CDF/DOC/COMP_UPG/PUBLIC/4503

Implementation plan for a data handling solution based on an Object Oriented Database System

W. David Dagenhart, Kristo Karr,
Simona Rolli, Krzysztof Sliwa

Tufts University

Abstract

In this note we describe an implementation plan for a data handling solution based on an Object Oriented Database System. The stages are: 1) to develop a software Loader/Reader module, to insure backward compatibility with TRYBOS/FORTRAN based modules; they already exist; 2) development of a data model (Run II data model) using input from the physics groups; 3) building a prototype of a database according to the Run II data model and optimize the clustering; 4) HSM integration: this has been already provided using HPSS and tested on the preliminary level at SLAC, Caltech and CERN. We could use another HSM, if available.

Objectivity/DB provides the tools to address completely issues like: setup and configuration of a new database system, managing of objects (names, OID, recipe, ...), storing and retrieving the same. C^{++} API and GUI tools are provided as well. We could use LHC++ or ROOT as data browsers and analysis tools. The effort could focus on development of the Run II data model and optimization of clustering of the data, which we believe is the most challenging problem, given the anticipated limited amount of disk space in Run II.

1 Introduction

We present an implementation plan for the data handling approach based on an Object Oriented Database System and its applications to CDF RUN II. Objectivity/DB, an object oriented database, has been proposed as a solution to data management problem and carefully evaluated by the RD45 Collaboration at CERN, as well as many HEP experiments (COMPASS at CERN, STAR and Phenix at RICH, BaBar at SLAC ...) which choose to use it as the candidate for their data handling system.

The model is based on the concept of a distributed object database. In Objectivity/DB the individual databases (files) can be spread across multiple servers on the network to form a federated database which appears to a user as a logically single database. This solution offers a clear advantage in scalability because data can be distributed on multiple servers (these servers may be different machines, made by different vendors with different operating systems). Objects which will be accessed most often can be clustered on fast access medium (disk), while those which are requested less often can be physically located on slower media. Technically the federated database can reach the volume of tens of petabytes with large files (order of 100 GB) now . In Version 6.0 of Objectivity/DB (due end of 1998) the mapping between containers and files will be implemented allowing the petabytes size of the federated database to be built out of small files.

The reader is strongly suggested to read the documentation complementary to this note.

2 Interface requirements and implementation stages

Objectivity/DB provides tools to address completely issues like: setup and configure a new database system, managing objects (names, OID, recipe, ...), as well store and retrieve objects. C^{++} API and GUI tools are provided as well.

To take full advantage of the ODBMS users should, at the analysis level obviously, use one of the OO languages, e.g. C++ or Java. On the other hand we know that C++ and FORTRAN modules will have to co-exist in the CDF Run II reconstruction code. In the solution we propose reconstruction code does not interact directly with persistent objects, but only with transient ones (TRYBOS banks). clustering and managing of persistent objects is done in an intermediate level between transient and persistent layers through Loader and Reader Modules.

Two possible scenarios are proposed: 1) a Loader module immediately before reconstruction so that this reads and writes objy objects through input/output modules. 2) the Loader module immediately after reconstruction: this reads YBOS files and write OBJY objects through an output module. Only part of RAW data are going to be put into the federated database.

2.1 Stages

Two Loader modules exist already corresponding to different data schemata, one as a A_C++ module, the other as a standalone module running using YBOS: the conversion to TRYBOS and A_C++ is straightforward. Two Reader modules exist already.

This corresponds to our first implementation stage, or software stage, where we provide Loader and Reader modules. Backward compatibility with FORTRAN modules is guaranteed: through the transient/persistent mechanism reconstruction software does not see directly Objectivity objects, but interacts with TRYBOS banks, (the TRYBOS event record becomes more flexible allowing allocating banks conforming to the user preference).

A second stage consists in the development of the CDF data model for Run II: it will proceed through

- discussions within the physics groups on how to split the data into disk-resident and tape-resident information;
- optimization of the schema.

Part of this stage is already underway (see the input from the Gedanken experiments).

A third stage will consist in building a prototype database according to the Run II data model. Data from Run I can be imported in the new database and studies on optimization of clustering can be performed. The first version of an Objectivity/DB organized according to the Run II schema (based on a zeroth data model) can be delivered by May 1998.

Finally a fourth stage, hardware stage, is the one when we will start using a hierarchical storage manager interfaced to the database. In this way we will be able to create federated databases on the HSM using our HSM-enabled AMS. This has been done until now using HPSS, at SLAC, Caltech and CERN. This stage can start immediately. Two additional FTE computer professionals are needed, as well as close cooperation with the FNAL CD groups working on HSM's.

2.2 Platform compatibility

Objectivity/DB runs on every of the CDF supported platforms. As far as LinuX is concerned port actually has been done, it needs testing. This will be done by Dirk Duellmann from CERN, who will spend 6 months at the Objectivity headquarters. A similar port to Sinux (a Siemens version of LinuX) has been extensively and succesfully tested in the past. The GNU compiler has been used for the moment.

3 References

"A solution for data handling based on an Object Oriented Database System and its Applications to CDF RUN II"
CDF/DOC/COMP_UPG/PUBLIC/4346

"Gedanken experiments for a data handling solution based on an Object Oriented Database System and its Applications to CDF RUN II"
CDF/DOC/COMP_UPG/PUBLIC/4492

" Flows and controls for a data handling solution based on an Object Oriented Database System "
CDF/DOC/COMP_UPG/PUBLIC/4493

"RD45 - A persistent Object Manager for HEP"
CERN/LHCC 96-15

"Object databases and mass storage systems: the prognosis"
CERN/LHCC 96-17

" ATLAS Computing Technical Proposal"
CERN/LHCC 96-43

"Object databases and their impact on storage-related aspects of HEP computing"
CERN/LHCC 97-7

"Object database feutures and HEP data management"
CERN/LHCC 97-8

"Using an object database and mass storage system for physics analysis"
CERN/LHCC 97-9

" Status Report of the RD45 project"
CERN/LHCC 97-6