ELSEVIER

Nuclear Physics B (Proc. Suppl.) 78 (1999) 727–731

NUCLEAR PHYSICS B
PROCEEDINGS
SUPPLEMENTS

www.elsevier.nl/locate/npe

An Oracle©Database for the AMS experiment

M.Boschini[a,b] ,M.Gervasi[a*] , D. Grandi[a†] , P.G.Rancoita[a] , L. Trombetta[b] and I.G.Usoskin[a‡]

[a]INFN sezione di Milano, via G.Celoria 16, I-20133 Milano, Italy

[b]C.I.L.E.A., via R. Sanzio 4, Milano, Italy

We present hardware and software technologies implemented for the AMS Milano Data Center. Goal of the AMS Milano Data Center is to provide data collected during the STS-91 Space Shuttle flight to users and to provide a User Interface as well to manage the data properly. Data are stored in a database that provides high level query and retrival features, the support being a magneto-optical juke-box. We describe the use of proprietary software (Oracle©) as well as custom-written software to enhance access performances. In particular we underscore the use of the Oracle Call Interfaces as a powerfull tool to interface the database and the operating system in a natural way.

1. Introduction

One major issue in nowadays high energy physics experiments, is data manipulation. Increasing sample sizes and data complexity require high level tools to handle, store and analyze the data. These tools have to be robust, flexible and provide an easy to use interface in order to allow front-end users to access and actually use the data, with little knowledge of the technical aspects of the tools themselves.

Furthermore, easy access to the data is strongly reccomended, if not required, by large international collaborations. Collaborators should be able to access data from wherever they may be and with minor requirements on their hardware and software.

One natural solution to this issue is offered by databases. Databases, used for a variety of different applications, ranging from stock exchange to hospital management, satisfy the above mentioned requirements of robustness, reliability and flexibility. Recently, databases have found a new field of application in particle physics where they have proven to be the best tool to manage large and complex data samples ([1],[2]).

In INFN Milano laboratory, the data sample has been organized in database to provide an easy access to the data sample and the database for all AMS collaborators.

2. The AMS experiment

The Alpha Magnetic Spectrometer (AMS), described in greater detail elsewhere [3], is a space borne particle physics experiment designed to search for and measure, with a much greater sensitivity than heretofore possible, cosmic rays particles and anti-particles. This will improve our knowledge on unusual types of matter (dark matter, antimatter) in the Universe as well as on cosmology.

The first data-taking took place in June 1998 on board of the STS-91 Shuttle flight. For the second mission AMS will be carried on the Shuttle up to the International Space Station (ISS) where it will take data for about 3 years.

The AMS detector has five major elements: a permanent magnet, a silicon microstrip tracker, a scintillator time of flight hodoscope, scintillator anticoincidence counters and an aerogel Cerenkov threshold counter. In addition there are electronics and support structure and interfaces.

During the STS-91 flight, AMS collected approximately 110 Gbytes worth of data and about 10 Gbytes of calibration, geometrical and positional data.

The actual data were sent to Earth during the

*and Dept.of Physics - University of Milano

†and Dept.of Physics - University of Milano

‡on leave from A.F.Ioffe Physical-Technical Institute, 194021 St.Petersburg, Russia

flight by means of two different bandwidths: a High Rate Data Link (HRDL) with a band-width of about 2 Mbit/s, which was used for the real data, and a Slow Rate Data Link (SRDL) with band-width of 100 Kbit/s which was used for calibration, maintenance (HK) and positional data (CAS).

3. The Milano Data Center

As already stated, the main goal of the Milano AMS Data Center, as approved by the INFN funding committee, is to provide all collaborators with an easy to use tool to access, retrieve and select data.

We decided to implement two different and complementary systems to access the data: on one side we keep the whole data sample *as it is*, accessible for direct download (see further explanation). On the other side, we organized the same data in a database, in order to allow the user to select and retrieve only those data satisfying any requirement or condition the user may impose on them.

This may prove to be necessary on one side because of the amount of the sample, which makes any full-sample analysis unfeasible; on the other side because of the complexity of the data and the relations between different parameters (e.g. track bending versus Shuttle position). This actually confirms the usefullnes of databases in modern High Energy Physics (HEP) experiments.

Furthermore, being AMS an international collaboration, we decided to build a *Network User's Infterface* based on the `http` protocol, so that all the user needs to access the data is a browser.

When the Milano Data Center was still under project, the total data sample was evaluated to be of about 70 GBytes, thus we estimated our disk space needs to be around 180 Gbytes (70 Gbytes for the data sample, 100 for the database implementation). At that time the media with the best Price/Gbyte ratio was an HP Magneto Optical JukeBox. Since the Magneto Optical JukeBox is best administered with a proprietary package from HP (*OmniStorage*©), we adopted HP platforms.

Tab. I is a summary of the hardware and software used in the implementation of the Milano Data Center. The Database resides on one server (DB-Server) to which the Magneto Optical JukeBox is connected. The *as it is* data sample is accessible from another server (FTP-Server), connected to a 72 GBytes disk-array.

Component	**Hardware**
Workstations	HP C160
X-Terms(2)	HP 5211A
DataBase Server	HP D210 (44 GB)
Magneto Optical JBox	HP 200 FX (197.6 GB)
Data Distribution Server	HP D220 (8 GB)
Disk Array	HP (72 GB)
O.S.	HP-UX 10.20
RDBMS	Oracle©RDBMS v.8.0.3
JBox Admin	HP OmniStorage A.02.02
Develop. Env	OCIs, SQL+, Perl5.004

Tab.I Hardware and Software used in the Milano Data Center.

Finally, we decided to use a relational database: despite the fact that the data sample is currently being stored in an object database (from now on indicated as ODBMS) at Johnson's Space Center and at CERN, we preferred the semplicity and the reliability of a relational database to the complexity of an ODBMS. Furthermore, recent studies ([2]) seem to indicate that if no object oriented implementation is needed for the project, an ODBMS is not necessary and can be avoided.

We preferred to use a commercial relational database (referred to as RDBMS throughout the rest of the paper) in order to have full support from a vendor, and being Oracle©the most widespread commercial RDBMS, we installed Oracle©RDBMS Server v.8.0.3.

Due to some transmission problems during the flight, we recieved only about 50% of the whole data sample. The entire data set arrived at the

Milano Data Center later in July on DLT tapes. Any implementation here described refers to the partial sub-sample, and throughout the rest of the paper we'll refer to this sub-sample if not differently specified.

4. Database implementation

Due to the data sample size and the data structure itself, we had to exploit some of the most advanced features of the Oracle©Server.

Generally speaking, implementing a database for an HEP experiment, implies at least two operations: a *translation* from the original data format to one understood by the RDBMS, and a *loading* of the data thus manipulated, which heavily involves the operating system of the machine on which the database is implemented.

The more the *translation* reflects the nature of the data themselves, the more the RDBMS's logical representation of the data is similar to the data. This, at least in principle, is the optimal implementation of a RDBMS [4], because any possible normalization or re-normalization is already taken into account for.

Furthermore, a similar implementation reduces any possible space-overhead due to redundant or incorrect normalization of the logical rapresentation of the data [5].

Finally, the less manipulation the *translation* requires, the faster any application dedicated to load the data in the database will run.

We have let us guide in definig the logical structure of the data by the data format itself. AMS data, both HRDL and SRDL, can be thought of as variable length binary strings, the *events*, part being a fixed length *header* and part being the actual data (signals from detectors, machinery status, etc.). The *header* is in turn composed by 10 fixed length words [1] containing information about time, detectors and sub-detectors.

Since these are the variables that will be used to query the database, any event can be stored in a table having 10 alpha-numeric fields and one binary field. Query fields need to be at least alphanumeric in order to be queried upon. In this way, the phisiological space-overhead due to a binary to alpha-numeric conversion is kept at the minimum.

In this way, we defined a biunivocal relation between the input data and its logical representation in the RDBMS, satisfying the above mentioned goal of a strict relation between input data and RDBMS implementation.

With respect to the binary field, since its length may vary from a few bytes up to nearly 1kByte, we had to use a particular datatype: the so-called *Long Raw*, which can be efficiently used only with v.8.0.3 and higher of the Oracle©RDBMS server.

In particular, using this datatype, we were able to optimize the loading procedures in terms of time needed to load an event in the database.

In general, if one wants to load some data in a RDBMS, one can rely upon some conventional tools based on SQL, which in the case of Oracle©can be SQL+ or Pl-SQL. However these tools are too far from the operating system, and thus impose a time overhead on the loading procedures.

Furthermore, not all datatypes are accessible to languages like SQL+ or PL/SQL, which is limiting the power of the RDBMS itself; besides, this means also a *non-natural* approach to the implementation of the database, in the sense that one succeeds in representing correctly the data as defined above, but is not able to store them in the database according to this representation.

To solve both these issues, operating system "vicinity" and datatypes accessibility, we exploited the so-called C - Oracle©Call Interfaces (OCIs). This is an application programming interface (API) that allows an application developer to use a third-generation programming language, like C, and its native procedures or function calls to access the Oracle©database server and control all phases of SQL statement execution. Oracle Call Interfaces provide a library of standard database access and retrieval functions in the form of a dynamic runtime library, that can be linked in by the application. This eliminates the need to embed SQL or PL/SQL within the user's code. Oracle Call Interfaces supports the datatypes, calling conventions, syntax, and semantics of the chosen high level language and the

[1]Not all words have the same length: some are 3 bits long, others 4 or 6 bits long.

particular version of the RDBMS that is used.[6]

Using the OCIs we were able to load the data using code which is very near to the operating system, allows full exploitation of the RDBMS features and is optimized for this particular application.

In particular, the code runs directly on the data sample *as it is*, unpacks the event into header and data, unpacks the header into sub-headers and loads everything in the database.

A similar and specular procedure retrieves data from the database: the program queries the database for events satisfying a particular request, reads all the fields for these events, re-packs header and data info and writes out a file which has exactly the same format as any input file.

This feature was one of the main goals of the Milano Data Center: we want to make any physicist in the collaboration able to run the same analysis program both on data from the data taking and on data extracted from the database. This means maximum flexibility and transparency for the user.

5. User's Interface

One of the goals of the Milano Data Center was to make the data available via network by just using a browser (Netscape or InternetExplorer), without requiring the user to have any knowledge of Oracle©neither of the paricular database implementation.

Therefore we provide an HTML form that enables to query the database and select only those events satisfying a particular requirement. The user is totally free in the selection of the variables of interest. Some control mechanisms of the auto-consistency of the query are implemented and guide the user in the filling of the form.

The form is associated to a CGI-BIN written in Perl5.004 that, once the query parameters are obtained from the user, translates them in an input argument to the OCI procedure and submits the query to the database.

Once the data are obtained and "re-packed" in a file, as previously described, the file is automatically copied (by means of rcp or scp [2][7]) to the work station specified by the user in the form and, finally, the user is notified by e-mail about completion of the operation.

[2] *Secure Shell's* copy.

Similar interfaces and mechanisms have been developed to access the *as-it-is* data sample. The user selects up to 20 MBytes worth of data files, organized by data-taking period, and the CGI-BIN rcp's or scp's them to the remote workstation.

Altough explaining the procedures which authorize and controll any access to the data goes beyond the scope of this paper, we can briefly define an authorized user as someone from a collaborating institution who has been given a set of passwords to access the HTML form and the database. Conventional password checking techniques, RSA crypting and SSI controls have been implemented.

Currently, both HRDL and SRDL data from the first 6 days of data taking are available in the database, while the whole data sample is available for direct access, for a total of approximately 50 GB.

In the first 3 months of operativity, Milano Data Center has had nearly 200 accesses from 15 authorized users, whith a total data transfer of approximately 20 GB.

6. Conclusions

We described the main golas of the Milano Data Center and how we succeded in providing more than 50 GBytes of data to the user who can access them with only a web browser.

In particular, we underlined the fact that using a "natural" approach to describe the data in the database and to implement loading procedures as near as possible to the operating system, optimizes disk space occupancy and loading times. Futhermore, as the extension of the Milano Data Center to the complete STS-91 data set will demonstrate, a natural approach thus defined, ensures scalability and maximum flexibility.

REFERENCES

1. D. Torretta et al. "Online-offline calibration databases evaluation", Fermilab paper PN-

519, 30 August 1995.
2. RD45 Collaboration, LCB Status Report/RD45, CERN/LHCC 98-11.
3. AMS's home page. http://ams.cern.ch/AMS/ams_homepage.html
4. J. Martin, "Computer Data-Base Organization, 2nd Ed." Prentice-Hall (1980).
5. G. Koch, K. Loney, "ORACLE: the Complete Reference", 3rd Ed., Oracle Press, 1995
6. "Developing Applications with Oracle Call Interfaces Release 8.0" Oracle Technical White Paper, June 1997
7. "SSH- Secure Shell", http://www.ssh.fi/sshprotocols2/index.html