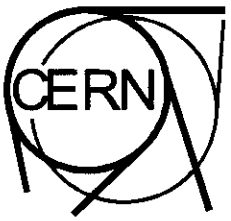

EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

CERN/LHCC 99-28
LCB Status Report/RD45
21 September 1999

STATUS REPORT OF THE RD45 PROJECT

The RD45 collaboration
CERN, *Geneva, Switzerland*

This document has been produced for the October 1999 LCB review of the RD45 project. In this report, we present the status of the project, including a summary of the responses to the milestones[1] set at the 1998 review by the LCB, suggestions for future activities and a revised risk analysis of the current RD45 strategy.

RD45 documents may be obtained through the Web via http://wwwinfo.cern.ch/asd/cernlib/rd45/index.html.

[1] See the minutes of the April 98 LCB meeting (http://www.cern.ch/Committees/LCB/minutes/minutes_apr.html) for a list of milestones.

The RD45 Collaboration

David Malon
Argonne National Laboratory, Argonne, Illinois, USA

Martin Purschke
Brookhaven National Laboratory, USA

Julian Bunn, Harvey Newman, Rick Wilkinson
Caltech, USA

Eva Arderiu Ribera, Dirk Düllmann, Bernardino Ferrero Merlino, Gunter Folger, Romuald Knap, Marcin Nowak, Andreas Pfeiffer, Jamie Shiers (spokesman), Kurt Stockinger
CERN/IT
Geneva, Switzerland

Pavel Binko, Koen Holtman, Vincenzo Innocente, Arthur Schaffer[2], Lucia Silvestris[3], Heinz Stockinger, Lassi Tuura, Ian Willers
CERN/EP
Geneva, Switzerland

Martin Gasthuber
DESY/Hamburg, Germany

David Quarrie
Lawrence Berkeley National Laboratory
Berkeley, CA, USA

Youhei Morita
KEK, Oho,
Tsukuba, Ibaraki, 305 Japan

Stansislaw Jagielski
Faculty of Physics and Nuclear Techniques,
UMM Krakow, Poland

Alexei Klimentov
MIT, USA

Christian Arnault
Laboratoire de l'Accelerateur Lineaire
Orsay, France

[2] Permanent address, LAL, Orsay, France.
[3] Permanent address, INFN, Bari, Italy.

Yemi Adesanya, Jacek Becla, Gabriele Cosmo, Andrew Hanushevsky
Stanford Linear Accelerator Center, CA, USA

Sunanda Banerjee
Tata Institute of Fundamental Research
Bombay, India

Simona Rolli, Krzysztof Sliwa
Tufts University, USA

TABLE OF CONTENTS

1 Executive Summary

Since 1995, the RD45 project has been investigating solutions to the problem of providing persistency to physics data of the LHC experiments, assumed to be in the form of (collections of) *objects*. At the end of the first year, a potential solution, based on standards-conforming products, was presented. During the second year, this possible solution was studied further – performance comparisons with existing systems and tests of functionality and scalability were carried out and production demonstrations were made.
During the past 18 months, production services have been offered, based on a combination of the two most promising commercial products identified in the first stage of the project, namely Objectivity/DB and HPSS. These tools, together with a small amount of HEP-specific code, are now distributed via LHC++[4]. In parallel, research has continued on a number of critical issues identified by the experiments. In addition, an in-depth risk analysis has been performed.

In this report, we summarise the activities of the RD45 project since the last review by the LCB in April 1998. We report on progress on the milestones set by the LCB, experience with experiments such as BaBar, COMPASS and NA45, the on-going risk analysis and make suggestions concerning possible future activities.

The RD45 collaboration has continued to focus on ODMG-compliant solutions, and has demonstrated the use of a standard, off-the-shelf ODBMS product for storing and managing HEP event data in a production environment.

Despite the focus on ODBMSs, RD45 progress in other areas is also followed, such as persistent object managers, Object-Relational Databases, including object-oriented offerings from the traditional relational (RDBMS) vendors and so forth.

As in the past, RD45 participates in the Object Database Management Group (ODMG) – the standards body that defines and maintains the various standards for ODBMSs, as well as the Object Management Group (OMG), and the IEEE Computer Society Executive Committee on Mass Storage Systems (IEEE MSS EC).

[4] See http://wwwinfo.cern.ch/asd/lhc++/index.html.

1.1 Summary of Activities Since April 1998 Review

Since the last LCB review the RD45 collaboration has:

- Together with IT/PDP group, provided production services, based on Objectivity/DB and HPSS.
- Produced a series of "white papers", covering a wide range of issues, from recommendations concerning the setup of Objectivity/DB federations to Risk Analysis.
- Made production releases of the HepODBMS class library and associated documentation.
- Held regular meetings and workshops at CERN, focussing on issues related to the RD45 milestones and object persistency mechanisms in general.
- Participated in similar workshops at SLAC.
- Collaborated closely with the MONARC project, particularly on issues related to the use of an ODBMS and Objectivity/DB in particular.
- Given a number of tutorials on database issues, including to the MONARC project, members of the ATLAS collaboration, the CERN School of Computing and at SIGMOD.
- Participated in Objectivity user meetings and ODBMS workshops in Europe and the US.
- Extended the existing license agreement for Objectivity/DB to cover the Linux operating system.
- Pursued the list of outstanding enhancement requests with Objectivity.
- Performed a revised in-depth risk analysis and a preliminary manpower estimate for the development of an alternative solution to Objectivity/DB.

1.2 Conclusions

The technologies and specific systems identified by RD45 are used as the basis for production services at a number of HEP laboratories, including CERN, DESY and SLAC, as well as at regional centres and other institutes involved in the corresponding experiments at these laboratories. The experience gained by these experiments – some of which are acquiring data at rates and in volumes that are similar to those expected at the LHC – will be extremely valuable in helping to make a decision concerning the choice of ODBMS to be deployed at the LHC. However, it now appears that previous predictions concerning the evolution of the ODBMS market were over-optimistic. Given the lack of any convincing commercial alternative to Objectivity/DB and HPSS, the risk analysis that has been performed by the RD45 collaboration suggests that the development of a fallback alternative is required. A first step in this direction is the preparation of a revised list of requirements and estimates of the manpower that would be needed to develop such a system. In the longer term, the possibility of using a suitable Object-Relational product, which now appears likely to become the dominant DBMS technology, should also be considered.

2 Milestones from the April 1998 LCB Review

The RD45 project was reviewed by the LCB in April 1998, and recommended for continuation for a further 18 months, with the following milestones and comments:

"The project has achieved the initial R&D goal of investigating and identifying potential solutions to the problem of persistent data storage for LHC experiments. The proposed solution: ODBMS (Objectivity) is now adopted for data persistency not only by all the LHC experiments but by many others (BaBar, NA45, COMPASS, RHIC) ready to take data in 1-2 years.

RD45 has met the 1997 milestones set by the LCB and is congratulated for its excellent work. The project has addressed many of the initial questions about the use of object databases for data persistence during its four-year lifetime and shown that commercial solutions can be applied. The project is now moving into a new phase combining the creation of a production data management service while at the same time continuing R&D to enable the technologies to be used effectively in LHC collaborations. In this new phase, in which R&D will be carried out in parallel with production activities which are equally important in validating the overall strategy, the LHC experiments are expected to actively participate in defining and performing the necessary R&D. The LCB recommends that the project be continued with the next status report scheduled in 18 months time in order to accommodate both production and R&D activities. The LCB internal referees should make a brief status report to the LCB at 6 month intervals."

LCB endorsed the following proposed milestones:

1. Provide, together with the IT/PDP group, production data management services based on Objectivity/DB and HPSS with sufficient capacity to solve the requirements of ATLAS and CMS test beam and simulation needs, COMPASS and NA45 tests for their '99 data taking runs.
2. Develop and provide appropriate database administration tools, (meta-)data browsers and data import/export facilities, as required for (1).
3. Develop and provide production versions of the HepODBMS class libraries, including reference and end-user guides.
4. Continue R&D, based on input and use cases from the LHC collaborations to produce results in time for the next versions of the collaborations' Computing Technical Proposals (end 1999).

LCB asked that a list of very specific R&D activities, concerning point 4), be identified between the project and experiments, and discussed in the next meeting.

Work on these milestones and recommendations is covered in detail below.

3 Interim Referees' Reports

3.1 December 1998

The following is reproduced from the minutes of the LCB meeting held in December 1998. The full minutes are available via http://www.cern.ch/Committees/LCB/public/minutespub/minutes_dec98public.html.

The LCB referees reported on the RD45 project status based on the recent RD45 workshop and the extensive published white papers and other documents. The LCB notes that excellent progress is being made on both the production and R&D milestones. Important lessons are being learned from the production service, identifying issues and potential problems that will now be solved well in advance of the LHC. The LCB appreciates the hard work by the groups responsible for running this production service.

The LCB in particular looks forward to publication of results from the ATLAS 1 TeraByte tests which will begin to answer important performance questions on production size databases. In general the experiments appear to be getting more involved with the RD45 project, although the project still awaits updated user requirements and use cases from the experiments. The cooperation between RD45 and the new MONARC project should help in this regard, as well as enhance the investigation of wide area database performance.

The LCB discussed the RD45 risk analysis and agrees that it would not make sense to do a full port to a second OODBMS vendor's product at this time.

The LCB agrees with the other suggested steps to mitigate risk, with the addition of trying to insure that user code in reconstruction and analysis programs is kept as standards compliant as possible.

3.2 June 1999

The second interim referees' report was made at the June 1999 LCB meeting. The minutes of that meeting were not available at the time of writing, but will eventually be accessible via http://www.cern.ch/Committees/LCB/public/meetingspub.html.

4 Milestone 1

The first milestone set at the April 1998 review of the RD45 project was as follows:

"Provide, together with the IT/PDP group, production data management services based on Objectivity/DB and HPSS with sufficient capacity to solve the requirements of ATLAS and CMS test beam and simulation needs, COMPASS and NA45 tests for their '99 data taking runs."

4.1 Introduction

The model that has been adopted to support production Objectivity federations has been as follows:

- Each experiment or project is assigned a dedicated system on which the lockserver process for the associated federation is run.
- All such systems, or "lockservers", including a hot stand-by, are identical Sun Ultra Vs, with 64MB memory, 9GB external mirrored disk.
- The federation catalogue, boot-file and any journal files also reside on these systems.
- The catalogue and any important meta-data databases are regularly backed up.
- Databases are maintained on one or more "dataservers".
- Only the database administrators of a given experiment are able to login to the corresponding lock and data servers.

4.2 Production Setup

The following text is reproduced from "Objectivity/DB Services - Installation and System Configuration[5]" and describes the setup of production Objectivity/DB services at CERN.

4.2.1 Introduction

An Objectivity service comprises a Lock Server and one or more Objectivity database servers. The Database servers may exceptionally be part of the Lock Server but in general will be hosted on separate system(s).

Objectivity/DB groups databases within **Federations** and each Federation has a unique Federation Identification number and is described by a Federation file. The **Databases** themselves map onto files in UNIX systems and are subdivided into **Containers.** The containers are further divided into **Pages**. Access to the database is controlled by locks allocated by the Lock Server process and this locking is performed at the Container level. Data access however is performed at the Page level.

[5] http://wwwinfo.cern.ch/pdp/te/objy/objyservice.html.

4.2.2 Lock and Database File Servers

The Objectivity Lock Server is the central point of a federation and a single lock server can support many federations. In particular, the Lock Server holds the **federation file** which are a catalogue of the federation database structures and schema information. Lock Server systems will run both the Lock Server Daemon (ools) and the AMS Server even where the Lock Server hosts no database files. File servers for Objectivity Databases will in general be separate systems from the Lock Server and will provide a secure filebase.

The proposed directory structure on UNIX systems is as follows with the convention that the federation and boot files have uppercase filenames.

/usr/objy/FDNAME/FDNAME.FDDB
/usr/objy/FDNAME/FDNAME.BOOT

Objectivity/DB journal files are written into the directory /usr/spool/<federation-name>/journal

During pilot tests of Objectivity/DB, databases may also be housed on the Lock Server in the **/usr/objy** namespace as shown below. The backup directory is used for periodic disk-to-disk backups of the Federation and Boot files.

/usr/objy/db/FDNAME/databasefile(s) . . .
/usr/objy/db/backup/FDNAME/ backupfile(s) . . .

4.2.3 Installing Objectivity/DB Software

The complete suite of Objectivity/DB administrative commands will be installed on all Lock Server and Database servers. New services will run Objectivity/DB release 5.0 although Objectivity/DB release 4.0.2 may be installed on some UNIX systems for architectural or other reasons. In the case of Windows NT, Objectivity/DB version 4.0.10 is available but this release is not part of the supported releases for UNIX systems. The main distribution directories are located in AFS as described below.

4.2.4 Objectivity/DB 4.0.2

The distribution directory[6] is:

/afs/cern.ch/sw/lhcxx/export/sun/objectivity/4.0.2

Upon installation, the Objectivity/DB commands and libraries are copied to the target system and are located within the /usr/local directory. The total volume of installed files is about 60MB. The installation is handled by the interactive C-Shell script:

[6] The current production machines are all Sun systems. However, the software also exists for DEC, HP, IBM, Linux, SGI and NT. See http://wwwinfo.cern.ch/asd/lhc++/Objectivity/index.html for further details.

/afs/cern.ch/sw/lhcxx/export/sun/Objectivity/4.0.2/INSTALL.CSH

Within the installed directory, **/usr/local/Objectivity/4.0.2,** there are several subdirectories:

lib
bin
doc
include
demo
etc

4.2.5 Objectivity/DB 5.1

The most recent production release of Objectivity/DB is Release 5.1[7]. The 5.1 release cannot coexist with 5.0 on the same system.
The distribution directory is:

/afs/cern.ch/sw/lhcxx/export/sun/Objectivity/5.1

Upon installation, the Objectivity/DB commands and libraries are copied to the target system and are located within the /usr/local directory. The total volume of installed files is about 110MB. The installation is handled by the interactive C-Shell script:

/afs/cern.ch/sw/lhcxx/export/sun/Objectivity/5.1/INSTALL.CSH

Within the installed directory, **/usr/local/Objectivity/5.1,** there are several subdirectories:

lib
bin
doc
java
include
Toolkit
demo
etc

4.2.6 Starting the Lock Server and AMS Server

Two command scripts are used to start the server daemons: **oolockserver** and **oostartams.** The servers are started under a generic usename/groupcode which for example, in the case of CMS would be **objsrvzh/zh.** The Ultra5 lock servers are administered as part of the SERCO contract and automatic scripts have been written which periodically check for the

[7] Release V5.1.2 of Objectivity/DB is used in the September 1999 release of LHC++. The V5.2 release is expected in October – November 1999.

presence of the Lock Server daemon and restart it if necessary. This approach ensures that the Lock Server daemon is started with the correct user and group codes. User restarts of the Lock Server on the central lock servers are discouraged.

4.2.7 Coexistance of Objectivity/DB Releases 4.0.2 and 5.1

Objectivity/DB release 4.0.2 and 5.1 may exceptionally be installed on the same UNIX system. Switching between the two versions will be made by setting appropriate values for the PATH (Command search path) and LD_LIBRARY_PATH (Shared Library Path) variables.

In order to set the variables mentioned above, a script is available in afs:

/afs/cern.ch/rd45/objectivity/objyenv.{sh|csh}

This script should be invoked after first setting the global environment variables OS, OBJY_VERS and OBJY_DIR as shown in the examples below:

To select Objectivity/DB 5.1 using the Bourne Shell:

```
OBJY_VERS=5.1; export OBJY_VERS
OBJY_DIR=/usr/local/Objectivity/$OBJY_VERS; export OBJY_DIR
. /afs/cern.ch/rd45/objectivity/objyenv.sh
```

To select Objectivity/DB 5.1 using the C Shell:

```
setenv OBJY_VERS 5.1
setenv OBJY_DIR  /usr/local/Objectivity/$OBJY_VERS
source /afs/cern.ch/rd45/objectivity/objyenv.csh
```

4.2.8 Objectivity Backup Issues

The backup scheme used for Objectivity/DB federations is described in section 5.2 below.

4.2.9 Lock Server Naming Conventions

Six SUN Ultra5 systems have been purchased for use as Lock Servers (see Table 1).

User	Group	Experiment	IP Alias	IP Name	IP Address
objsrvzp	zp	atlas	lockatl	oolock04	137.138.236.117
objsrvzh	zh	cms	lockcms	oolock01	137.138.236.115
objsrvzh	zh	cristal	lockecal	oolock02	137.138.236.114
objsrvvy	vy	compass	lockcomp	oolock05	137.138.236.118
objsrvyt	yt	na45	lockna45	oolock03	137.138.236.116
objsrv	N/A	N/A	spare	oolock06	137.138.236.119

Table 1 - Lockserver Hosts

4.2.10 Objectivity Databases residing in AFS

In order to support databases created in afs file space, the lock server must be started with the **noauto** option:

```
ools -OO_NO_AUTOREC
```

4.3 ATLAS

4.3.1 Database Servers

Two development systems are currently in place for ATLAS:

- **ATLOBJ01** – a Sun Ultra 5 with 90GB of disk space, used for testing.
- **ATLOBJ02** – a dual CPU Sun E450 with 72GB of RAID 5 disks and 27GB of internal scratch disk , used for program development. Some tens of users are registered on this machine.

Both the above systems run Objectivity/DB V4 and V5[8], including the HPSS-aware AMS.

4.3.2 1TB Milestone

An internal milestone of the ATLAS collaboration was to store 1TB of realistic physics data into Objectivity/DB + HPSS by the end of 1998. The main purpose of this milestone was to demonstrate the feasibility of the different elements of the ATLAS software using a first approximation of the raw data model of an ATLAS event. A secondary goal was to understand the performance of the various components of the system, although actually obtaining the maximum throughput was not the main focus of the test.

The configuration that was used is shown in Figure 1. Raw events (digitisations from jet production using GEANT-3 based software stored in ZEBRA FZ format) were staged to disk and then converted into Objectivity/DB format using 5 concurrent clients on IBM AIX systems (RSPLUS nodes). Two systems running the Objectivity/DB AMS (data server) were used – both Sun Solaris machines. The data rates achieved were as follows:

- ~1.5MB/s (databases migrated from disk to HPSS).
- ~3MB/s (disk-only).

It is understood that disk contention was the primary cause for the lower data rate in the former case.

[8] Each major release of Objectivity/DB uses a different set of port numbers (increased by one with each version).

As the tests were performed over the Christmas period, the operational efficiency was lower than would normally be achieved – around 50%. As a result, some 19 days were required to reach the 1TB milestone.

It is intended that the test be repeated – perhaps on a smaller scale – using a modified event model, based on the use of a segmented VArray with an optimised iterator.

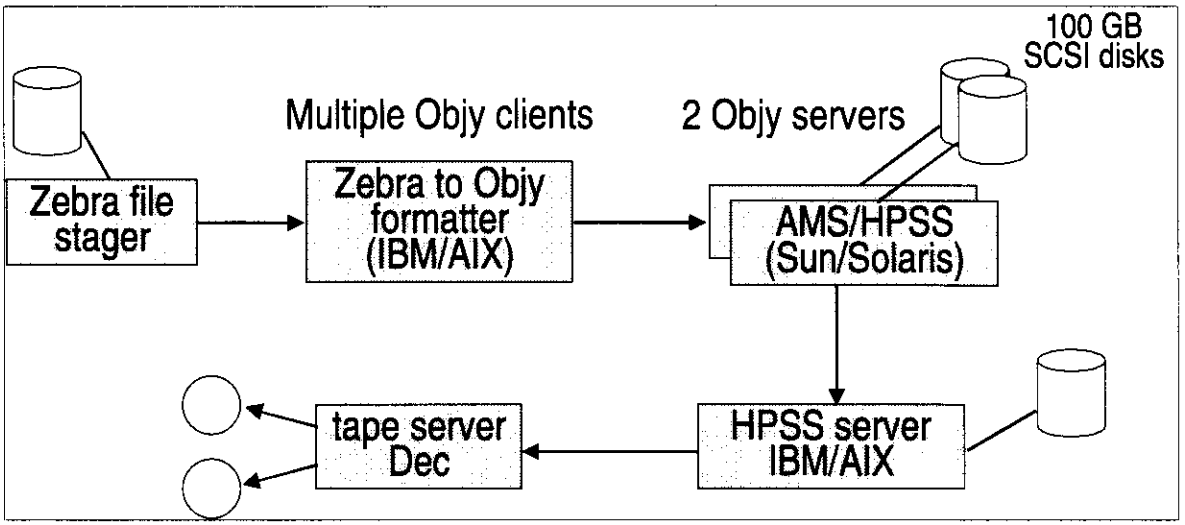

Figure 1 - ATLAS 1TB Milestone Logical Configuration

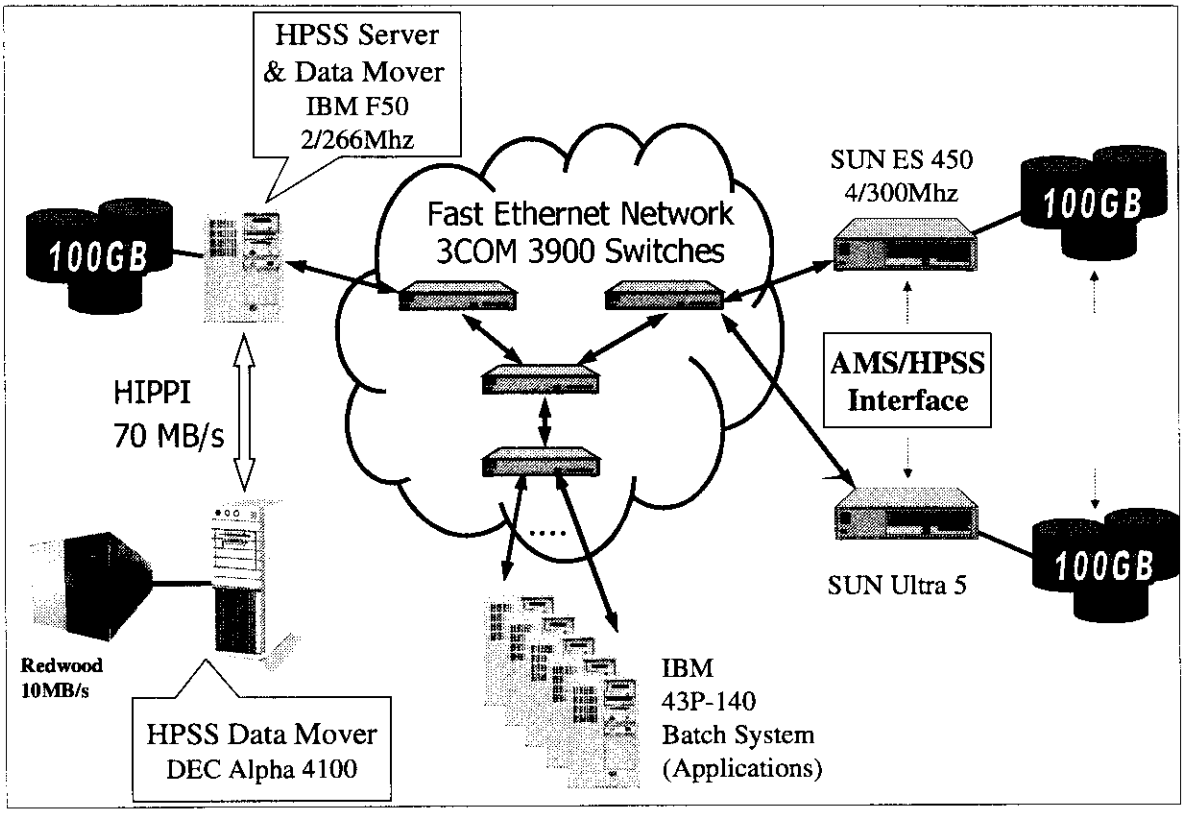

Figure 2 - ATLAS 1TB Milestone Physical Configuration

4.4 CMS

4.4.1 Test-beam Activities

CMS have used Objectivity/DB for numerous test-beam activities (H2b, X5b, T9). The Data acquisition framework is common (Figure 3). Only few specifics "storage classes" need to be specialized to reflect differences in the readout electronics.

For each test beam, a separate federated database is populated. These federated databases are composed of:

- System databases (meta-data), e.g.:
  - Beam configuration.
  - Log book.
  - ListRuns.
- Run Databases – each containing a collection of events taken under the same conditions.

Two federations are used per test-beam: one online and the other offline. Data are written locally on the online system and then the databases (files) copied to the offline data-server system using *rfcp* and attached to the offline federation (see Figure 4). All Run Databases are also stored under HPSS.

Prompt data monitoring is performed directly online polling the online federation while it is populated (a "commit-and-hold" is issued each few thousand events).
Offline analysis is performed directly on the data-server or remotely, accessing the data through the AMS.

Although data are also stored in HPSS no automatic tape staging is used yet, due to the unavailability of a reliable version of the "linkable AMS". CMS plans to start tests of the Objectivity-HPSS interface, on a non-production machine, this fall.

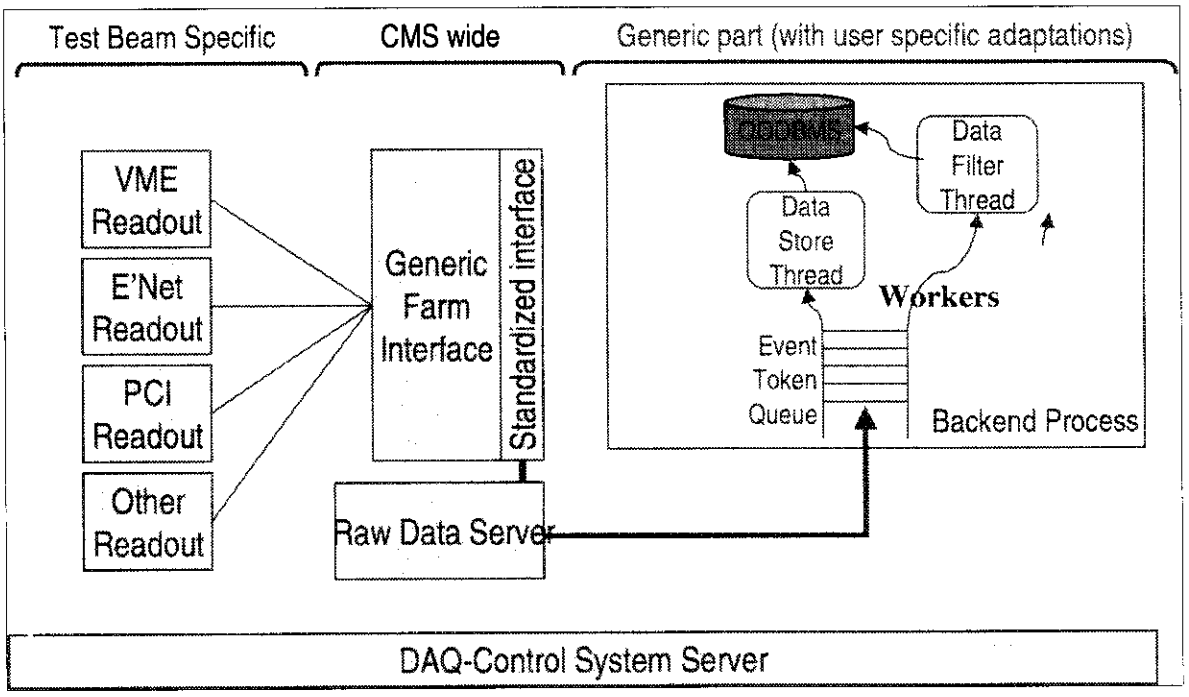

Figure 3 - CMS Test Beam Setup

The data volume handled in each test beam is summarized in Table 2. It is worth noticing that during last X5b test-beam up to 25 concurrent users were accessing data (mostly interactively) on the offline system without any observable degradation of its response.

A new test-beam for tracker detectors will be performed in October in X5b. About 150 GB of data are expected to be collected.

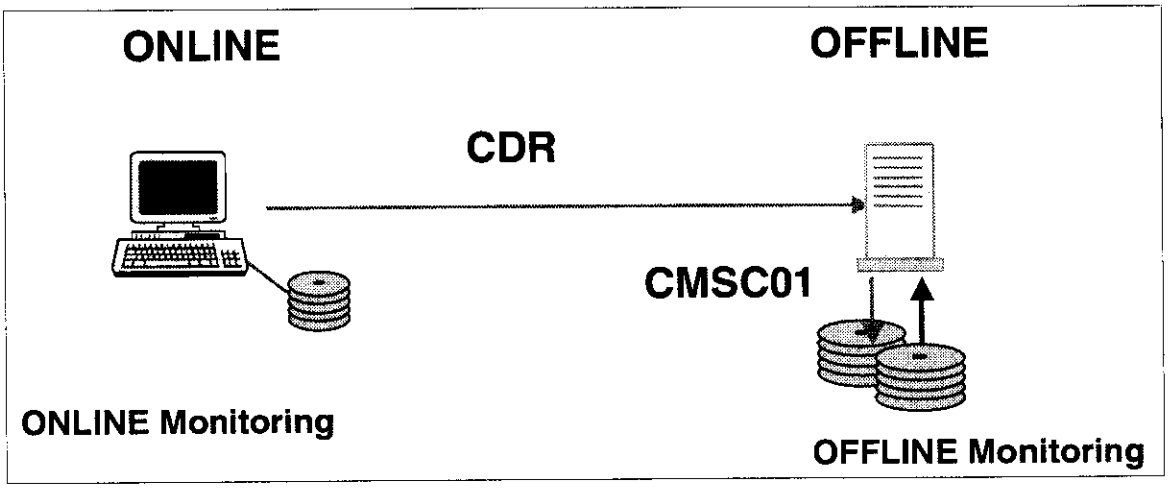

Figure 4 – CMS Online and Offline Federations

The data acquisition rate is currently limited to 1MB/s by the bandwidth of the network used by the central data recording system to transfer files from the test beam area to the offline system. An upgrade of this network to 100MB Ethernet is foreseen for next year.

Test Beam	Period (Real DAQ)	Number of Database files	Total Size (GB)	Average file size (MB)	Average data transfer speed (KB/s)	Number of users
T9 (Tracker)	3/5-10/5	350	150	500	800	10
H2b (Muon and Silicon beam telescope)	16/7-21/7	100	6	30	800	3
X5b	14-8-25/8	560	500	900	900	25

Table 2 - Summary of CMS Test Beam Data Volumes (1999)

4.4.2 Test Beam I/O Simulation

To study possible performance bottlenecks of the test-beam setup we run a simulation of the data recording system focusing on the Objectivity/DB part.

The simulation was run on a SunEnterprise 450 (4 X UltraSPARC-II 296MHz) with a memory size of 512 Megabytes. The system was equipped with a local 80GB striped file-system of 19 slices 128K "away".

Such a system is capable of a raw I/O rate in excess of 90 MB/s.

We created a persistent raw data structure similar to those typically written in current test beams with objects of random size (in average 2.4KB of useful data each).

Each process produced 4000 events with 45 raw-objects each (180K raw-objects in total) corresponding to a total of 436MB of useful data. Including event structure and overhead a total of 469MB were written to disk per "run".

Objectivity I/O was optimized (as in real Test-Beam DAQ) using a federation with 32KB page size and a data model with no associations (only VArrayT<ooRef()>). Critical run time parameters were "INDEX mode" set to 0 and the maximum cache size set at 100 pages. A commit and hold was performed each 1000 events.
The test was run changing the number of processes running in parallel (from 1 to 16) and populating the same federation.
Results are summarised in Figure 5 where the total elapsed real time (and the corresponding I/O rate) is plotted against the number of concurrent processes running.
The figure includes the time spent to create databases and containers, which requires the processes to acquire a federation lock and therefore to queue.
These results prove that we can write a realistic raw-data event structure using Objectivity/DB at 5MB/s per processor. Parallel I/O does not introduce major performance

penalties. Indeed in a later test, running on two similar systems concurrently (single federation, remote "system databases", local run-databases), we were able to reach a sustained aggregate speed of 40MB/s confirming that a remote federation catalog and remote application "system databases" are not a performance bottleneck.

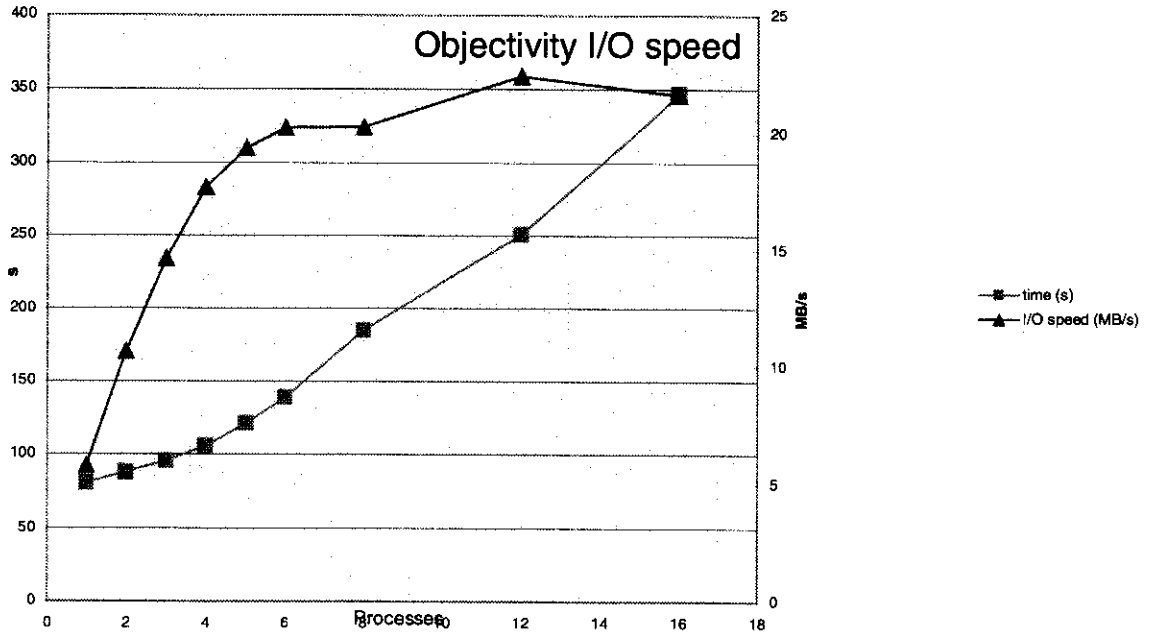

Figure 5 – Real Time and I/O Rate vs Number of Concurrent Processes

4.4.3 100MB/s Milestone

An internal CMS milestone, to be achieved by June 1999, was to populate an ODBMS with raw data at 100MB/s. This milestone was successfully accomplished ahead of time, and at significantly higher data rates. Using CMS multi-jet QCD events, data rates of up to 172MB/s were obtained using the Caltech 256 node SMP Exemplar system (Figure 6).

The simulated raw events consisted of 6 objects and had an average event size of around 260KB. All Objectivity/DB client processes were running on the Exemplar system, whereas the lockserver and federation catalogue were maintained on separate HP workstations.

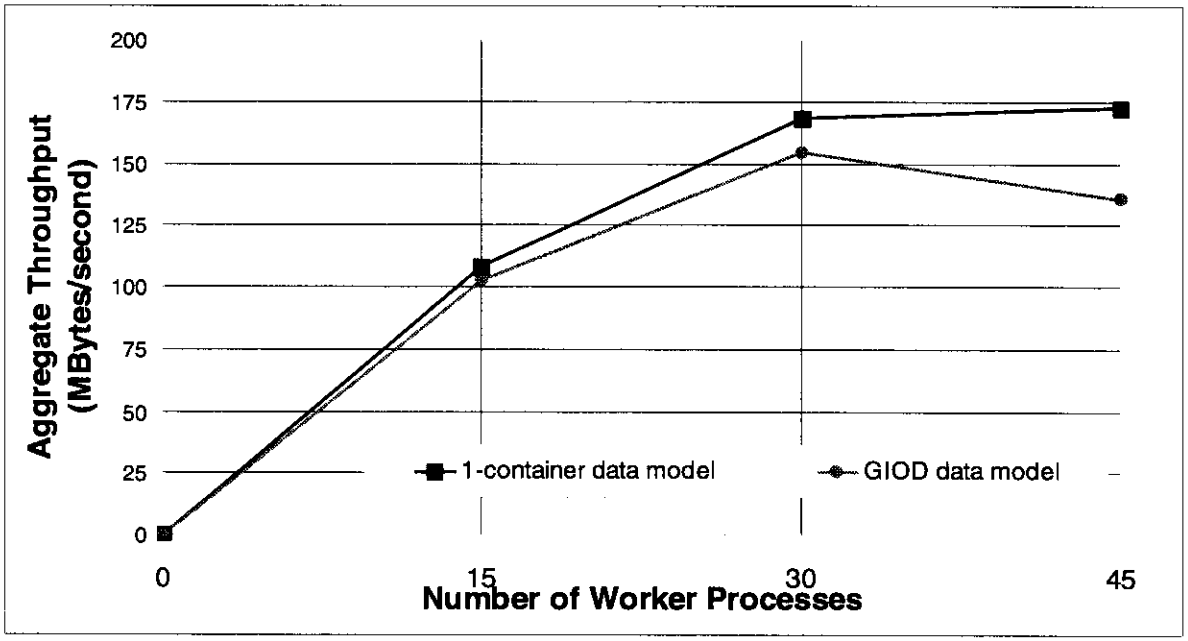

Figure 6 - Aggregate Throughput vs Number of Worker Processes

4.4.4 Cristal

The CMS Cristal project uses an Objectivity/DB federated database to store information related to product and process tracking for CMS detectors. The databases are populated from Windows/NT clients. As their disk space requirements in 1999 are rather modest, two trays of 18GB disks are connected directly to the lock server in a RAID 5 configuration, including a hot-spare disk. The total usable disk space is 200GB of which some 1.5GB were occupied at the time of writing. However, more than 1TB of data are expected over a four year construction period, including sites at CERN, in China, France, Italy, Russia and the UK.

The main goals of the Cristal project are to provide:

- A distributed repository for the storage of CMS detector data.
- To track the status of defined assembly sequences.
- To implement tools to monitor the production cycle and to provide redirection facilities.
- To provide quality control throughout the CMS lifecycle.
- To provide controlled access to the repository for physics applications (e.g. calibrations, simulations etc.).

The current work of the project is focussed on:

- Deployment of the Cristal software in more regional centres and subdetectors.
- Implementation of a query facility that uses meta-data for query optimisation and provides a GUI.
- Implementation of a domain independent meta-data manager.

- Database replication of data over WANs using XML.
- Making production data available to physics applications (pre-calibration, calibration, simulation).

4.5 CHORUS

The CHORUS experiment is looking for neutrino oscillations in the CERN muon neutrino beam. For active identification of potential tau neutrinos, 1800 kg of nuclear emulsions have been exposed. The CERN Chorus group is building automatic scanning stations, using commercial, industry-standard components in terms of hard- and software wherever possible.

Extrapolations into the emulsion sheets of tracks reconstructed by the electronic detectors are obtained from DSTs and stored in an object oriented database (Objectivity), which is implemented on a server running AIX. The four scanning stations, which are equipped with one PC running Windows NT each, run Objectivity client software to obtain the predictions. On each scanning station, a digital Megapixel CCD camera is connected via interface boards both on the microscope and the PC side, with a DSP and a frame grabber in the PC. The PC also communicates via serial interface lines with the light source controller and the motor controller. The motors move the desired section of the emulsion sheet below the microscope objective.

In automatic mode, the frame grabber serves mainly for the display of the captured images from the camera and for the monitoring of the local DAQ, whereas the DSP is in charge of controlling the camera and capturing the digital images which are processed on the DSP to recognize grains. The Pentium CPU of the PC is used to run the software to recognise tracks in the emulsion. The system is designed to eventually make optimal use of the possibilities of pipelined processing offered by the hardware. Data of a given image are then sent back to the database server, where they are stored. Because of the high network bandwidth required, FastEthernet connections are being used.

Control messages about the status of each scanning station are sent to a central dispatcher running on the AIX station, from where they can be requested by virtually any machine on the LAN. The operator interacts with the scanning stations via a dispatcher client, on top of which a Tcl/Tk panel has been implemented. Manual scanning, which is performed after automatic scanning to cross-check interesting events, also uses a Tcl/Tk based interface. Another dispatcher client uses Mathematica for further quasi-online analysis and monitoring of the data.

Most currently used software components, which have been written in the framework of the project, use C++ as implementation language, with some parts written in Tcl/Tk. While the DSP code is being re-written in Assembler for performance reasons, components which involve network communication and/or user interfaces are being migrated to Java.

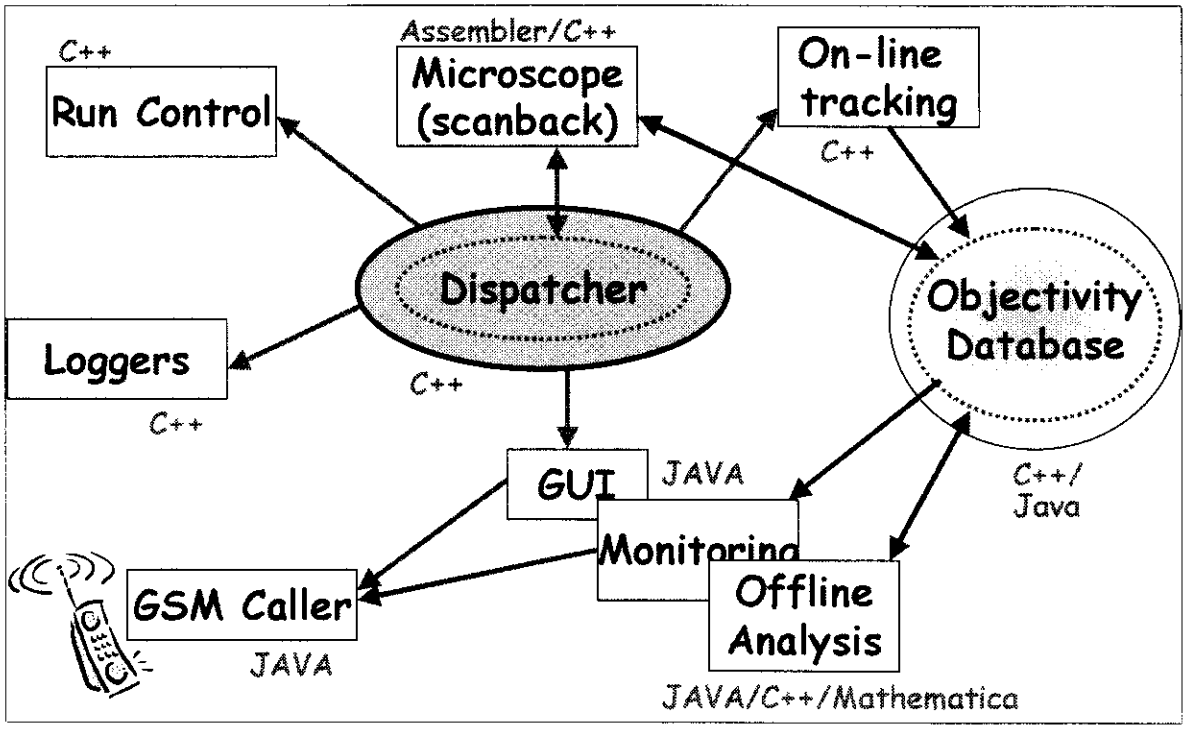

Figure 7 - CHORUS Scanning System

4.6 NA45

NA45 – the first HEP experiment to use Objectivity/DB in production – plans to store data at 15MB/s in a data taking run starting in November 1999.

Data is collected by 4 PCs running Linux. 2 Sun 450s are used to convert the data into Objectivity/DB format, and Gigabit Ethernet connections are used to transfer the data to the CERN Computer Centre. An aggregate bandwidth of 20MB/s has been achieved, using 4 streams per sending PC, writing databases locally to the Sun systems (no AMS).

4.7 COMPASS

The COMPASS experiment expects to acquire over 300TB of data per year at data rates up to 35MB/s. Both these volumes and rates are similar to those of ATLAS and CMS, but need to be handled some 5 years earlier.

In the COMPASS system, data are sent from the online system as a byte-stream. They are received in the Computer Centre and stored in Objectivity/DB, as shown in Figure 8. The setup is designed to avoid single points of failure, including mirrored disks for critical information and a hot spare for the lockserver. Tests of robustness have been made by inducing a crash with many open transactions and replacing the lockserver with the spare.

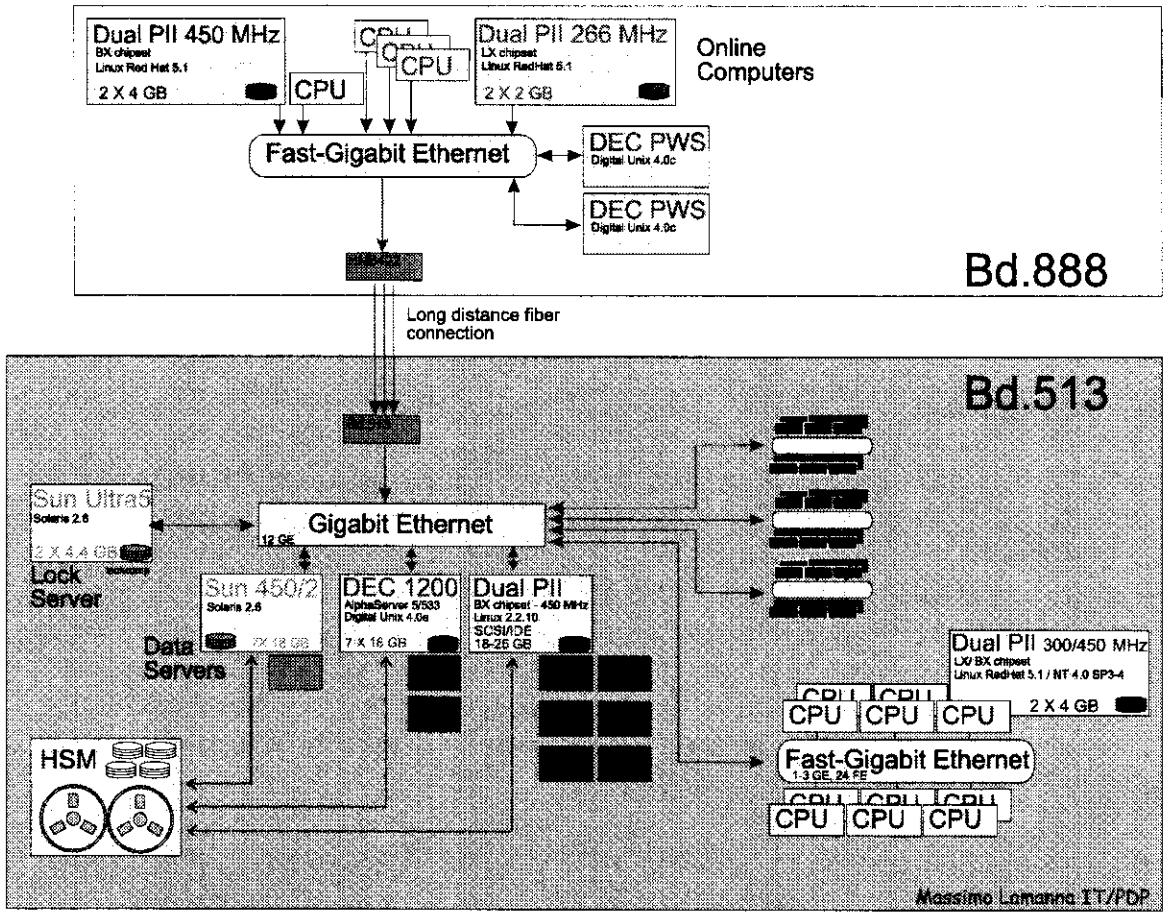

Figure 8 - COMPASS Central Data Recording

4.8 Conclusions

The production infrastructure required for ATLAS, CMS, NA45 and COMPASS has been provided. A number of important internal milestones have been met by these experiments, including the ATLAS 1TB and CMS 100MB/s tests. The use of central data recording has been demonstrated by the NA45 and COMPASS collaborations at data rates of 15 and 35MB/s respectively. Production use of Objectivity/DB has also been made by other experiments, including the CHORUS collaboration. Other laboratories – in particular DESY and SLAC – have also gained valuable production experience with Objectivity/DB and the regular RD45 workshops continue to be an important forum for information exchange, identifying new requirements and following up on outstanding enhancement requests.

5 Milestone 2

The second milestone set at the April 1998 review of the RD45 project was as follows:

"Develop and provide appropriate database administration tools, (meta-)data browsers and data import/export facilities, as required for [milestone] *(1)."*

5.1 Introduction

In order to support the production services required for milestone 1, adminstration tools are clearly required. Although Objectivity/DB comes with a set of tools for managing Objectivity/DB federated databases, these tools are somewhat basic and typically need to be integrated into the local environment. For example, the *oobackup* tool allows federated databases to be backed up and restored, but will typically be embedded in a site-specific script, much the same as any Unix backup program. Similarly, whilst Objectivity provides a generic browser, there is a clear need for an extended browser that understands the object model in question, as well as local conventions, such as those used by BaBar for database naming.

Under the LHC++ Web page, an "Objectivity/DB at CERN" page has been setup[9]. This page contains information about how the software is installed at CERN, how a local installation may be performed, where to find and how to run various test programs and benchmarks, pointers to documentation – both local and Objectivity-provided – and a description of the various tools that have been developed. These tools are described in more detail below.

5.2 Federation Backup for Production Environments

5.2.1 Federation Backup Goals

To ensure a reliable use of Objectivity/DB in production a strategy for backup and restore of the production federation has to be established. The backup procedure should cater for recovery from data loss caused by either hardware faults (e.g. disk failures) or the more likely case of data corruption by faulty application software.

Since object databases like Objectivity/DB allow data located in different files to be associated (e.g. using cross-file object IDs), the recovery of a consistent transactional state of the whole federation is usually more complicated than storing and retrieving single unrelated files from tape. The backup procedure needs to be integrated with the database

[9] See http://wwwinfo.cern.ch/asd/lhc++/Objectivity/index.html. Similar pages are also available for other packages, e.g. http://wwwinfo.cern.ch/asd/lhc++/ObjectSpace/index.html for the ObjectSpace class libraries.

lockserver and should only copy data that is in a consistent state (not currently updated by an ongoing transaction). Also one has to make sure that all copied files belong to the same transactional state of the federation.

For federations containing large data volumes one would typically like to apply different backup strategies to different parts of the data model. In many cases one can identify databases that contain data that is relatively easy to regenerate so that data loss could in fact be accepted. On the other hand are some parts of the federation – like the file catalogue and the schema – that are of central importance for the ODBMS (or the particular application) so that data loss in this area can not be tolerated. A flexible backup facility would therefore allow restricting a backup run to e.g. the subset of the database files, which contain important data in order to minimise the I/O overhead, media costs and the time required performing the backup.

5.2.2 Backup Tools provided by Objectivity

To address the problem federation backups Objectivity/DB provides several tools (e.g. *oobackup* and *oorestore*) which allow either complete or incremental backups on a deployed federation. A complete backup stores the content of all database files to a backup medium (e.g. tape), the incremental mode allows storing only the contents of those containers that have been modified since the last backup.

To guarantee the consistency throughout the federation, the Objectivity tools access every database in the federation and check it for updates. For federation containing large volumes of read-only data stored in a mass storage system this approach is clearly not very practical since it would yield to a complete transfer of all data in the MSS to disk.
An extension of the current backup tools has therefore been requested from Objectivity, which would allow the user to specify explicitly the set of databases that should be considered for the current backup. For other databases (typically containing read-only data) the user (or the mass storage system) would take the responsibility of keeping valid data copies.

5.2.3 Implementation for the Objectivity Production Service

Since these extensions to the Objectivity backup tools have not yet been implemented, the production service is currently using a set of scripts that implements a partial federation backup based on database file copies. To guarantee consistency the backup script acquires an update lock for all involved databases and keeps it until all files have been successfully copied. The set of databases that are copied defaults to the federation file and the boot file. This list is supposed to be extended by the user to include any other central application databases like collection registries or calibration databases.

5.2.4 Example: Backup for the NA45 production federation

The NA45/CERES production system uses a central lockserver machine, which also stores the central database meta-data like the boot file and the federation file. In the case this

machine should experience a hardware or software fault it is replaced by a second identical machine that always contains a valid disk copy of all central federation files. The backup procedure in fact utilizes the fallback machine to store temporarily a complete copy of the production federation until it has been copied to tape.

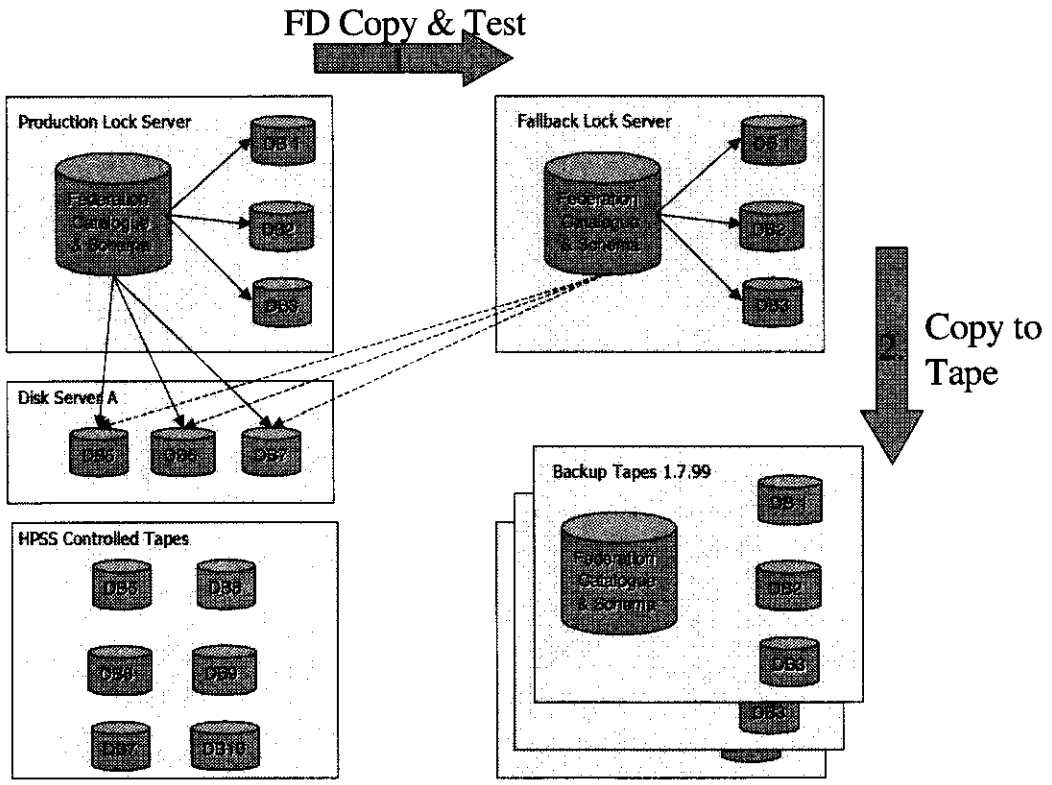

Figure 9 - Objectivity/DB Backup Procedure

The backup procedure involving these two machines is organized as follows:

- **Determine if the production federation is in a consistent state.**
  This check may be implemented either by acquiring appropriate database lock or by querying the lockserver process controlling the production federation. If the database is not in a consistent state the backup has to be deferred. Since there is currently no reliable way to quiescent the federation there is no guaranty that the backup will ever succeed.
- **Copy all relevant database files to the fallback machine.**
  The file copy between both machines is currently implemented by cross mounting the disk of the fallback machine using NFS onto the production machine.
- **Install the Backup Federation on the fallback machine**
  The installation is done using the *ooinstallfd* tool with the *–nocheck* option. This option allows the catalogue of the backup federation to be updated to reflect the files that are copied but leaves all catalogue entries pointing to the data server untouched.

- **Consistency Check of the Backup Federation**
  Pending the availability of an better tool for consistency checking, the *oodump* command is used to access every object in the copied databases at least once.
- **Copy backup federation to tape**

5.3 Database Browsers and Adminstration Tools

5.3.1 The CERN DRO_Tool

DRO_TOOL (Data Replication Option Tool) is a tool for managing the configuration of a federated database. With this tool, the database administrator is able to observe, control and manage the fault tolerance, and detect failures in the system as soon as possible.

Objectivity/DB provides tools and programming interfaces to help perform administration tasks. The tasks and tools are similar on all platforms, but without a friendly interface (most of them are command line programs).

DRO_TOOL is a visual tool that offers the same capabilities as the command line tools included with the Objectivity/DB, together with some additional functionality that is provided in the Java binding. The tool has been implemented in Java to avoid multi-platform problems and has been tested on Windows NT and Solaris.

The tool has been developed using the JDK 1.2 beta 2, and uses the JFC Swing package included on it (Swing 1.0). It also uses the Voyager package from ObjectSpace and has recently been ported to Objectivity/DB V5.1.

The tool is available via http://wwwinfo.cern.ch/asd/rd45/tools/dro_tool.html.

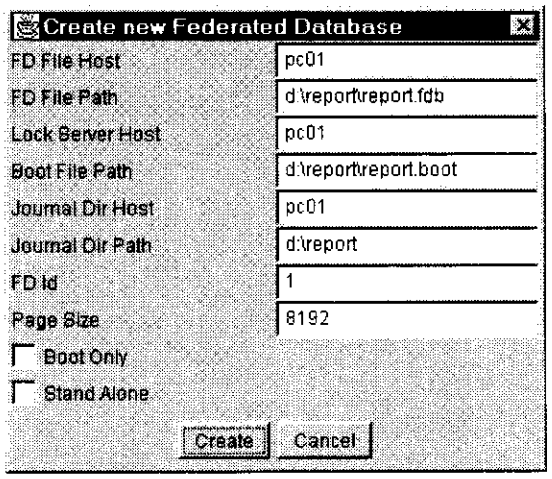

Figure 10 - Creating a Federation with DRO_TOOL

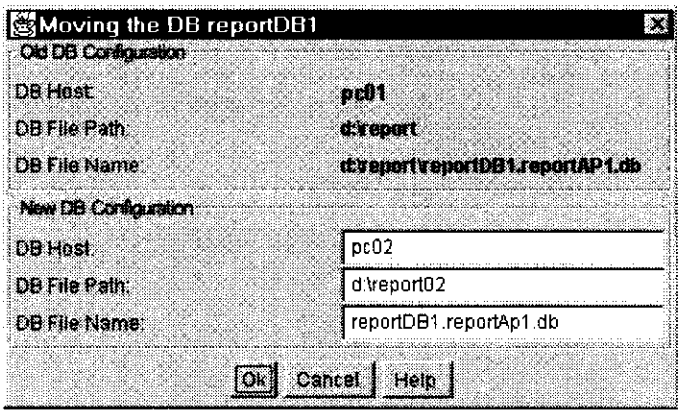

Figure 11 - Moving a Database with DRO_TOOL

5.3.2 Hudson

Hudson is a database browser developer by Micram Object Technology – the distributors of Objectivity/DB in Germany. Hudson offers a number of features that are similar to DRO_TOOL, including the ability to create new autonomous partitions and databases, as shown in Figure 13.

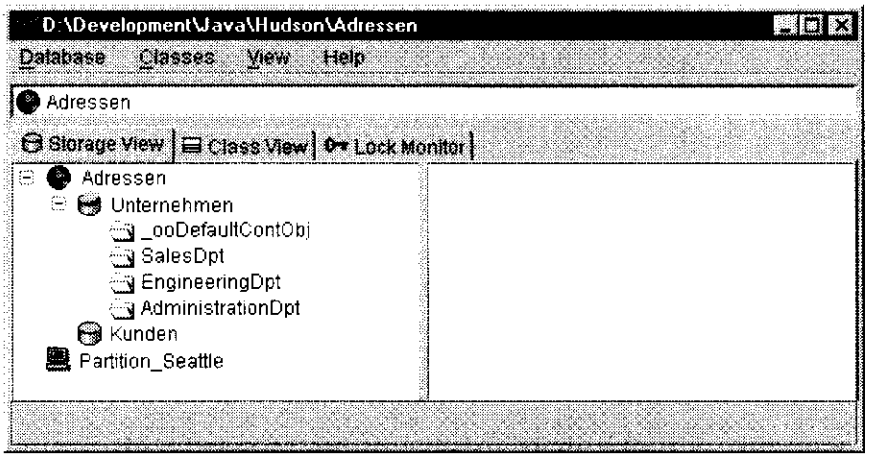

Figure 12 - Hudson Main Window

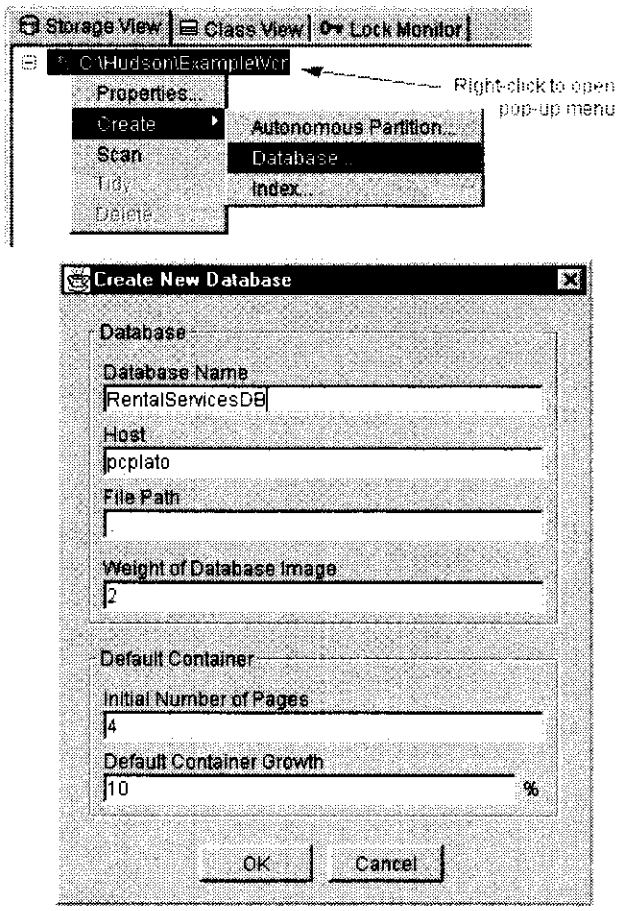

Figure 13 - Creating a New Database

Hudson also contains a lock monitor, that displays entries for all locks being currently held by applications working against the federated database opened by the browser (see Figure 14).

Each entry contains information about:

- whether the lock is held or released,
- the resource the lock is held on (federated database, database or container),
- the age of the lock,
- the process ID of the process holding the lock,
- the transaction ID of the transaction holding the lock,
- the host ID of the host which the process holding the lock is running on,
- the user ID of the user who owns the process holding the lock.

The content of the lock monitor is refreshed with a period specified in the settings dialog.

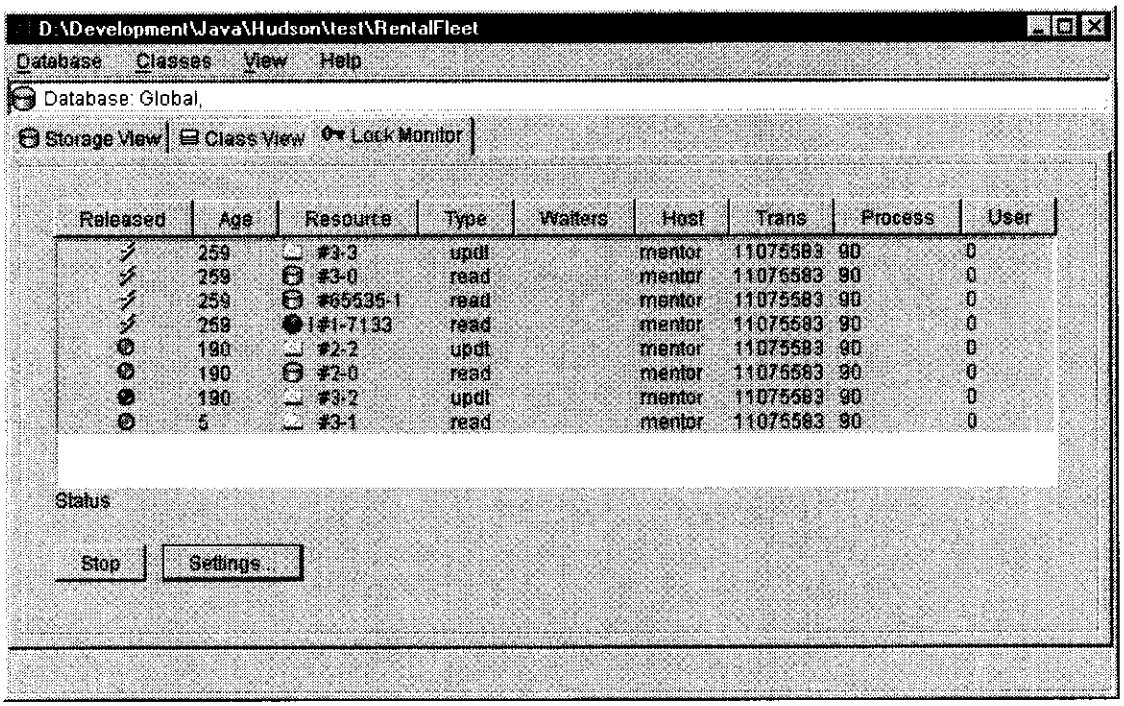

Figure 14 - Hudson Lock Monitor

5.3.3 The SLAC Database Browser

The following information is taken from the BaBar database browser Webpage[10].

The BaBar database browser provides both logical and physical views of a BaBar federated database. The client is implemented in Java and uses CORBA to communicate with a C++-based server that accesses the database using the Objectivity/DB C++ interface.

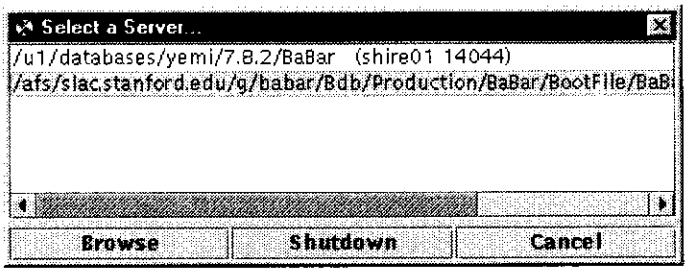

Figure 15 – Starting the BaBar Browser

The main GUI panel (see Figure 16) contains two tree widgets: The Database Tree and The Event Collection Tree (see Figure 17). They allow you to browse through database files and event collections respectively. Place the mouse pointer above a database icon for a brief moment: The database ID should be displayed in the form of a "tooltip".

[10] See http://www.slac.stanford.edu/BFROOT/www/Computing/Offline/Databases/Docs/BrowserGuide.html.

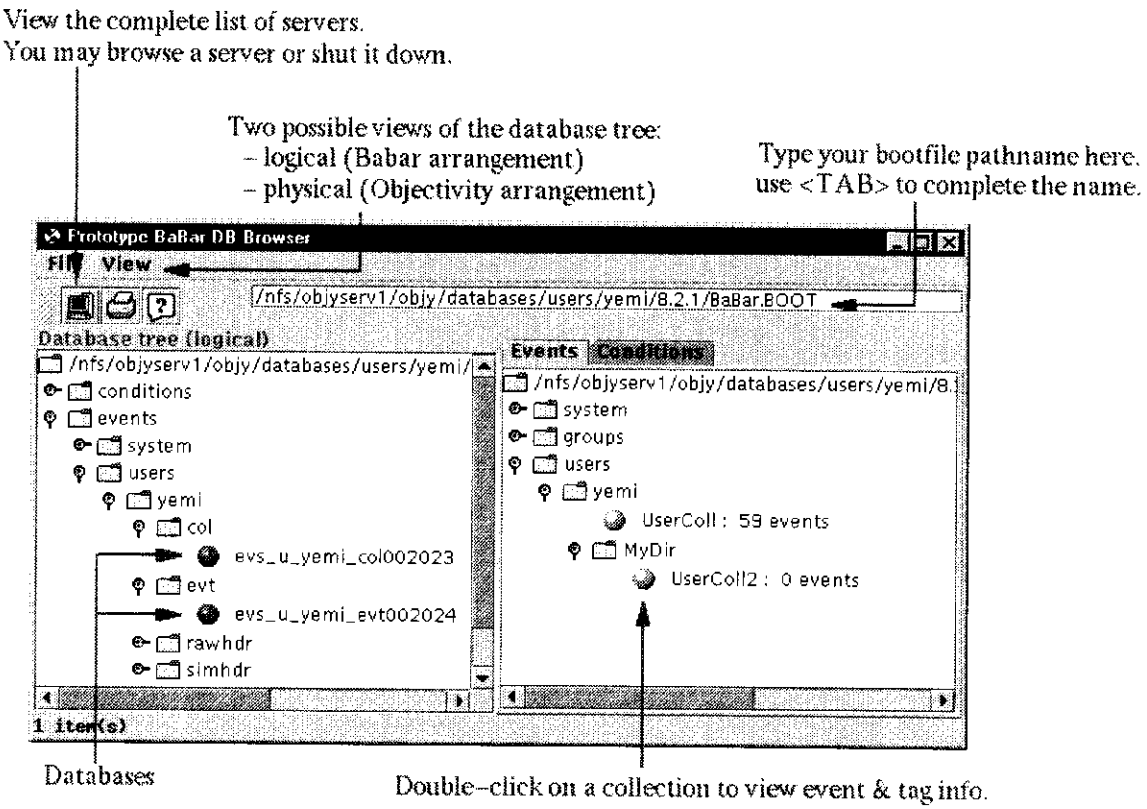

Figure 16 – Main Window of BaBar Browser

The Database tree presents either a logical or physical abstraction of the database hierarchy. Use the "View" menu to toggle between the two modes. The logical view is a BaBar-oriented structure of domains and authorization levels. The physical view shows the placement of databases according to file hosts and filesystem paths.

Both the Database tree and Event Collection tree have pop-up menus which may be accessed by clicking the right hand mouse button. You may hide the database/collection icons if you're browsing a large federation. Also, you may choose to display the number of events by default (rather than having to double click on every collection). Use this option with care since a read lock is required in order to retrieve the number of events.

The ability to browse event collections is a high priority in the list of requirements. Double-clicking on a collection will now allow you to view the list of BdbEventIDs and any subcollections. Double-clicking on an event entry will open up a new window displaying event information. The list of event stages will be shown. If the Event contains a tag, the tag data will be listed. Please use with caution especially if other users are writing to a federation.

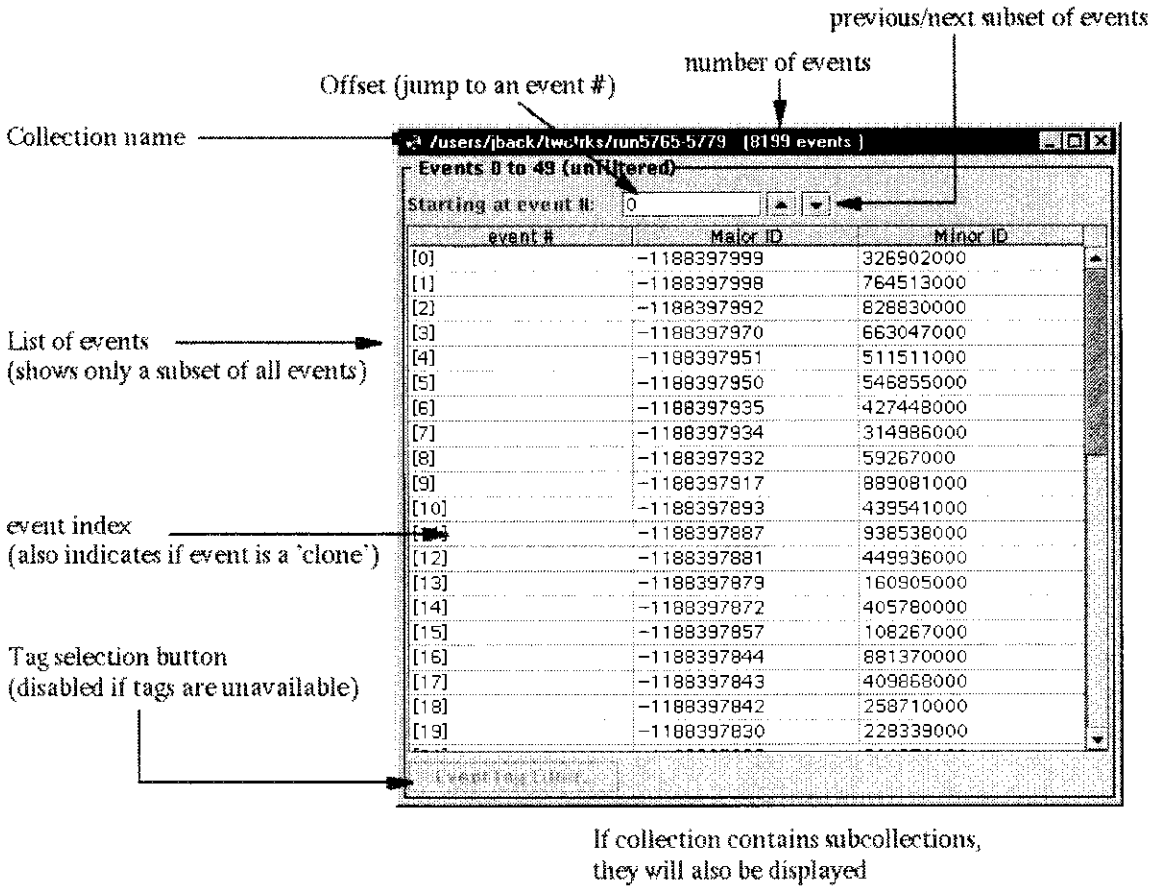

Figure 17 - Event Selector Window

In the case of collections that contain descriptions of tag attributes, you can make simple tag queries. The Event browser (shown above) now displays an "Event Filter" button. If the button is active, the collection contains tag descriptions. Double-clicking on an Event entry will display its tag attributes. These attributes may be used as a means of selection. Use the "Event Filter" button to open the Filter dialog window. Select an attribute and a value. Apply your selection and now the Event browser will only display events that match the selection.

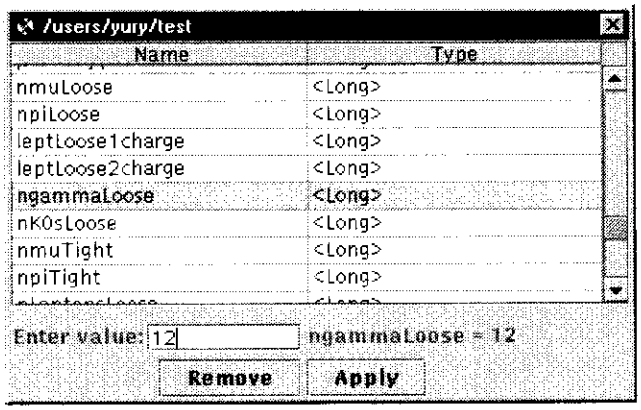

Figure 18 - TagFilter Window

5.3.4 Conclusions

A number of database browsers are available, in either prototype or production form, covering a range of functionality from database administration functions to object-level browsing. There are useful features in all of these tools and it would appear to be an area where collaboration and code-sharing would be of benefit. The ideas discussed at the RD45 workshop held in July 1999 concerning an extensible browser framework clearly require further thought.

5.4 Data Import/Export

Although the movement of databases between different federations is carried out by numerous experiments, by far the largest experience in wide-area data import and export has been gained by the BaBar collaboration. We therefore describe the data distribution techniques developed by BaBar[11].

BaBar currently stream their data into two streams, which are stored in separate (sets of) databases and even filesystems:

1. Bulk (~85%)
2. IsPhysics (~15%)

It is expected that the number of streams will increase in the future, probably to four. All data distribution is performed using the "IsPhysics" stream, with data exported to centres in France, Italy and the UK, together with a full copy at IN2P3 in Lyon.

5.4.1 Event Store

Two sorts of data export from the event store are supported:

- Event collections
  - Specify collection(s), component(s) (RAW, REC, SIM, AOD, ESD, ...)
- Incremental
  - Specify "from date", component(s), stream(s)

In both cases, the tool will extract the databases in question and attach them to the target federation.

5.4.2 Conditions

In the case of conditions data, incremental export is supported as follows:

[11] See http://www.slac.stanford.edu/BFROOT/www/Computing/Offline/DataDist/.

- Specify "from date", domain name (conditions, configuration, temporal, spatial, ...)
- Detector name

The databases are then extracted and attached as above.

5.4.3 Copying / Shadowing

Database maybe exported between federations using a number of techniques, including *copying* and *shadowing* (see section 7.4).

- **Copying** – one or more database files are physically copied, e.g. using *cp*, and the copy is attached to the target federation.
- **Shadowing** – one or more database files are attached to more than one federation simultaneously. In this case, the file must be in readonly mode.

5.4.4 Summary

The above scheme for data import and export has been used in production at BaBar since the first day of data taking (import/export between federations at SLAC). Many improvements are needed, included crash-recovery and minimisation of the interference with data taking and production.

5.5 Conclusions

The development of the appropriate tools for managing an Objectivity/DB federation – such as administration scripts, database browsers and import/export facilities – is an on-going activity. Given the different environments that need to be supported, it is unclear whether generic tools can be provided and further work is required to understand if (sufficiently) extensible browsers and administration scripts are feasible. However, the provision of the appropriate tool set is clearly an important element of a production service and work in this area will continue.