Current Status

The EDM has been in use since January 1998, and is used by all existing Run II reconstruction software in DØ. It has recently been ported to Windows NT, where testing is underway by the Level 3 software group. The current version is being used for production of events for the MCC99 Challenge.

2.3.3 Future Plans

While the current version of the EDM is adequate for present use, experience gained over the past year has shown several ways in which the model can be improved. Design changes are now being discussed by the General Review Committee. Our current goal is to release an improved version of the EDM in March 1999. This version will provide the additional functionality for which need was discovered during the use of the current version.

2.4 Program Framework

The program framework [5] acts as a kind of glue which binds together program fragments called packages (not to be confused with d0library or cvs packages). The program framework will be used for large programs, such as the Monte Carlo and DØ reconstruction programs, as well as by individuals writing analysis code. The program framework provides a convenient way for code developers to plug their own code into a working program.

Each framework package consists of one C++ class that multiply inherits from base class Package, and from one or more interface (or hook) base classes. At program startup, the packages and their corresponding interfaces register themselves with the framework. The framework then instantiates the packages and begins calling the interfaces in some appropriate sequence, as determined by the framework rcp file. The framework internally manages a work queue that can contain action codes and/or data. Typically, a single event will be passed through a fixed sequence of packages and then discarded. More complicated data flows are also possible. Non-data-event actions, such as begin and end run events are also represented as work queue entries.

2.4.1 Standard Framework Interfaces

There is no fixed set of interfaces that are supported by the framework — new interfaces can be added withoug modifying the framework source code. However, the framework does provide a standard set of interfaces. The standard interfaces are shown in Table 1 in their standard calling sequence.

The order of calling the interfaces is also not fixed, but can be controlled by the programmer at run time by changing the framework rcp file. The Generator and Decide interfaces have access to the framework work queue. These hooks are able to modify the normal sequence of interface calls. It is only in this way that the RunInit, RunEnd, and JobSummary interfaces are normally called.

Table 1: Standard framework interfaces.

Interface Name	Purpose
Generator	Place a new event on the work queue.
Decide	Modify work queue.
Builder	Construct an event that is already on the work queue.
Merge	Merge events on the work queue.
Filter	Optionally stop further processing on this event.
Process	Process (modify) an event.
Dump	Dump an event.
Analyze	Analyze an event.
RunInit	Called at beginning of run.
RunEnd	Called at end of run.
JobSummary	Called at end of job.

Table 2: Utility packages.

Package Name	Purpose
NewEvent	Place an empty event on the work queue.
ReadEvent	Read an event using DØOM and place it on the work queue.
WriteEvent	Write an event using DØOM.
DumpEvent	Dump an event.
MergeEvents	Merge several events on the work queue into one event.

2.4.2 Utility Packages

The framework itself has no intelligence with respect to the processing of data. The framework knows nothing about, nor is it physically coupled to, the DØ event data class Event. All non-trivial processing of DØevent data, including I/O, is handled by packages. Several utility packages are provided along with the bare framework to handle common functions, mainly I/O. The utility packages are listed in Table 2.

2.4.3 Framework Status

The first working version of the framework was released in March, 1998. The framework has been adopted for use by the DØ algorithms group for the DØ reconstruction program, and is now widely used in DØ.

The current version of the program framework only supports batch processing. Input to the framework consists of a framework rcp file. Rcp files are text files that allow variables to be defined and accessed by name and type. The contents of the framework rcp file are basically a list of interfaces to call, and a list of packages to instantiate, along with the corresponding package rcp files. There is no ability to interact with the current batch framework after the program starts.

It has always been recognized that an interactive version of the framework will be required for use online, for event displays, and for other reasons. The design of the interactive framework is now underway.

2.5 RCP

In Run I, DØ used Run Control Parameter (RCP) objects to control and configure executable programs; for example, DØRECO, the DØ reconstruction program. We plan to use RCP objects, similarly, in Run II. At an abstract level, an RCP object is a container of (**name**, **value**) pairs, which we refer to as parameters. The **name** is of type **string**. It identifies the parameter. The **value**, of the parameter, can be one of the following types

- bool
- int
- float
- string (from the C++ Standard Library)
- RCP

or vectors (again, from the C++ Standard Library) thereof. Note that RCP objects can contain RCP objects. The current implementation of the RCP package does not yet support vectors of types. This improvement plus an implementation of the RCP database are needed to complete the RCP package. The RCP database is being worked on as part of the Support Databases project with the Computing Division.

2.5.1 The RCP Database (RCPDb)

Our need to enforce uniqueness of RCP objects, and to make them persistent over many years, makes a strong case for the use of a real database. However, it is not necessary to use an object-oriented database to store objects; a relational database will do. Currently Oracle is the database of choice. The RCP (Oracle) database (RCPDb) is the ultimate repository of RCP objects. The CVS repository is the ultimate repository of official RCP scripts. These scripts serve as the specification, blueprint if you will, for those RCP objects destined to be used in critical DØ programs, like DØRECO. Other RCP objects can enter RCPDb via specifications that can come via the Web. We anticipate that a large fraction of the collaborations Monte Carlo event generation will be done at sites remote from Fermilab. We failed miserably in Run I to provide the necessary tight (and automatic) binding between Monte Carlo files and the specifications for their creation. We plan to effect this binding by embedding RCPIDs in Monte Carlo events. For this really to work as we wish, a very high level of reliability is required in the functioning of the database and the interfaces to it. RCP objects are to be stored in RCPDb in a fully dissected form. This will allow us to use standard SQL tools to query, in detail, RCP objects. There will be write-only access from the CVS repository to RCPDb, the updating of which will occur automatically upon each code release from CVS. Once updated, there will be an automatic extraction of RCPDb to a read-only, exportable, file system for the benefit of sites remote from Fermilab. The RCP objects in the extracted database will, most likely, be stored in a manner that renders access to them as efficient as possible. A human-readable form of any RCP object can be had by invoking the objects print method.

2.5.2 Summary

Our aim in building, from scratch, a new RCP system for Run II is to effect a marked improvement in the bookkeeping of data-sets be they from the Tevatron, or from a remote Monte Carlo farm. We wish to build a system that will provide a tight and automatic binding between the instructions for creating the data-sets and the data-sets. The RCP mechanism is our proposed solution.

3 Monte Carlo

3.1 General information on the Monte Carlo group

The Monte Carlo (MC) group maintains its documentation on the WWW. The main page website is

http://www-d0.fnal.gov/computing/MonteCarlo/MonteCarlo.html

It includes separate sections on (and links to) the group organization, the D0 detector simulation program (D0GSTAR/simpp/digi), a Parametrized Monte Carlo Simulator (PMCS), and the D0 Monte Carlo Challenge (MCC99).

3.2 History

The MC group was created in the summer of '97. We adopted the D0GSTAR product that was developed over a long time by Yuri Fisyak and John Womersley. At that time it was a standalone program residing on a separate disk which was not part of the D0 Run II environment and it needed a lot of testing and updates. In addition, we decided that since GEANT4, the C++ version of GEANT, was not going to be ready soon enough for us to use, we would develop a wrapper (simpp) which will convert the FORTRAN/ZEBRA information generated by our GEANT3 program into the new Event Data Model format. Later on we also decided that the digitization for each detector will be written either inside D0GSTAR or preferrably outside simpp.

D0GSTAR's first official release in CVS (v1.0) took place on Dec. 22, 1997. The next main releases happened on Jan. 19, 1998 (v1.1), May 27, 1998 (v2.0), and June 17, 1998 (v2.1). Following the completion of the transition from the "isolated" program to the D0 official Run 2 environment and version 1 of the wrapper ("simpp") we released the first version in the Run II SRT environment on Aug. 27, 1998 (v3.03). After quite a few releases which were done following improvements, updates, bug fixes, inclusion of the digitization packages etc, we settled on our (and D0's) very first Run II Production release on Nov. 17, 1998 (p01.00.00). With a few more bug fixes we declared our latest version of the production release, p01.00.02, as our official version to be used for the D0 Monte Carlo Challenge.

Several months ago we have also started working on a Parametrized Monte Carlo Simulator (PMCS). The proposal was submitted to the collaboration in June '98. Following no major objections, the code development has started with code already included in the CVS library.

3.3 Current Status

The MC group currently consists of the following active people (with their main responsibilities marked by their names):

```
Boaz Klima - Head
Bogdan Pawlik - GEANT3/CVS/Releases/Production
Shuichi Kunori - simpp/CVS/Releases
Vladimir Sirotenko - Muon (*)
Yuri Gornushkin - Muon
Marek Zielinski - Calorimeter (*)
Vishnu Zutshi - Calorimeter
```

```
DooKee Cho - Calorimeter
Kwok Ming Chan - Calorimeter
Bogdan Pawlik - SMT (*)
Lisa Shabalina - SMT
Meena Narain - SMT
Norman Graf - CFT (*)
Phil Baringer - CPS (*)
Don Coppage - CPS
Ming Liu - FPS (*)
Dennis Shpakov - FPS
Sarah Eno - PMCS (*)
Nick Hadley - PMCS
Emmanuel Gouveia - PMCS
Elemer Nagy - PMCS
Harry Melanson - PMCS
Armand Zylberstejn - Testing (Saclay)
Pierre Bonamy - Testing (Saclay)
Laurent Duflot - Testing (Orsay)
Pierre Petroff - Testing (Lyon)
Michel Jaffre - Testing (Lyon)
Gilvan Alves - Testing (Rio)

* - in charge
```

The three main ingredients of the simulation program, D0GSTAR, simpp and the digi packages, have been developed and tested quite extensively over the last several months. All Run II detectors, SMT, CFT, CPS, FPS, MUON, CAL, are included in the program with their up-to-date geometry and digitization. Lots of plots testing different aspects of these programs on many different type of events can be found directly from the following MC web page:

http://www-d0.fnal.gov/computing/MonteCarlo/mc-tasks.html

For the production version in particular we've generated thousands of events, varying from simple single-particle events all the way to the most busy/complicated Top events.

The PMCS subgroup is currently working on developing the code based on the proposal approved 6 months ago. The proposal which includes a list of available/used PMCS's, goals, input formats, a description of simulation of hits and objects, and the content of version 1 was presented to the MC group, to the GCAS group and was distributed among the Physics Groups' leaders.

3.4 Future plans

Our future plan consists of the following main elements:

- D0GSTAR

  1. Continue testing the geometry and digitization as implemented, fix bugs, update according to hardware changes, and improve parameterization
  2. Review GEANT parameters for tracking and calorimetry simulation
  3. Introduce a more precise magnetic field map
  4. Handle better random number generator and seeds

5. Tuning according to performance (event size, cpu/event)

- simpp/digi

  1. Continue testing the conversion code and the digitization, fix bugs, update according to hardware changes, and improve parameterization.
  2. Review output chunks.
  3. Link to MC++ (wrapped version of our MC generators).
  4. Add event mixing to take care of multiple interactions.
  5. Tuning according to performance (event size, cpu/event)

- Miscellaneous

  1. Develop an official testing package for D0GSTAR/simpp/digi.
  2. Support other platforms including an official certification procedure.
  3. Improve documentation.
  4. Agree on common geometry database with D0RECO.
  5. Develop a Parameterized Monte Carlo Simulator.
  6. Shower Library(?).
  7. GEANT4(?).

3.5 Monte Carlo Challenge

On July '98 the Computing and Software leaders, Wyatt and Nick, established a committee to define the goals, scope, and definition of a Monte Carlo Challenge for the D0 experiment, to exercise the computing and software systems for Run II. The charge for this committee was given below.

The committee is charged to deliver, by Sept. 30, 1998, a plan for the D0 Run II Monte Carlo Challenge (MCC99). This plan should include:

o A statement of the specific goals of the challenge with regard to testing the Monte Carlo program (D0GSTAR) and the infrastructure for running it, testing the reconstruction software and the infrastructure for running it, and exploring the physics performance of D0 in Run II

o A list of the Monte Carlo samples required to achieve these goals

o An estimate of the computing resources required to acquire these samples by the date of the Challenge

o A plan for how and where the Monte Carlo data from the Challenge samples should be stored and accessed

The list of committee members is: Boaz Klima (chair), Paul Grannis, Shuichi Kunori, Lee Lueking, Harry Melanson, Pierre Petroff, Alberto Santoro, Heidi Schellman, Ransom Stephens, Armand Zylberstejn.

Since then the committee met five times. We agreed with the main goals of the MCC99 as written in Nick/Wyatt's charge and elaborated on details. We came up with the following understanding of the main goals:

1. Testing the Monte Carlo program (D0GSTAR) - this basically means testing D0GSTAR's performance both in terms of computing (CPU and size per different types of events) and wrt its accuracy in representing all detectors in D0 during Run II.

2. Testing the infrastructure for running D0GSTAR - the idea here is to use as much of the Run II computing and software systems as possible to test its performance and reliability in a large scale operation. This includes running D0GSTAR both in and out of Fermilab, transferring files, using the database, file storage, file accessibility etc. This aspect of the MCC99 will serve as important input to the experts in charge of these resources. Any Run II resource which is not going to be ready by the time of the MCC99 will be replaced by a temporary solution.

3. Testing the reconstruction software (D0RECO) - this entails examining the efficiency and rejection power of RECO algorithms, hits and track finders as well as particle id's.

4. Testing the infrastructure for running D0RECO - this is similar to goal 2 but for D0RECO. Currently the plan is to process events with D0RECO using Fermilab farm(s).

5. Exploring the physics performance of D0 in Run II - here we expect detailed studies using different kind of physics processes to examine algorithms' performance, particles' momentum resolution, background rejection, efficiencies for physics processes, discovery potential, etc.

The strategy is to do the MCC99 in two steps:

1. Phase 1 - to be done as soon as possible after the first production release of D0RECO is made and the MC samples are processed with that version of D0RECO. This is currently expected for February '99. The primary focus for phase 1 is to develop understanding of the object and detector algorithms, to determine object efficiencies, and background contaminations. For this phase we'll concentrate mainly on goals 1-4 though goal 5 maybe partially met. The physics processes will be mostly simple and will be known to the analyzers beforehand.

2. Phase 2 - later in 1999, probably around Oct. '99. For this phase we'll concentrate mainly on goals 2-5 assuming that goal 1 was met during phase 1. The physics processes will be more complicated than during phase 1 and may include multi-interactions. Assuming that a trigger simulation program will exist by then one can contemplate including trigger studies in this phase. We might also consider a double-blind physics exercise.

The current schedule is believed to be quite tight but doable. We are still on time in general though some individual tasks have had difficulties which might delay their convergence. The current main internal milestones are the following:

```
Dec. 1, 1998 - start generation of MC files for phase 1 of MCC99.

Dec. 15, 1998 - PRODUCTION version of DOGSTAR is frozen for phase 1 of MCC99.

Dec. 18, 1998 - start DOGSTAR production for phase 1 of MCC99 at FNAL.

Jan. 11, 1999 - a. finish generation of MC files for phase 1 of MCC99.
                b. finish LINUX-ing DOGSTAR (code put in cvs).
                c. start DOGSTAR production for MCC99 on LINUX farm.
                d. start DOGSTAR production for MCC99 at BU.

Jan. 25, 1999 - start D0RECO production for phase 1 of MCC99.
```

`Feb. 10, 1999 - finish D0GSTAR production for phase 1 of MCC99.`

`Feb. 25. 1999 - finish D0RECO production for phase 1 of MCC99.`

`Mar. '99 - begin analysis of phase 1 of MCC99.`

So far the first three milestones were met. We have generated the majority of events to be used for phase 1 of MCC99. The PRODUCTION version of D0GSTAR/simpp/digi p01.00.02 was extensively tested and then frozen. We are currently in the production phase on FNAL and BU and in the code conversion phase on LINUX (FNAL and Saclay) and IBM (Lyon). As of Jan. 10, 1999 about 44K events were processed with D0GSTAR/simpp/digi.

4 Algorithms

4.1 Role of the Algorithms Group

The DØ Run II Algorithms Group was established in June 1996, with responsibility for providing the software needed to reconstruct physics objects from the DØ detector; the generation of simulated raw data (together with the Monte Carlo group); the development of code for the software trigger (L3); and the development of software for the alignment and calibration of the DØ Detector.

4.2 Organization

4.2.1 Detector Subgroups

A number of subgroups were established within the Algorithms group. At present we have four groups organized around subdetectors (with their leaders noted):

- Silicon Tracker - Emanuela Barberis, Doug Norman
- Fiber Tracker - Norman Graf (pro. tem.)
- Calorimeter (including Central and Forward Preshowers) — Marek Zielinski
- Muon System - Pushpa Bhat, Taka Yasuda

These groups are responsible for "close to the detector" software. This includes clustering of hit elements, formation of space points from stereo views, track segment finding, etc. Additionally they provide assistance with the simulation of raw data, with geometry definition and with calibration.

There are also three cross-detector subgroups:

- Global Tracking - David Adams, Norman Graf
- Alignment/Calibration - Harry Melanson, John Hobbs
- Level 3 - Amber Boehnlein, Dan Claes

The first is responsible for track-finding using the combined central detector system in DØ (and is also expected to assist in interfacing to muon tracking using the muon system). The second group depends heavily on people contributed by the detector subgroups, and is responsible for providing a geometrical description of the detector, for developing the alignment program(s), and for calibration code and databases. The third group depends heavily on people contributed by the ID groups, and is responsible for providing Level 3 trigger software.

4.2.2 ID Groups

In the fall of 1998 we felt that it was time to move beyond the subdetector boundaries, and established object groups charged with identifying and measuring the objects useful for Run II physics.

The object groups are initially constituted as part of the Algorithms group within the Run II Computing and Software organization and report to the Algorithms group leaders. Initial priority within the object groups was given to the effort to make working reconstruction and L3 tools for their object on a timescale as required by computing and software milestones. We anticipate that, as the object groups grow and take on the full range of responsibilities as listed below, the time will come when it will be appropriate for the groups to move outside this framework. The time for this transition will be determined by the spokespersons in consultation with the computing, trigger and software management.

The object group's responsiblities for its object include:

1. Algorithms for L1, L2, and L3 triggers.
2. Tools for estimating rates and efficiencies of the triggered object.
3. Algorithms for offline reconstruction relevant to identifying the object.
4. Definition of the object classes.
5. Development of analysis tools that define the quality of the object and making sure that the event information needed for those tools is available for analysis (through the object classes).
6. Development of tools that facilitate estimating the efficiency of post-reconstruction selection of objects as a function of object quality.
7. Calibration of the objects.

If the necessary tools for performing the above tasks do not exist, the object group will assist in making those tools: the group should not wait for others to do the job. The object groups will persist after the start of Run II, and will be responsible for approving that part of every analysis that uses their object.

The groups that were formed initially are:

- Vertex finding - Meenakshi Narain, Frederic Stichelbaut
- EM-identification (electrons and photons) - Ron Madaras, Serban Protopopescu
- Jet and missing E_T identification - Laurent Duflot, John Womersley
- Muon-identification - Daria Zieminska

- Tau-identification - Qizhong Li

It is expected that Luminosity and b groups, a separate Missing E_T group, and possibly others, will be formed later. (Note that b's and missing E_T depend on the objects, e.g. muons and vertices, from the other groups that will be formed first.) The subdetector groups continue to exist and continue to provide the subdetector specific code.

4.3 Reviews

We have created six review panels to oversee our software. Each consists of four or five people including those with experience in a particular detector or reconstruction and at least one experienced in object oriented design. Reviews so far have concentrated on the design and documentation, rather than carrying out code walkthroughs, and have been conducted with minimal formality. We have not yet required that review be a prerequisite to code insertion in the library. It is likely that we will move to a greater level of formal review over the next year especially for the L3 trigger code.

4.4 Software Milestones in the Past Year

We have used internal milestones at approximately three-month intervals to measure (and to inspire) progress in the algorithms efforts. So far this has proved to be an effective way of managing the project. The milestones fall often enough to allow subprojects which are falling behind schedule to be identified relatively quickly, while not being so frequent to seem that nothing could be accomplished before the next one.

<u>February 28, 1998</u>

Using a temporary framework, a temporary geometry description, `d0om` for persistency and DSPACK for i/o:

- read events generated by the `DOGSTAR` event simulation program
- implement a package to digitize central fiber tracker (CFT) information
- implement a package to construct CFT clusters suitable for track finding and fitting
- implement track finding and track fitting using only CFT
- write out event with reconstructed information

The milestone was successfully met on March 17.

<u>June 30, 1998</u>

Using official reconstruction framework and its required interface classes:

- implement digitization packages for all subdetectors
- implement geometry classes for all subdetectors
- implement clustering packages for all subdetectors
- implement track finding and track fitting in CFT and SMT (Silicon Microstrip Tracker)
- implement first pass at primary vertex finding

Most items were concluded on time. Clustering for the central preshower (CPS) and implementation of geometry classes for muon system were not completed until the end of July.

<u>September 30, 1998</u>

The main goals for this milestone were to improve the performance of packages delivered for the June milestone. Additional items were:

- migrate digitization packages to `DOGSTAR` C++ wrapper
- generate raw data format starting from digitized data for
  - calorimeter
  - muon
- add material description to geometry classes
- implement multiple scattering and non-uniform field in track finding and fitting.
- add central muon system to global tracking
- implement secondary vertex finding (first pass)
- improve primary vertex finding

Due to lack of manpower in global tracking, the implementation of multiple scattering could not be done in time for this milestone nor the addition of the central muon system. Although all the necessary infrastructure was in place the generation of raw data for the muon system was not fully implemented.

<u>December 31, 1998</u>

In this milestone we included all the items that were not met in the September milestone plus a first pass at physics objects identification:

- implement multiple scattering in global tracking
- add central and forward muon system to global tracking
- implement generation of raw data format for all subdetectors
- EM objects (electrons and photons)
- Muons
- Taus
- Jets
- Missing E_T

At the time of writing, many of the items in this (ambitious) milestone are already finished, but we do not know the final date for meeting the full list. DØ now has the first version of a full reconstruction program. It finds tracks and vertices, and reconstructs electrons, muons (standalone only at present), jets and missing transverse energy. It uses simulated Monte Carlo events in the DØ persistent data format and employs the DØ event data model to access and store information, works in a fully C++ environment, and runs in the DØ offline framework.

As an example, Figure 1 shows a comparison between jets found in QCD events using a simple cone algorithm and the K_T algorithm of Catani et al. Both algorithms are implemented as framework packages in C++ as part of the DØ C++ offline reconstruction.

Comparison of jet algorithms

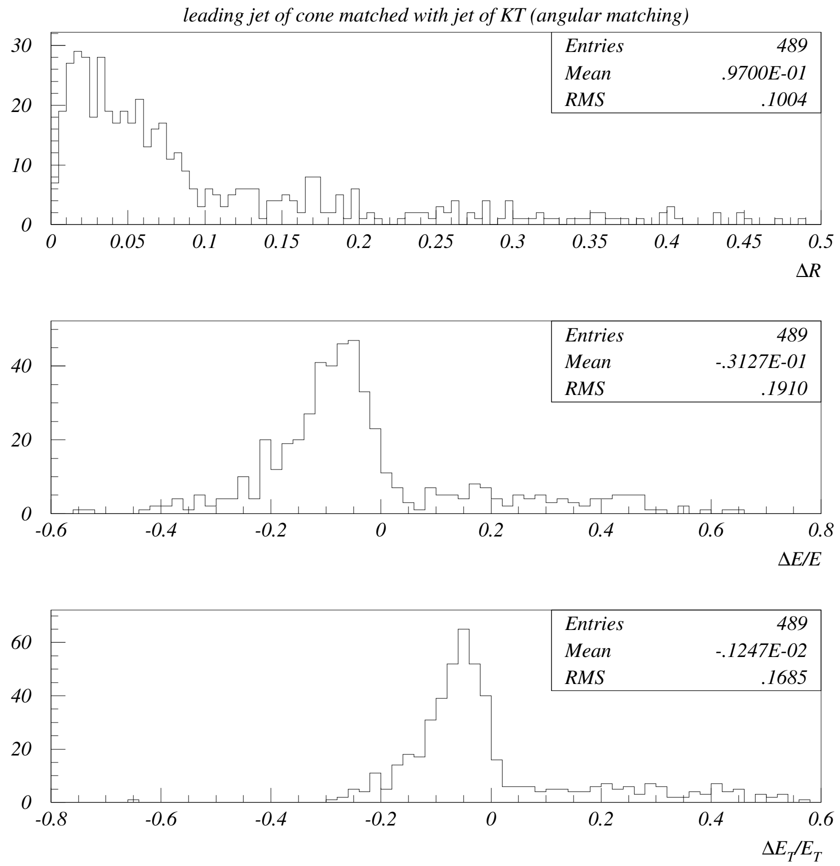

Figure 1: The leading jet of scone is matched with a jet of K_T with an angular matching on QCD events. The angular distance (top), the normalised difference in energy (middle) and E_T (bottom) are shown.

4.5 Future Plans

In the next six months most of the work will be in optimizing the performance of the reconstruction program and developing L3 algorithms. Major items that still need to be designed and implemented are: offline use of calibration data, alignment program, and energy calibration program.

4.5.1 Monte Carlo Challenge 1999

The reconstruction program after successfully meeting the January milestone and making a production release will be used to process all the data generated for the Monte Carlo Challenge 1999. Analysis of the results should provide guidance on required improvements, enable in-depth debugging, and serve to acquaint a much larger user community in DØ with the Run II software framework.

4.5.2 Future Milestones

We will have milestones in April 1999 and July 1999. The exact details for those milestones depend on how well we were able to meet the January milestone. Currently we expect to have:

April 1999 milestone

- complete global tracking of all subdetectors
- 2nd pass at physics objects identification
- improve secondary vertex finding
- optimize global tracking
- 1st version of alignment program for tracking detectors
- 1st pass at L3 tools for physics objects

July 1999 milestone

- Implement offline calibration for all detectors
- First version of energy calibration program
- optimize global tracking
- optimize vertex finding
- optimize physics object identification
- implement a package for b-physics objects
- 2nd pass at level 3 tools for physics objects

Milestones beyond July 1999 depend on the detailed status of reconstruction program and L3 tools at that time. We expect to have a second Monte Carlo challenge in October of 1999. Our overall goal is to have programs completed and ready for data taking and data processing by the beginning of 2000.

4.6 Areas of Concern

One major area of concern is the global tracking effort. While progress continues to be made in this area, it failed to complete the September 1998 milestone and is unlikely to fulfill all of its December 1998 requirements. The reason for this is a lack of manpower, especially of people able to devote a large fraction of their time to this effort. We are aware of the situation and are attempting to identify additional people, but it will no doubt take some time to bring any new recruits up to speed.

4.7 Conclusions

During 1998 there was a great deal of progress in development of a reconstruction program for Run II. Most of the necessary infrastructure is in place and algorithms for many of the tasks have been implemented. We have a management system which enables us to identify problems in a timely manner. With some additional manpower there is every reason to believe that an event reconstruction program, L3 trigger code, and alignment and calibration packages, will all be ready in time for data taking in the early spring of 2000.

5 Production and Data Access

The Production/Data Access group is responsible for the interface to the Computing Division for design, procurement, and management of the Run II computing systems, including farms, storage management systems, and analysis systems. We are working with the Joint Project's procurement process and through the Serial Media Working Group to achieve these ends. This group is also responsible for data handling and database services, and these activities are described below.

5.1 Analysis Computing Hardware

The Global Computing Model Working Group was formed within DØ to work out a model for the Run II computing hardware configuration. The result is summarized as follows:

- There will be three tiers of CPU, each with locally attached disk storage, consisting of
    1. A large Central Analysis Server (CAS) or centrally configured cluster,
    2. A handful of smaller Project Servers (PS), and
    3. Numerous desktop machines, which will be PCs (Linux or NT).
- Data will be exchanged between the servers and the mass storage robot via network attached tape drives.

The bulk of the computing resources will be in a large Central Analysis Server which will be about 80K MIPs, have 20 TB disk storage accessible at 400 MBps, and a high-speed network with a sustained throughput of 100 MBps. There is a desire among our outside collaborators to provide a few (5-6) smaller Project Servers each with about 5K MIPs, 2 TB of disk storage accessible at 40 MBps, and a network throughput of 20 MBps. Finally, the desktops will each, typically, have a few hundred mips, 50 GB disk, and network connections providing 10 MBps burst rates to the CAS and PS systems.

It is anticipated that the CAS will have large data sets which are common to all analysis, e.g. the thumbnails. It has been suggested that the PS systems might have specific stream data, e.g. electron, muon, jet, etc., and be used by specific analyses interested in that particular data. This

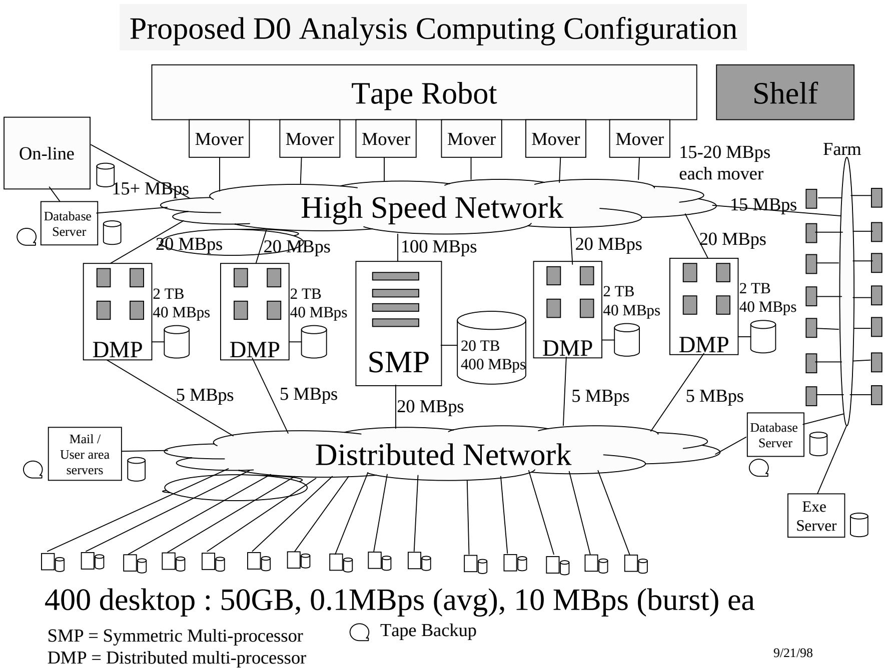

Figure 2: The DØ Analysis Computing Model.

would divide the usage load more evenly than if the servers were allocated by physics analysis group, such as top, susy, higgs, etc. It would also reduce the need for data transfers between the various servers.

We prefer a centrally administered large system for the bulk of the analysis CPU. This type of system is more easily managed than many smaller systems. It is also more convenient for most users to perform many analysis activities in this environment. Such large systems also enable disk storage to be locally attached , either with SCSI or Fibre Channel, if it is practical.

Disks on the servers and desktops will not be cross mounted. All data transfers between servers or desktops will be via ftp-like tools. In the next few years, as the Fibre Channel standards emerge, we may extend our use of this technology and consider using arbitrated loops for sharing data among homogeneous platforms.

We anticipate that there will be 400 or more desktops and that the average data transfer rate to and from the servers for each such desktop will be around 100 KBps. This rate would allow all 400 of the 50 GB disks to be completely updated with data from the servers once every two months. In reality, only a fraction of these desktops will frequently require large amounts of data moved to their local disks. The parameters for these desktops are simply our reasonable guesses about how moderately priced PC's, purchased during the run, will be equipped.

Using network-attached tape drives has several important advantages, and some drawbacks. The overall computing model includes the online DAQ and farm systems, in addition to the analysis computing system. The network approach provides flexible access to the mass storage system to accommodate changing resource usage. For example, we need to be able to increase the number of tape drives being employed by the DAQ when the detector data rates increase, and use these drives for analysis when the detector is not taking data. This approach allows us to add and maintain tape drives without affecting other parts of the system. It also provides for separate management of each of the major systems involved; DA, Farm, Analysis, and Mass Storage. We feel this will provide a more reliable and easier to maintain overall system. One disadvantage of this network based approach is that CPU is consumed moving data around. There is an especially large CPU overhead required for the receiving end. However we have studied this issue and are comfortable with the amount of CPU needed for the data throughputs we require.

The working model for the DØ analysis computing hardware is depicted in Figure 2. This shows the tape robot with several mover node machines. Each mover node will have a few locally attached tape drives and network connections. These movers are attached via the network to the on-line Data Acquisition, Reconstruction Farm, and off-line Analysis systems. Data-moving resources will be dynamically allocated to the various needs using the Enstore and SAM management tools.

Over the last 6 months, a joint Run II working group called the Physics Analysis Hardware (PAH) group has been meeting to understand the needs for the FNAL contribution to the analysis hardware. The goal of this effort is to produce procurement plans for hardware to be purchased in FY99 which will represent the first phase of the total analysis system. This group has had visits from vendors including SUN, DEC, and SGI, and has made visits to various vendor and user installations. A Request For Information (RFI) was sent late last year, and the results have helped DØ understand the CPU needed for our anticipated network transfers.

5.2 Serial Media Working Group

The group to evaluate the mass storage media issues for Run II, called the Serial Media Working Group (SMWG), was given its charge in late 1997. They looked at many options for drive and tape technologies which might be viable for Run II and have reported their conclusions in [11]. Some preliminary testing was performed on each of the drives considered to better understand performance and features. Among the items investigated were drive cost, media cost, physical size and capacity of each cartridge/cassette, bandwidth of reads/writes, and the stability of the technology. Because tape technologies were evolving, this group decided that it was prudent to defer the actual decision as long as possible, and suggested a choice be made in the spring of 1999. With this in mind, they recommended that a flexible robotic storage solution be purchased in FY98, and this has led to the acquisition of the first EMASS robot, which has been installed and is now being tested.

The SMWG has recently been reformed with a new charge and a few new members to take the final step of choosing the Run II tape media technology. This group will review the information already gathered and look at next generation of products which have arrived in recent months. Also, tests which were performed over the last year will be re-examined and further evaluation will be performed. Among the issues to be understood and quantified are: 1) head life and failure characteristics, 2) media life and failure characteristics, 3) interchangeability of tapes in a system with many 10's of drives, 4) mechanical weaknesses of drive designs, 5) failure modes such as SCSI resets or other firmware issues, and others. These tests will be performed in the remaining time before the decision has to be made as closely approximating the types of use and continuous duty factors as possible. For DØ, types of use include appending files to tapes using more than one drive, and serial as well as random access of files from tapes.

Other considerations will be included in the decision. One factor is the need for an exchange media between FNAL and our collaborating institutions. Although the choice for this purpose could be different from the choice made for mass storage at FNAL, having the same media would be a much easier approach. This factor, among others, has motivated the SMWG to lean toward lower cost commodity drives, such as Mammoth or Sony AIT, and away from more expensive data center type drives such as Redwood, Ampex or Sony DTF. Another consideration is the actual cost of operating a fleet of drives, and the on-going cost of replacing or repairing failed components.

5.3 Data Access

The Production and Data Access Group is responsible for delivering a data handling system which will provide rapid, convenient, and well tracked access to data. This must be well matched to the evolving priorities and phases of the experiment and must make optimal use of the finite and limited resources of tapes, tape drives, robot arm and shelf space, disk, cpu, and network bandwidth. The primary objectives are to store reliably all raw data produced by the online system and by Monte Carlo simulations, and to provide access to the data for all subsequent stages of processing, such that physics results can be extracted as quickly and efficiently as possible.

The Computing Plan of January 1997 [1] identified the scope of the Reconstruction Processing, Analysis Processing, and Monte Carlo simulations which will be needed for Run II. The scale of the data management effort which accompanies these processing tasks is set both by the anticipated size of the dataset and the number and diversity of the expected analyses. The DØ data set is expected to total at least 560 TB. Of this, 300 TB will be RAW data from the DAQ system and the remainder is needed for reconstructed and subsequent analysis stages. This plan also described a general view of the data handling system: one which will use a tape robot and tape storage for most data, cache frequently used data on disk, utilize standard Farm systems for Production processing, and rely on a large Central Analysis Server.

5.3.1 The Sequential Access Model

During the early part of 1997 much work was done, largely in the context of a Run II Joint Computing Project working group, to further refine the computing needs for Run II. The Data Management Needs Assessment [12] report resulting from this study was published in June of 1997 and along with the basic assumptions from the DØ plan, outlined above, provided input to development of a DØ model for organizing and accessing data. At the summer workshop of 1997 the experiment decided to use Oracle as its central and primary database management system; the file and event catalog and production tracking database therefore were to be implemented using Oracle.

The DØ data access model, or data management strategy, was also developed largely in the context of a Run II Joint project working group: one of 3 working groups convened in the area of Data Access. The model shares many features with the data management approaches used in previous runs and employs a file-based system of storage and bookkeeping. For this reason it has been called the "Sequential Access Model" or SAM. A detailed report on the Sequential Access Model was prepared for the October 1997 Von Ruden Review of Run II Computing [13]. The key features of this model are:

- Event Data are provided in files of event data units (EDUs) of some size and type. Data are expected to be accessed by reading sequentially through an entire file. The system is optimized for this type of access.

- An event is the logical storage unit which the overall data handling system catalogs and can arrange access to. Sometimes analyses are expected to want access to one particular EDU in a file.

- Access to the data will be through various different modes of access. There are 5 basic types of delivery needs anticipated: 1) File and event database scans, 2) Experiment shared random access disk, containing a small data selection for every event called a "Thumbnail", 3) Coordinated large dataset processing, referred to as "Freight train", 4) Single event picking, and, 5) Individual user on-demand file access. These modes of data access each have specifically defined requirements for mass storage bandwidth, disk staging space, processing CPU and the number of concurrent users.

The goal of this model is to optimize the use of storage resources such as tape mounts and drive usage. In order to facilitate this goal, we have four primary tools: 1) Clustering the data onto tertiary storage in a manner corresponding to access patterns, 2) Caching frequently accessed data on disk or tape, 3) Organizing file requests to minimize tape mounts, and 4) Estimating the resources required for file requests before they are submitted and, with this information, making administrative decisions concerning which data to deliver. In addition, it is desired to unload the burden of individual file tracking from the analysis physicists, and place it onto the data management system.

Clustering of data to optimize later access involves streaming the data directly from Level 3. Several possible streaming strategies were examined - exclusive streaming (an event is written to exactly one physical stream), inclusive streaming (some percentage of events, perhaps as much as 30% , are written to more than on physical stream), and a hybrid almost-exclusive strategy. The exclusive streaming strategy was the one proposed as having the greatest potential to provide the fastest and most resource-efficient access to the data. However it is not without complications with respect to

- Luminosity calculations

- Calibration data

- Providing Logical Streams of data to hide multiple physical data streams from the user

Streaming strategies were studied and discussed in [17] and [18].

A series of detailed studies were made on data access flows based on various patterns and volumes of data access thought to be likely in the different stages of evolution of the detector and data processing. This document included detailed spreadsheets and calculations, based on historical patterns of access and information gathered from various physics groups and individuals. It is called "Flows and Controls of the Seasons of SAM" [14].

5.3.2 Data Handling Review

In March, 1998, the DØ experiment held a Data Handling Workshop and Review to discuss and examine the proposed Sequential Access Model for data access. Several Physics groups were asked to carry out 'gedanken experiments' to explore in detail how particular analyses might proceed using the exclusive data streaming proposed and the specific data tiers of event data under consideration. A series of questions were developed for Data Access method and streaming scenario proponents. The result of this workshop was an endorsement of the proposed Sequential Access Model [19]. Streaming based on level 3 was deemed to be reasonable, as well as the proposals for the 3 data tiers EDU250, EDU50 and EDU5, and the various modes of access proposed.

However, there was some concern about the following issues and they require ongoing consideration:-

- ramifications of exclusive streaming, including effects of data loss and complexity of luminosity calculations
- the physics content of EDU50 and EDU 5, which remains to be finalized
- export of data to remote institutions

5.3.3 Luminosity Workshop

One of the more tricky issues associated with using an exclusive streaming method of data clustering is that of Luminosity tracking. The way in which streaming is to be done and the mechanisms for tracking data file event contents and luminosity have considerable effect on the Event/File Database design. In response to this issue, Heidi Schellman and Rich Partridge organized a two-day workshop on luminosity issues. A report from this workshop is available[20] as well as a more detailed discussion of a proposed method of normalization[21]. The responsibilities of the Online Data Logger to either record details about time-ordering of events, or possibly re-order events before writing to a Run Partition/Physical Data stream file, is still under discussion.

5.3.4 Data Handling Layers Implementation Strategy

It was understood that the full data handling system necessary to store and manage the data would require several components working together to provide a functional system. A goal of the Run II Computing Joint Projects was to identify and specify these components so that, as far as possible, common solutions could be developed for both DØ and CDF. With this in mind, several working groups and projects were instituted to look into requirements, define boundaries, and test possible solutions.

- RIP (Reconstruction Input Pipeline) to deal with the first phase of the data handling which involves moving data written to local experiment disk, by the Online system, to the Feynman center where it is to be archived and made available to the Farm Reconstruction system and to all subsequent stages of analysis and processing.
- Storage Management to provide for migration of data between disk and tape and for use of a tape robot.
- SAM Data Access to provide optimized and controlled access to data and to track the entire data creation, reconstruction and analysis process.
- Specification and development of a Production Management system for Run II, including Farm software and Batch system software.

Although DØ people are heavily involved in each of these projects and in particular in the successful integration and deployment of them at DØ, they are all being carried out under the banner of the Run II Joint CDF/DØ/CD Computing projects. In November 1998, a joint Data Handling Workshop was held to examine the prototype systems of both experiments and attempt to identify common ground. CDF's current data access model does not use SAM or ENSTORE.

5.3.5 Storage Management/ENSTORE

The Computing Division had gathered considerable experience in the use of the HPSS (High Performance Storage System) in the data center. As a result of this experience together with a detailed look at the requirements of Run II for storage management, including needs for reliability and placement of data on tape, an HPSS workshop was held in the Spring of 1998. Following this workshop a decision was made by the Run II Computing Joint Project Steering Committee to treat HPSS as a backup solution for Run to and to endorse the development of alternative storage management software [22].

The Run II Storage Management project is now focussed entirely on implementation of a Fermilab storage management system called ENSTORE, [23] based on a DESY design. DØ's requirements for Storage management are discussed in [24], in [15], and are further described in a presentation made at the November CDF/DØ data handling workshop [25] and a SAM interface document. These are being met by the ENSTORE system and testing of the prototype SAM system involves use of, and integration with, the ENSTORE system. The ENSTORE system delivers files over an IP network from robot resident tape volumes. All details of tape allocation, queuing, robot control and tape read/write error handling are managed by the ENSTORE system. Controls are provided to allow the Data Access layer to prioritize and re-prioritize the work in the ENSTORE queue, based on performance and operational statistics provided with each file transfer performed. The primary file/tape volume catalog is maintained by the ENSTORE system and is user visible via a pseudo NFS mounted pnfs file system. For performance reasons, and to allow queries spanning multiple facets of file and event attributes, a copy of the file/volume catalog is maintained within the SAM event and file database.

5.3.6 Data Access/SAM

The SAM Data Access project is building a prototype data handling system which 1) stores and deliver data files; 2) catalogs data files, data streams and events; and 3) records all processing steps and analyses in an Oracle database. The requirements document of June 1998 [15] remains the guiding document for the design and implementation of the SAM system. The architecture document of June 1998 [16] also serves as the conceptual basis for the detailed design and coding work which is taking place as a prototype SAM system is being built and tested. Additional detailed design work and specification of interfaces can be found in documentation and notes available from the SAM project web-site (http://RunIIComputing.fnal.gov/sam) [26]. This includes detailed design work on the entire database supporting SAM.

The recommendations of The DØ Global Computing Model Working group[2] to adopt a 3-tier analysis system architecture further validated the design criteria of SAM - to build fully distributed system supporting multiple heterogeneous analysis and reconstruction systems via network-attached tape drives.

5.3.7 RIP

The RIP project, which will use ENSTORE to write data to network-connected tape drives over an IP network, has become an extension to the Online systems projects. The way in which the data written by RIP is to be cataloged in the SAM Data Catalog has been specified and understood. The RIP requirements specified in October of 1997 [27] remain the guiding requirements, including those which deal with total rate, glitch buffer and the placement of files on tape according to physical data stream. The decision to use ENSTORE and IP network-connected tape drives for this part of the data handling system means that Gigabit Ethernet technology can, and has, been chosen as the network technology for the interconnection between the DØ experiment and the Feynman Computing Center for Data Logging.

The Files written from the online system through RIP will be written using ENSTORE and also cataloged in the SAM event and file catalog. The online data logger will provide the meta-data required by SAM for each file written, including the event ranges/timestamps for events in a file. In addition the online system will provide information about each event in a file, including its trigger and filter bits, timestamp, and data stream. [28]. This information will be used to construct the Event and Data Stream parts of the SAM Event and File catalog.

5.3.8 Production Management/Farms/Batch

A Farms Batch system design [29]and a prototype system for use has been produced by the Production Management Run II Project. There are two possible models of use of this system under discussion which integrate DØ production code, SAM, and the Farm system.

A Batch Working Group [30] has been set up to examine the general requirements for a Batch system for Run II analysis processing. The interaction between the SAM data delivery system and a Batch system used to control such resources as CPU time and scratch disk, is being examined in the context of this working group. The working group will report by March and identify further work which needs to be done. The LSF system is the primary contender for a Run II batch system,and we will either extend its features, if necessary, or in the worst (unlikely) case, look for another batch system.

5.3.9 SAM and ENSTORE Prototype Data Handling System

A prototype SAM system is being constructed based on a suite of CORBA-based Servers which manage communication with the analysis/reconstruction programs and delivery of files to those programs. The Servers include:

- Project Master - to manage file delivery to multiple consumers for a single project
- Station Master - to manage the disk, file system and network resource available to a single 'Station" and to control startup and access to Projects
- Stager - to handle interactions with the Enstore system
- Optimizer - to control and optimize flow of file staging requests to Enstore
- Logger, Information Server and Database Server - to manage internals of SAM and provide information

All the following Client interfaces have version 0 interfaces: importing files, specifying file meta-data, specifying an analysis project, starting data delivery for an analysis project, and consuming data files. CORBA IDL interfaces have been defined and code has been written in C++, Java and Python to provide the client functionality.

A full-featured database system has been designed to track events, files, and processing information. Version 0 of this database has been implemented as an Oracle database.

A standalone test system has been put together to facilitate testing of SAM and ENSTORE and to simulate delivery of data to multiple network-connected 'stations' and to multiple projects, each with many analysis user programs requesting access to the delivered data. The SAM test systems (samson, delilah, sammy, samadams, samting, sameggs and samham) are rack-mounted PCs running RedHat Linux and are connected to a similar ENSTORE PC teststand via a gigabit-capable switch connection. The node samson also serves as an ENSTORE system local test node to move data files between SAM and an ENSTORE storage system: in this case, a disk based system capable of storing up to 18 GB of files. The gigabit connection between test systems has not been available until recently and so no high-rate tests between systems have yet been conducted. The ENSTORE system has demonstrated movement between Sony AIT tape and client disk at full tape speed rates however.

Details of the design and work of the ongoing prototype SAM can be found on the SAM home page [26].

5.3.10 Milestones and Future Plans

October 11, 1998 - Functioning Prototype for Single Project This milestone was met and a test consumer process was able to obtain files from a SAM project,managed by a Project Master server process. The SAM servers were able to communicate with ENSTORE servers which were used to store and retrieve files from a pseudo-robot archive. Shortly after this the prototype system was exercised with ENSTORE servers and movers connected to the STK robot and Redwood tape drives.

November 9, 1998 - Demo of SAM at Supercomputing 1998 in Orlando The SAM PC teststand was operational for this milestone and we were able to demonstrate running multiple different projects each fetching data files from an internal SAM Enstore system on node samson. Several 1GB data files were used as well as a number of few MB datafiles. Up to 30 different analysis programs simulataneously requested delivery of the files from a project - each simulating a wait to analyse the file and randomly creating error conditions whereby the file was not successfully marked as processed. Details of the project's definition and files and the consumption of these files by each analysis process were kept in the database.

November 30, 1998 - SAM Prototype Connected using ENSTORE to Grau Robot This milestone was achieved by December 21st 1998, although only brief tests were conducted at that time.

December 30, 1998 - Documented Prototype for Monte Carlo File Storage and Retrieval This milestone is late although the system is actually capable of storing Monte Carlo files. Documentation exists for specifying the meta-data associated with a Monte Carlo file, including all its parent files and parameter files.

January 31, 1999 - Version 0 of Command Line User Interface and Farm System Integration The V0 interface is documented and largely implemented. Integration with the Farm system is underway.

February 3-5, 1999 - Performance tests/24 hour tests with ENSTORE. Demonstrate system at DØ collaboration meeting.

February 15, 1999 - Version 0 Production Event/File Database ready for use. Farm-mode tests with the Farm Batch System (FBS), SAM, and Enstore.

February 28, 1999 - Version 0 User Guide/File Import area on delilah

March 30, 1999 - Version 0 Integration with DØ Framework. This includes porting and testing of all user code to SGI machines and full integration with SoftRelTools.

April 15, 1999 - Version 1 of User guide

April 30, 1999 - Simple Pick Event capability, Tape Import capability

May 30, 1999 - Design of Station, Optimizer and Interface to Batch System complete

June 15, 1999 - V1 Event and File Database

June 15, 1999 - Begin Import Run I data into SAM

July 31, 1999 - V0 Implementation of Station, Resource Management and Batch Interface

September 15, 1999 - V2 Event and File Database

September 30, 1999 - V1 SAM Servers and Resource Management software complete

November 15, 1999 - V0 design for EDU5 (Thumbnail) data access

December 15, 1999 - V2 SAM system including Pick Event and Thumbnail access complete

5.4 Databases

5.4.1 Databases Strategy

The Databases group (leaders: Vicky White and Lee Lueking) exists in order to coordinate much of the Database work, which will take place in the context of several other areas of DØ Run II Computing. The plan is to use a central Oracle database, one for the Online and one for the Offline system, and to provide for uploading and synchronization of data between them. The Databases group tries to ensure that a common approach is followed in designing the database and database applications. They disseminate information and common tools and provide examples of ways to access the database. Work in the area of the DØ persistency mechanism DØOM also involves databases. Part of the central Oracle database must be exported to remote institutions, where Oracle may not be available. It is part of the charge of the group to propose and provide the mechanisms for this.

The group provides the database infrastructure needed, largely through liaison with efforts of the Support Databases Project, which is part of the Run II Joint CDF/DØ/CD Computing effort.

A Sun 3500 and a Sun 4500 Server machine have been purchased and are in the process of being installed. These machines will serve respectively as the production database machine and a development and test machine for the next year. Following extensive testing using the 4500 machine, we will be in a position to understand the CPU capability and I/O capability needed for the offline database machine.

Meetings of the Databases group have been infrequent until now with initial meetings held mainly to establish the candidate database applications and timescales. In the past two months people have been starting to think about their database issues and regular monthly meetings of the Databases group may soon become appropriate.

An overview of the DØ Database Strategy was presented in a poster paper at CHEP 98 - http://www-d0.fnal.gov/ white/d0db/ChepPoster-Oracle.ps .

The group maintains a web page at http://www-d0.fnal.gov/ white/d0db and a users page at http://d0db.fnal.gov/d0db.

5.4.2 Databases Candidate Implementations

Nine different candidate areas requiring a supporting database have been identified. These are:- File and Event Catalog, Run Conditions, Calibration (Online and Offline), Geometry, Alignment, Triggers, RCPs, Luminosity, and Hardware Addresses.

Work to specify the nature of the database and the way in which it is to be used is taking place in some of these areas, as described below. In other areas this work has not yet begun.

5.4.3 File and Event Catalog /SAM database

The SAM database has received most of the attention and therefore leads the way in providing expertise and tools for the other areas. Detailed analysis and design work has been done using the Oracle Designer tool to document the Entity/Relation diagram of the database and generate the tables. In addition a Database Server approach has been used, based on CORBA IDL interfaces and this is forming the basis for a core database server, which may underlay specific database servers in each of the other areas.

5.4.4 Support Databases Joint Project and Common Tools

All of the work which has taken place has been supported by, and is in cooperation with, the Support Databases Joint project. Members of the Support Databases project work closely with DØ people to specify the requirements and design the database and database applications.

A list of potential common tools, not only for DØ Database Applications, but also for CDF use, is being developed as part of that project and many such tools may be used in support of the Database Applications and in the DØ Persistency package database back-end work.

5.4.5 Trigger Database

A database for tracking trigger configurations, statistics, and more, is being specified by Carmen Silva and Amber Boehnlein from DØ, working with Support Databases member Barbara Angelos. They are nearing the end of the design phase and will shortly have Version 0 of a Trigger Database.

5.4.6 Alignment Database

V0 of an Aligment Database has been developed by Support Databases member Julie Trumbo, working with Al Ito from DØ. Al is able to perform all uploads of alignment data from his NT PC desktop using Excel and Microsoft Access to front-end the Oracle database.

5.4.7 Hardware Database

Work is being done by Stan Krzywdzinski from DØ on the design of this database.

5.4.8 RCP Database

Marc Paterno and Harrison Prosper, working with Jim Kowalkowski, Vicky White and Steve White (from Support Databases project) are in the process of designing and implementing the RCP database.

5.4.9 Calibration Databases

Ela Barberis and Vicky White are working on an initial prototype implementation of an offline calibration manager and database. This exercise is also being used to understand which extensions to the DØ OM package are either necessary or desirable to support database persistency.

5.4.10 Milestones and Future Plans

There are few specific milestones for this area since they are either related to the Support Databases Joint project, for database infrastructure needed, or they depend on manpower from other areas of DØ Run II Computing being ready to start work on their database problem. However, good progress is being made in those projects which are active and this should serve to lighten the work needed in other areas.
January 15, 1999 - V0 prototype for Offline Calibration Database and Offline Calibration Manager.
March 15, 1999 - V0 of RCP Database
April 15, 1999 - V0 of Trigger Database

Calibration Databases need to receive the most attention, after the Event/File Database for SAM. They will have to hold and provide fast access to significant amounts of data.

5.5 Data Access and Databases Open Issues

A number of issues remain open and finding solutions for them has not necessarily been included in the list of Milestones. Further experience with using the systems for Monte Carlo data and possibly Run I data may help to resolve some of these.

- Physics Content for EDU50 and EDU5. Decision on whether different streams may have different physics objects.
- File format for EDU5. Understanding of where ROOT file format or n-tuples begin and DØ OM file format ends.
- Need for a freeware database and ability to extract part of the file/event catalog and suitable calibration, geometry and alignment data for a particular dataset. MSQL has been talked about and tried, but with the advent of Oracle on Linux PCs and the prevalence of Microsoft Access databases on NT machines, MSQL may not be the obvious choice.
- Confirmation and testing that the Oracle file and event catalog can perform well with 2 billion events and several million files and that trigger and filter bits can be efficiently stored and selected.
- Further work on exclusive streaming with respect to data delivery and luminosity.
- Splitting event information into distinct physical storage locations, or sub-event level streaming. In other words, various parts of each event are clustered into physical storage providing more efficient access.

6 Online System

This section describes the hardware and software architecture of the online system in Run II.

6.1 Functions

The principal functions of the DØ Online Computing system are:

- Detector hardware control and monitoring;
- Primary (secondary) event data acquisition at rates up to 50 Hz instantaneous with 250 kbyte/event (20 kbyte/event) data size;
- Event data monitoring;
- Control room activities.

6.2 Hardware Architecture

6.2.1 Functions

The hardware design of the Online system is dictated by the various functions to be performed. These functions and the different hardware strategies suggested are summarized in Table 1 *Hardware System Functions.* The complete specification of the requirements of the hardware components of the Online system is dependent upon the software architecture. The following Design section assumes some of the schemes to be discussed within the Software Architecture component of this document.

System Function	System Requirements
Database operations	High availability
Primary event data acquisition	High availability High I/O bandwidth High network bandwidth Large application memory buffer space Large I/O disk buffer space
Secondary event data acquisition	Moderate network bandwidth
Detector control and monitoring	Embedded processors GUI interfaces Isolated network (security)
Event monitoring	High network bandwidth High CPU capacity
Control room applications	GUI interfaces

Table 1 Hardware System Functions

6.2.2 Hardware Design

To supply the previously listed functionality, the computing hardware design includes the following components:

- Gigabit Ethernet Network;
- Central UNIX Host System;
- Control Room PCs;
- Monitoring PCs; and
- Front End Processors.

Figure 3 illustrates the design. Each of the basic components is described in the following sections. Note that there is no tape writing capability (other than for backups) local to the Online system. Completed event data files of approximately 1 Gbyte size are sent over the network to the Feynman Computing Center (FCC) for logging to tape.

Gigabit Ethernet Network The network backbone is a multi-port Gigabit and 100 Megabit Ethernet switch. Only the UNIX Host nodes and the link to the FCC are at full Gigabit speeds; other devices are connected at 100 or 10 Megabits.

- Existing Run II Online host nodes are currently connected to the legacy Online FDDI ring.
- A few 100 Mb capable network devices are in hand:
  - 3COM 12-port 100 Mb hub
  - Allied Telesyn 16-port 10 Mb switch with 100 Mb uplink
- Requisitions are currently in Purchasing for 100 Mb network components for the Moving and Fixed Counting House areas:
  - (2) 3COM Model 3300 24-port 100 Mb switches with 100 Mb fiber interlink modules optional Gb uplinks

- (3) Allied Telesyn 24-port 100 Mb switches
- (7) Digi 16- and 24-port terminal servers, with fiber transceiver modules for 3 units to be located on the Platform.

- The single-mode fiber installation between DØ and FCC is complete and has been tested at 100 Mb speeds. The single-mode fiber will allow eventual Gb transmissions.
- The Computing Division / Network group will install in January a temporary Catalyst 2948 switch with 2 Gb ports and 48 100 Mb ports. This is a precursor to an eventual multi-port Gb switch purchase.
- Plans are being drawn for the network cabling infrastructure in the FCH and MCH areas. The general scheme is for a 100 Mb switch and a terminal server to be located in each floor of the MCH, in the Control Room, and in the FCH2 computer room areas. In the MCH, multi-UTP cables will run to several patch panels distributed throughout the equipment racks; individual devices will be connected to the patch panels.

Central UNIX Host System A set of Compaq/Digital AlphaServer UNIX servers is chosen to perform those activities which are most demanding in terms of availability, I/O bandwidths, network bandwidths, and process memory space. Thus these servers are the hosts for database and the main event data acquisition activities. Note that the central Host system is not a high capacity CPU resource, having only the database access and data transport requirements. The servers are arranged in a cluster fashion, with a shared RAID disk set housing the databases and the important Online applications. The cluster also allows rapid failover of applications from one node to another. High availability is achieved by using redundant controllers and power supplies in the shared RAID arrays and by having redundant Host nodes which can singly provide all Host functions. The Host servers each have local large high speed Ultra SCSI disk arrays for event data buffering. They additionally support high speed Gigabit Ethernet network interfaces. Two of the three eventual Host servers have been purchased (as of 1/10/1999). They are both model AlphaServer 4000s, each with four independent PCI busses providing a total of 16 PCI peripheral slots per node. In practice a PCI bus will likely be dedicated to serving the Gigabit Ethernet interface, with the other busses used to support disk controllers. The AlphaServer 4000 can support two CPU boards; currently each has only a single processor.

A Host UNIX server has been purchased in each of FY97 and FY98. The nodes, associated peripherals, and installed software sets are:

- Node d0ola
  - Digital AlphaServer 4000, 5/466, 512 MB
  - Digital UNIX V4.0b
  - FDDI NIC active, 100 Mb NIC installed
  - 1 local device shelf with approximately 13 GB of disk.
  - HSZ50 RAID controller (the eventual shared RAID set) with 2 device shelves and approximately 108 GB of disk.
  - PCI-based RAID controller with 3 device shelves and approximately 54 GB of disk (yet to be configured).
  - DEC C and C++, Fortran, and Kai C++ compilers

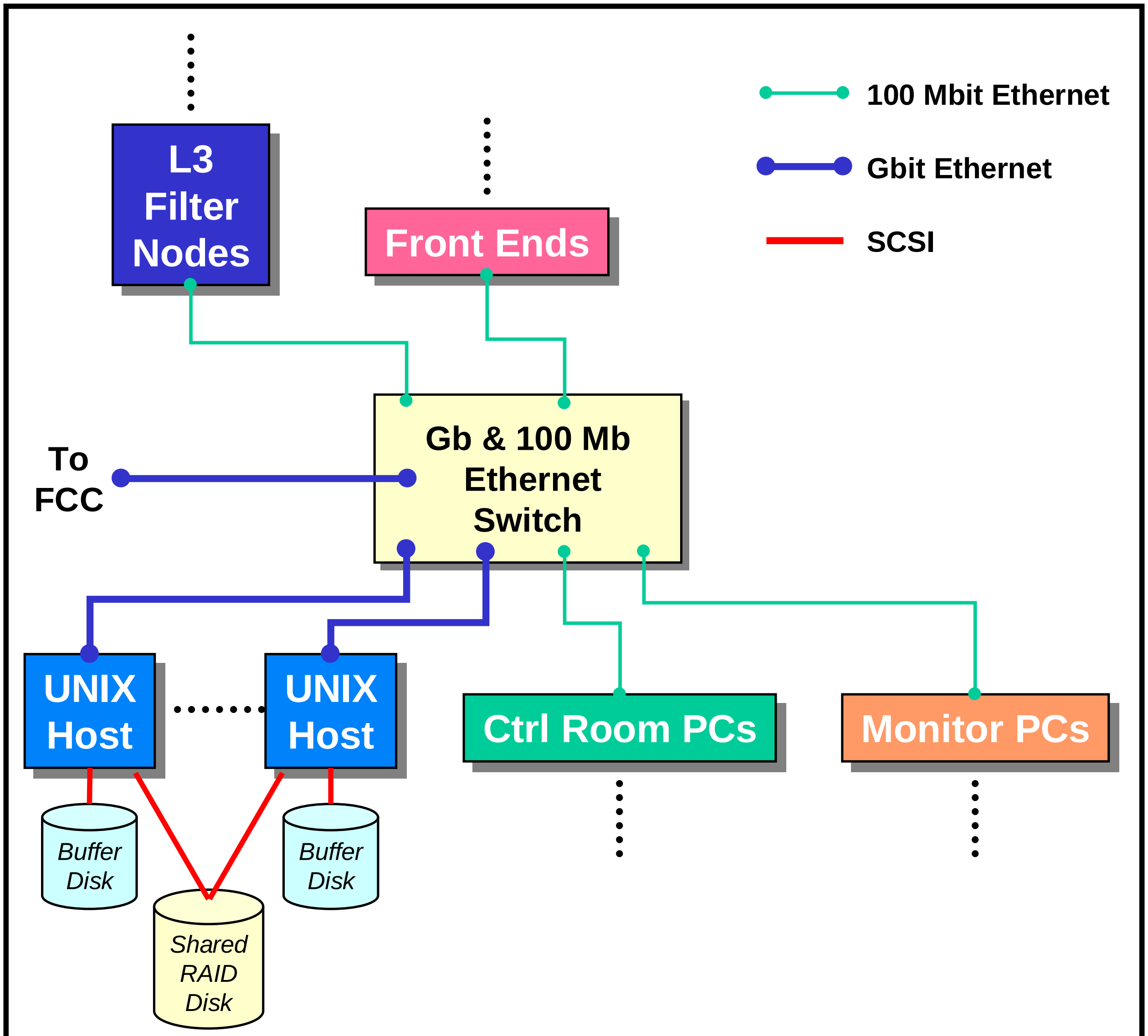

Figure 3: Basic Hardware Components of the Online System

Control Room PCs A number of computing stations are required for Control Room operations. For this function we have chosen to use commodity PCs running Windows NT. The PCs provide an inexpensive and easily manageable platform. We have chosen Windows NT both because of the familiarity of DØ personnel with its interface and because of our growing familiarity with its programming environment, particularly as related to the Level 3 software development effort. As will be noted in the Software section, the Control Room nodes are typically running GUI client applications which communicate with external server processes. We are developing the GUIs using a suite of standard (e.g. Python) and DØspecific (Client / Server) packages. The use of the NT platform is restricted to these user-interface applications.

There are 2 Pentium II / 450 MHz / WinNT PCs in the control room which are awaiting connection to the network. There are 2 older PCs purchased for the Online system available.

Monitoring PCs The real-time monitoring of the event stream by partial reconstruction and analysis is a highly CPU intensive activity. For certain types of analysis, it is also a network intensive activity. The software to perform these analyses is typically a variant of the Offline reconstruction and analysis software, and hence development and operation benefit from a similar environment. We have chosen to meet these needs by the use of a collection of PCs running the Linux operating system. We choose PCs because of their excellent processor price/performance, with expected continued improvement. We use a collection of PCs to divide the monitoring duties and network loads among a number of processors. Finally, we choose Linux in a pseudo batch environment because of the similarity to the Offline reconstruction and analysis environment.

One PC, node **d0linux01**, has been established as our prototype Linux node. The UNIX host systems nfs serve software product disks for both Digital UNIX and Linux flavors.

Front End Processors There are three classes of embedded processors in use at DØ with either hardware, readout, or trigger control functions. The trigger processors are detector type specific, and are the responsibility of the detector hardware groups. Other than the support for network connections, downloading, and remote access, they are not considered part of the Online system documented here. The processors situated in the detector readout crates have several functions. During normal event data taking they can parasitically access the detector raw information and perform low-level data quality monitoring. During dedicated calibration periods, events are directed to the local processor for determination of the calibration constants. The local processor does not participate in normal data acquisition, but a debugging and commissioning mode allows for readout of the VME crate and data transmission to a host receiver. Finally, the embedded processor serves as the remote link to the VME crate address space, and manages the configuration and downloading of the devices within the crate. Most of the functions of these processors are determined by and developed by the individual detector groups. However, the Online Computing group provides much of the infrastructure for the operations and software aspects of these nodes. The third class of embedded VME processor is dedicated to control and monitoring of the detector elements, including for example low voltage, high voltage, and environmental conditions. These processors are an integral part of the Online Control System, and are managed and programmed as part of the Online Computing sub-project.

All existing front end processors are of the Motorola 68K family. There are currently approximately 5 front end nodes installed in test stands at DØ and the detector test areas. A purchase order for 3 additional front end nodes is in process. We expect to order soon several PowerPC processor family nodes for development. We also expect to order soon nodes to be placed in the MCH.

6.3 Software Architecture

6.3.1 Component Introduction

The software components of the Online system provide the following required functions:

- Shared software infrastructure;
- Hardware control and monitoring;
- Event data acquisition;
- Event monitoring; and
- Calibration.

Each of these components is discussed in more detail in the following sections.

6.3.2 Shared software infrastructure

There are several software packages that are used by numerous higher level applications. These are:

Database The database management system of choice for the Online system is ORACLE. Within the DBMS will be kept several independent databases for:

- EPICS control system parameters (Hardware Database)
  The Run I HDB RDB database has been imported into an ORACLE database on the fnalu cluster, and some ORACLE operational experience is being obtained. ORACLE has not yet been installed on the Online UNIX hosts, but there is a machine now available at the appropriate OS version level.
- Detector configuration sets
- Trigger configuration sets
- Run and data file information
- Calibration information
- Detector monitoring information
- Luminosity information

The tools being developed for managing the DBMS and the entries within the databases will be shared among the above activities.

Inter-task Communication The distributed, real-time nature of the Online software applications dictates the need for efficient and reliable communication among cooperating tasks. DØ has chosen to address this need by creating a C++ Client / Server library, providing a common means by which applications can exchange messages. This package, called d0me, is built upon the publicly available ACE product. We have chosen to use the messaging paradigm as opposed to a distributed object paradigm, as would for example be supported by a CORBA implementation. We believe that the messaging abstraction fits well our application environment. The detailed operational aspects of the socket-based messaging package are also easily controlled and monitored.

The d0me Client / Server package has been through several iterations of release and evaluation, and is generally available for use. The user-level API is expected to be stable within a week. Example applications are distributed as part of the package, and several applications have been built to date. A Python wrapper for d0me has been created, though needs slight modification to match the latest d0me changes.

6.3.3 Hardware Control and Monitoring

With a wide range of hardware device types to control, and a need for numerous custom interface applications, it is clear that there are a large number of software packages and applications to be developed. The DØ Online group has chosen to use the EPICS control system package as the basis for our control system. Upon this base are built a number of additional applications for specific DØ purposes. DØ uses the Python scripting language to build many of these applications.

Control and Monitoring effort to date has been centered on test stands, either specifically for use in software development or in support of detector test activities in Labs B/D/3.

The package and application list follows.

Hardware Database EPICS makes use of distributed database-like files on the front ends to characterize specific device properties. DØ has chosen to derive these distributed databases from a central ORACLE database instance, denoted as the Hardware Database. Into this database goes information for each hardware component within the detector to be accessed via the control system. The database includes access (name to address translation), field bus parameterization, data transformation (into natural units), and alarm limit information.

The EPICS control system uses a distributed database to define devices and records. To date, these flat files have been created by hand. The goal is to create these automatically by extracting the information from an ORACLE database. As noted above, the Run I database has been imported into ORACLE. Efforts are in progress to define ORACLE tables with the relevant information for EPICS and to load specific device information into the database.

VME Downloading Numerous VME-resident hardware components need to be configured or otherwise downloaded in preparation for data acquisition activities. Access to the VME memory space is provided via EPICS devices. EPICS controlled access to registers within the VME address space has been demonstrated.

Rack Monitors Environmental monitoring for the detector electronics is provided by a standard Rack Monitor device. Software support for the Rack Monitor includes the EPICS device definitions and a GUI application interface. The Rack Monitors are specific examples of devices reachable over the 1553 bus, for which DØ has developed EPICS device and record support.

A Python based GUI making EPICS calls to display a Rack Monitor properties has been written and demonstrated.

Low Voltage Power Supply Control The remote control and monitoring of the Low Voltage power supplies is again achieved through the use of EPICS as the layer below custom Python graphical user interface applications.

En EPICS based prototype LV PS control program exists. Work is in progress to enter database information for test devices, and a GUI application is being developed.

High Voltage Power Supply Control The software developed for High Voltage supplies supports higher level complex operations, for example Ramp, in addition to the simpler On/Off/Read operations. The GUI interface is also more complex in order to manage effectively the hundreds of HV channels within the detector.

Design of an EPICS based HV PS control system is progressing.

Clock Control The DØ Clock synchronizes the detector to the accelerator cycle. The Clock can be configured, via EPICS control, to provide various signals to detector elements.

Software utilities to be accessed within a front end for control of the Clock are being developed and debugged. Once available, they will be incorporated into an EPICS device definition. A GUI application for Clock control will follow.

Alarm System The DØ detector is continuously monitored for failing components. The bulk of the monitoring information is acquired by the embedded processors, either by accessing the local VME hardware or by connection to devices on the 1553 auxiliary busses. Cooperating software applications may also generate alarms to indicate abnormal conditions. The Alarm System contains a number of software components:

- Alarm Server
- Alarm Display
- Alarm Logger

There is also a connection between the Alarm System and the Run Control system, allowing interruption of data acquisition upon notification of error conditions. An early version of a requirements document has been produced. The design of the Alarm system will closely follow that used in Run I.

Secondary Data Acquisition It is possible to use the Control path to perform data acquisition from a limited number of VME crates via their associated Front End processors. This secondary DAQ system uses software within the Front End to access the data buffers over the VME back plane, then transmits the information to an Event Builder application which collect data from multiple sources. The Event Builder passes the events to the Collector / Router application within the primary DAQ path for further logging and monitoring. Much of the Secondary DAQ path is built upon the Fermilab DART product.

The DART based Secondary data path is in common use in the detector test stands. Plans are being developed for extensions allowing multiple-crate readout and for merging the DART run control and event data paths with the Primary data path components.

EPICS Extensions There are several general extensions to EPICS which need to be performed in support of some of the aforementioned Control and Monitoring applications. These include:

- Device specific records
- Alarm broadcast modes
- Vector generalization of process variables
- Hierarchical process variables.

A preliminary examination of requirements has been made.

6.3.4 Data Acquisition

There are three aspects to Data Acquisition:

- Detector configuration and Run Control;
- The Event Data Path; and
- Trigger and DAQ Monitoring

The principal software components of each are described below. In addition, the event data path is illustrated in Figure 4.

The efforts relative to the event data acquisition needs have been centered on prototype development. The integrated working of the data acquisition system is one of the major near-term goals.

Configuration and Run Control A single application, COOR, is responsible for establishing and managing the configuration of the detector, trigger systems, and data taking run status. This application communicates with almost every other Online sub-system: it accesses pre-defined configuration sets in the database, commands detector configuration and downloading, controls the trigger system, initiates and terminates runs, and makes log entries into the database. Associated with the COOR application is TAKER, the user control interface application.

A Python based prototype of COOR has been created. Some of the protocols for communication with Trigger system components have been established.

Download Manager COOR's interface to the detector hardware system is via another application which acts to translates COOR's logical configuration requests to the appropriate control system actions. Embedded within this download manager are the detector-specific functions for downloading calibration parameters, as well as for general hardware configuration.

Work on a Run II version of the download manager awaited the availability of both the Client / Server framework and hardware control via EPICS. With both now available, development of a prototype will soon proceed.

Collector / Router The Online system, with scope as defined in this document, sees data events as originating from the Level 3 filter nodes. The first application in the event path within the Online system, hence the target for Level 3 transmission of events, is the Collector / Router. There are multiple possible instances of the Collector / Router, so as to provide parallel data paths supplying a higher aggregate bandwidth. Data events are passed on to the appropriate Data Logger, as determined by the event trigger characteristics, and to one or more instances of the Data Distributor. The role of the Collector / Router in the event data flow scheme has been examined, but no application design or coding has begun.

Data Logger The Data Logger is where events are assembled into files on disk. DØ has chosen to produce nearly-exclusive data streams at the Online system as an aid to post-reconstruction data access. The Data Logger(s) are thus managing multiple data streams and building multiple simultaneous data files. As with the Collector / Router, the Data Logger is designed to be run with multiple simultaneous instances, where each instance would be associated with a subset of streams. The Data Logger is responsible for synchronizing data file creation with database entries associated with accelerator luminosity recording.

A prototype Data Logger has been created. The next step is to integrate the prototype with the other components of the data acquisition path.

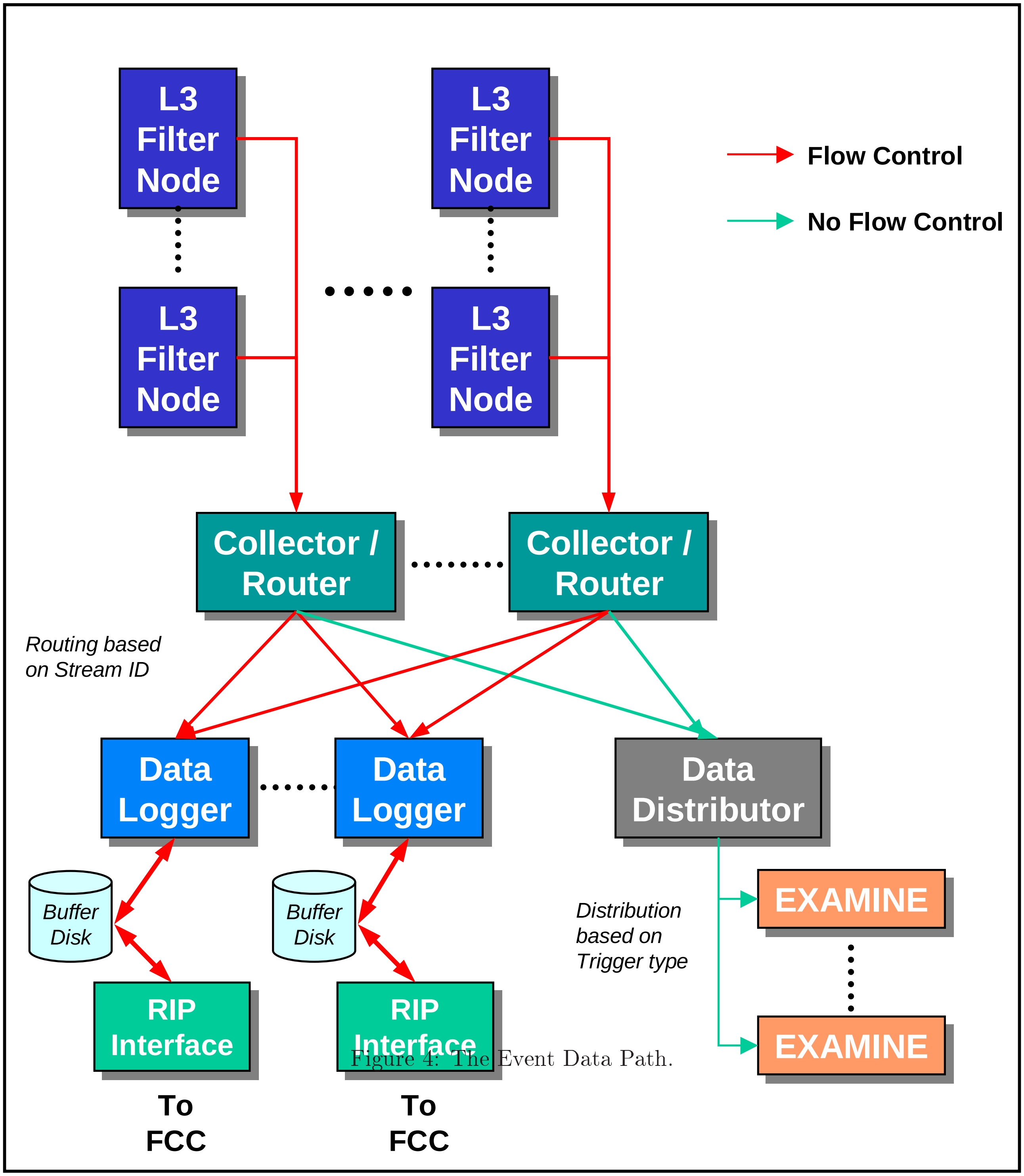

Figure 4: The Event Data Path.

Reconstruction Input Pipeline (RIP) Once completed data files are assembled by the Data Logger(s), they are transmitted over the network to the Feynman Center for logging to serial media. Data management at Feynman is part of the RIP project. The Online interface to RIP provides the method for transmitting the data files and for monitoring the transmission. Work on RIP proceeds as part of the joint CDF / DØ / Computing Division project.

Trigger and DAQ Monitoring With numerous trigger levels, possible simultaneous runs, and distributed and parallel data path components, the operation of the Online data path is quite complicated. There are a set of monitoring applications connected to each independent trigger and data path component, and an integrated monitoring application devoted to summarizing the activity of the system. Little has been done to date, but work should now proceed rapidly with the recent hire of a physicist to be dedicated to this task.

6.3.5 Event Data Monitoring

In addition to the lowest level monitoring of the detector hardware, it is essential to monitor the quality of the data acquired. To do this, event data is distributed parasitically to a set of processing nodes. The components of this activity are described in the following sections.

Data Distributor Events originating in the Level 3 nodes are distributed by the Collector / Router applications to the Data Loggers and to one or more Data Distributor applications. The transmission from Collector / Router to Data Distributor is by broadcast or on a best effort mode, such that the state of the Data Distributor has no possible effect on the successful smooth operation of the data logging path. The Data Distributor receives events in an unbiased fashion, and can further distribute such without bias. Event consumers, as described below, subscribe to specific subsets of events as determined by trigger type. The Data Distributor maintains a queue of events for each consumer, transmitting the next event upon request. The Data Distributor is also responsible for distributing the run state information to the consumers. Requirements and design documents have been created and coding has begun.

EXAMINE The ultimate data event monitoring consumer application is EXAMINE. There are many EXAMINE varieties and instances, where the varieties monitor different functions or trigger sets and the instances can distribute the load and pool results. The EXAMINE applications are basically reconstruction programs, sharing the same structure and much of the same code with the Offline reconstruction and analysis programs. The typical output of an EXAMINE is a set of histograms or plots, which can be remotely viewed by a browser portion of the application. We have studied both HistoScope and ROOT as the underlying analysis and display tools. HistoScope will be used in the short term, with an expected straightforward migration to ROOT. Demonstrations of both products on the Online computing platforms have been made. Prototype analysis programs using the DØ Offline analysis framework have been written, but integration into the real-time environment awaits improvements to the framework.

6.3.6 Express Line

In addition to the real-time monitoring supplied by EXAMINE, it may be necessary to provide a farm of processors to perform more complex reconstruction and analysis operations. The Express Line will serve this function by acting as a small local batch farm, operating upon (parasitically copied) event data files. This is a future activity, not undertaken yet.

6.3.7 Calibration

Calibration is very detector specific. Some calibration activities occur on local Front End processors, others within Level 3, and still others occur at the Host or even Offline. These various activities are coordinated in two locations - the databases containing the calibration information, and the Calibration Server, which acts as an interface between the user and the various sub-systems. Many of the details of calibration are still in design phase. Calibration activities occurring in the Front End processors will be very similar to the local monitoring activities and the Secondary DAQ path operation. Data from the local crate will be asynchronously collected and analyzed, with the results made available via Control System calls. Level 3 calibration operations are quite similar to normal event filter activities, but the results must be made available by task-to-task communication techniques. Preliminary studies of calibration needs have been completed, and a preliminary calibration system structure has been designed. Some work has been completed on understanding calibration within front end nodes and how the resultant information is communicated to the host databases.

6.4 Schedule

The Upgrade management keeps the complete MS-Project schedule for the Online Computing sub-project of the DØ Upgrade Project. A version of the schedule is available on the Online web pages (see reference in the Conclusion section).

The Online system is expected to be available for commissioning of the DØ detector elements beginning in the Spring of 1999. Many of the software components will be commissioned at the same time, as only with hardware present can the software be exercised. Most of the later (late 1999 and beyond) effort on Online software will center on the specific needs of the detector groups, and will require extensive input and programming contributions from members of the hardware groups. Such effort is not expected until the bulk of the detector construction and installation efforts are nearing completion.

The major near-term Online milestones are shown in Table 1.

Date	Milestone
2/99	Network installation
2/99	Cluster software installation
2/99	ORACLE installation
2/99	ICD commissioning: Integrated power supply control; hardware downloading; single-crate readout over primary data path; data logging; data monitoring
4/99	Muon PDT commissioning: Multi-crate readout over secondary data path
4/99	Calorimeter commissioning: Multi-crate readout over primary data path; prototype calibration; integrated run control
6/99	Alarm system commissioning

Table 1 Near-term Online Computing milestones

6.5 Manpower

Table 2 lists the number people who are currently active, or will soon be active, on DØ Online Computing projects. The table is divided into organizational unit categories.

Category	FTE available	FTE shortfall
FNAL/PPD/DØ Physicists	3.25	–
FNAL/CD/DØ Physicists	1.0	–
University Physicsists	1.5	4.75
FNAL/PPD/DØ Computing Professionals	4.0	1.0 (position open)
FNAL/PPD/DØ Computing Professionals	2.5	0.25

Table 2 Manpower Status

The obvious large shortfall of physicists (independent of affiliation) is a result of the need for people from the detector groups to begin participation in configuration, calibration, and monitoring activities specific to the detectors. The detector groups in some few cases have already identified individuals to participate in these activities, but their efforts are for the most part not yet coordinated with the Online group. The timing of the participation of detector experts is dependent upon the detector installation and commissioning plans. This shortfall figure is representative of the peak need if all efforts are simultaneous, but is indicative of the number of people and fraction of effort which are eventually needed.

6.6 Conclusion

This document gives the current status of the components of the DØ Online Computing system. Updated information on the onlien system can be found on the Online web pages:

http://d0server1.fnal.gov/www/online computing/online computing.html

7 Summary of Run II Computing and Software Status

We have the following overall plan for 1999:

- Test the simulation programs, introduce improvements, and tune parameters. Produce data sets for the two phases of the Monte Carlo Challenge.
- Complete the reconstruction program in early 99; tune it on the data set from MCC99 Phase 1; make another production release incorporating improvements from the 1999 milestones for final tuning on the dataset from MCC99 Phase 2.
- Begin Level 3 tool commissioning on trigger simulation data and data from MCC99 Phase 2.
- Begin detector commissioning with the new online system.
- Test the SAM/ENSTORE prototypes with MCC99 Phase 1; then evaluate the system and make any necessary design adjustments. Complete these systems by January 2000.
- Bring the calibration and alignment databases up for all subsystems and make calibration and alignment programs for them.

- Commission the new computing hardware for Run II.

These major efforts will require a further ramp up of personnel from the collaboration (new postdocs and students, primarily) and at least a constant commitment of Computing Division personnel at the present level of support for the Joint Projects.

We have one year to go; we have the new systems up and running to first order; now we need to JUST DO IT.

References

[1] Run II Computing and Software Plan for the DØ Experiment, http://www-d0.fnal.gov/computingff/r2csp.ps.

[2] Report of the Global Computing Model Committee, http://sbhep1.physics.sunysb.edu/ grannis/gcm_report.ps .

[3] DØOM User Guide, http://www-d0.fnal.gov/d0dist/dist/releases/test/ d0om/doc/d0om_user_guide.ps.

[4] Edm tutorial, http://d0server1.fnal.gov/www/edm/tutorial/welcome.htm.

[5] Framework User's Guide, http://www-d0.fnal.gov/d0dist/dist/releases/ test/framework/doc/user_guide.ps.

[6] RCP document?

[7] DØ Note 2537, http://www-d0.fnal.gov/software/languages/note2537.ps.

[8] R. Zybert, DSPACK User's Guide, http://hpna49_1.cern.ch/dspack.html.

[9] DØ Note 2819, http://www-d0.fnal.gov/software/languages/ recommendations.ps.

[10] http://www-d0.fnal.gov/d0dist/dist/releases/test/evpack/doc/ evpack.ps.

[11] Jon Bakken, et. al.,"Serial Media Working Group First Run II Report", March 31, 1998. D0 NOTE 3435, http://cdfaix.fnal.gov/cdf/pub/smwg/smwg_public.ps

[12] Drew Baden et. al.,"CDF/CD/D0 Data Management Needs Assessment", June 4, 1997. http://runiicomputing/runiidocs/NAG/cdrun2needs.ps

[13] Wolfgang Kowald, Lee Lueking, Kurt Ruthmansdorfer, Heidi Schellman, Vicky White "Sequential File Access Model Evaluation and Design for Run II", October, 1997. D0 Note xx. http://runiicomputing.fnal.gov/sam/doc/architecture/data_access.ps

[14] Mike Diesburg, Lee Lueking, Kurt Ruthmansdorfer, Heidi Schellman, Vicky White "Flows and Controls in the Seasons of SAM", December 4, 1997.D0 Note xx. http://runiicomputing.fnal.gov/sam/doc/requirements/five_seasons.ps

[15] Jon Bakken et. al. "Requirements for Sequential Access Model", June 1998. D0 Note 3465. http://runiicomputing/sam/doc/requirements/sam_vw_req_arch.ps

[16] Mike Diesburg et. al. "Architecture for Sequential Access Model", June 1998. D0 Note 3464. http://runiicomputing.fnal.gov/sam/doc/architecture/sam_architecture.ps

[17] Jeff Bantly, Amber Boehnlein, John Hobbs " New Phenomena Group Comments on Run II data access",July 10, 1997. http://fnpspa.fnal.gov/personal/cope/data_needs.ps

[18] Heidi Schellman and Iain Bertram "Study of possible Run II Streaming Scenarios using a Run I data sample", September 19, 1997. http://www-d0.fnal.gov/ schellma/upgrade_computing/bits_v3.ps

[19] http://www-d0.fnal.gov/dam review.html

[20] Heidi Schellman and Rich Partridge, "Summary of Luminosity Workshop", http://www-d0.fnal.gov/ schellma/lumwork.ps

[21] Heidi Schellman,"Assurance of Data Integrity in Petabyte Data Samples",December, 1998, Presented at CHEP 98, September, 1998. http://fnalpubs.fnal.gov/archive/1998/conf/Conf-98-335-E.html

[22] http://runiicomputing.fnal.gov/runIIdocs/runIIsteering/hpss/hpss.ps

[23] Don Petravick,"ENSTORE - An Alternate Data Storage System", Presented at CHEP 98, September 1998.

[24] DØ Run II Computing and Planning Board, "MSS Software for Run II - DØ Evaluation", June, 1998.

[25] http://runiicomputing.fnal.gov/sam/doc/talks and http://runiicomputing.fnal.gov/sam/doc/design/enstore-functional-components.html

[26] The SAM Home Page, http://runIIcomputing.fnal.gov/sam

[27] The RIP Home Page, http://runIIcomputing.fnal.gov/RIP

[28] http://runiicomputing.fnal.gov/sam/doc/design/RIP-interface.html

[29] The Run II Farms Home Page, http://www-hppc.fnal.gov/farms/r2farms.html

[30] The Run II Batch Working Group Home Page, http://www-hppc.fnal.gov/r2prodman/r2bwg/r2bwg.html