Integration of a Relational Database in the CERN PS Control System

J.H. Cuperus, M. Lelaizant,
CERN, 1211 Geneve 23, Switzerland

Abstract

The control system for the CERN 26 Gev Proton Synchrotron and its injectors is a generic system which can be adapted to other accelerators. Most configuration data are in a relational database. From these data we can generate object interfaces for equipment, configuration files for front-end computers, a read-only database for accelerator control interfacing, and full dynamic documentation on the Web. The database is also used in real time for runtime references and archives, and for the working data of several programs.

1 Introduction

The control system for the CERN PS accelerator complex (7 accelerators, not including SPS and LEP), is a generic system which can be adapted to other accelerators. It must steer the beams through the interconnected accelerators with up to 5 particle types accelerated in cycles grouped in a supercycle. This means that thousands of parameters must be changed each cycle of about 1.2 seconds. Some help from a database is required.

2 The database

2.1 History of the database

The development started in 1980 when a file system database made by us on a NORD computer was used to support a data-driven alarm system. The data expanded to cover other subsystems and, in 1986, the data were moved to a Oracle-V5 database system on a central mainframe. In 1991 they moved again to a local server, dedicated to accelerator control.

2.2 The database server

The relational database management system (RDBMS) is Oracle-V7.3, running on an IBM RS/6000 server with 256 MB of memory and 3 hard disks of 2.2 GB each. Operation is continuous with on-line backup during the night. In case of hardware malfunction, we can quickly switch to a backup machine without loss of data, but there has been no need for this in 6 years of operation on two different servers. The mean load of the database is low so that almost all requests can be serviced quickly.

2.3 Data structure

The hardware and software is described with about 100 core tables managed by the database section. Of these, 60 describe software classes or hardware types and the rest stores the attributes for the instances of these types.

An example of a hardware type description is table MODULETYPES, which lists the fixed attributes of VME, CAMAC, G64, and other module types. Some attributes, such as default VME base addresses and interrupts, are only filled in when relevant.

An example of an instance description is table MODULES which lists the location, function, addresses, and exceptions of individual modules.

When the number of attributes is too large, or when the list of attributes varies too much from class to class, we fall back on a structure like: INSTVAL = {CLASS + MEMBERNO + VARNAME + VALUE} which contains values for all the static (read-only) attributes which are defined in the class. These values can be of any type, even array, but are stored as strings.

2.4 Data input

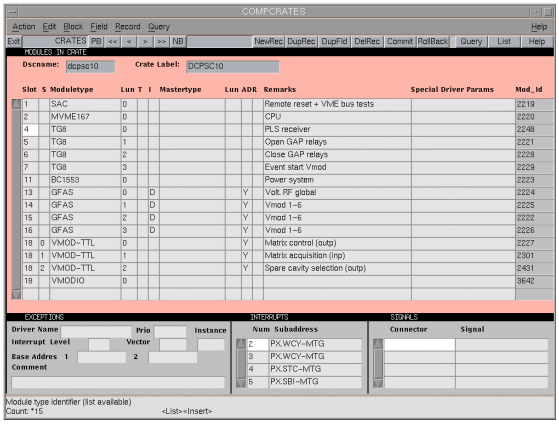

Fig1: An example of a data input screen

Input of static data is mainly through interactive forms (Fig.1), grouped in menu structures per application. The forms are generated with the help of our templates on top of the Oracle 4.5 Forms.

Some bulk loading from external data sources such as data files, spreadsheets, or other databases is also done and, occasionally, the data manager can transform the data with SQL database language statements or with scripts. Dynamic on-line data, such as accelerator settings, measurements, and user actions, are mainly written to the database by C procedures with embedded SQL.

And some of the data migrated over 17 years through 5 platform changes.

2.5 The real-time database DBRT

A subset of the data in the RDB can be downloaded in a simple real-time database called DBRT, which is based on NDBM, a commercial UNIX hash table system. This has the following advantages :

- Each accelerator can have its own copy which makes the system less dependent on the central database.
- Access is through procedure calls and the application programmer does not need a SQL precompiler.
- Access time is better guaranteed.
- Complex derived objects can be constructed off-line.
- DBRT shows a consistent snapshot of the data at the moment it was generated.

A variant of DBRT is now experimentally used for initialising Java objects through UDP network calls. This may later be replaced by a direct RDB connection through JDBC (Java Database Connectivity) or through embedded SQL in Java..

2.6 Dynamic web documentation

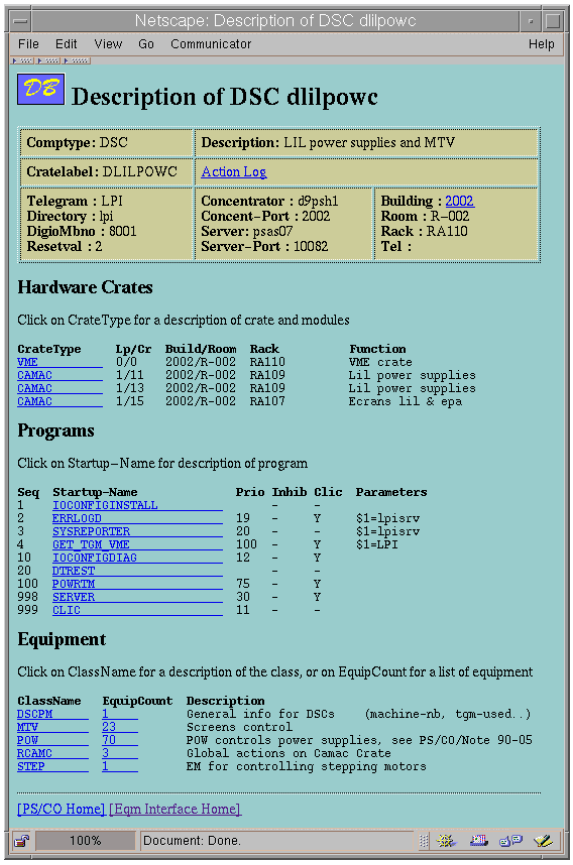

Fig.2: An example of dynamic documentation on the Web

A Web page can reference a script in the server directory cgi-bin which calls procedures in the database programming language PL/SQL. These procedures, stored in the Oracle database, construct dynamic pages (Fig.2) according to the parameters with which they are called. These pages are typically also full of links which refer to other dynamic pages and so on. At present, 113 procedures can generate tens of thousands of different pages in 113 formats. They cover the whole control system, including the structure and status of the database.

3 Application

The most important applications are described below. Support for these applications goes from major responsibility for the database section (2 persons) to merely providing the database environment.

3.1 Object oriented equipment interface

Access to the accelerator equipment is with Control Modules, which are seen by the application programmers through a uniform object oriented interface [1].

A piece of equipment is identified by its name or, alternatively, by its class name and member number. About 100 classes are defined in the database in a hierarchical structure with inheritance. All instance read-only data (constants) are also defined in the database.

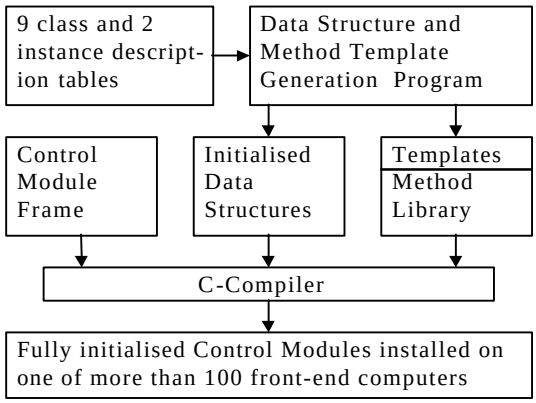

Fig3: Control Module generation

When the data are filled in (and the method library updated), a set of classes can be automatically compiled and installed in one of the 100 front-end computers (called Device Stub Controllers or DSC) each sitting in a VME crate (Fig.3).

3.2 Automatic front-end computer configuration

From information about VME and other crates and modules, equipment, and programs, a file named rc.local is generated for each DSC (Fig.4). This file contains all information for starting up and initialising the drivers and other programs which are specific to this DSC, in the proper order and with the right priority and arguments. The file contains also comments which describe the configuration of the DSC and are readable by a maintenance program [2].

3.3 Alarm system

Any control module (CM) class can inherit variables and methods from the ALARMS class, on top of its normal inheritance. This inheritance includes the generic method ALARM which depends for its execution on control and

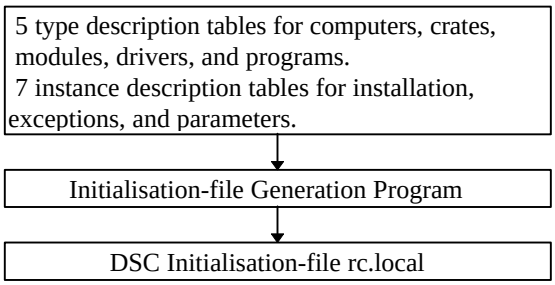

Fig.4: Generation of rc.local initialisation files

status word descriptions in its class and instance variables. These values are derived from the database [3].

An active scan program acquires a list of equipment and periodically scans the corresponding CM with the method ALARM which responds with an alarm code (Fig.5).

The display program can display the corresponding alarm messages and give details on request.

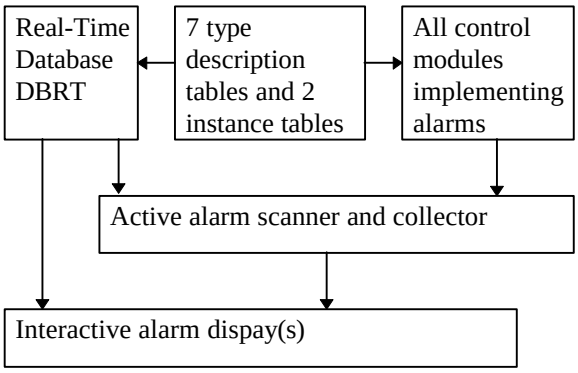

Fig.5: Data flow in the alarm system

3.4 Equipment archives and references

Each piece of equipment can have the values of several attributes stored for each of 24 virtual accelerators. These references are set to the operational values on request of the operator, individually or for any set of them. Later, the operator can use these values to restore normal operation after experiments or corruption.

Any set of reference values can be stored in a named archive. A large number of archives is allowed. These archives are used to come back to settings which gave good results in the past.

3.5 Generic user interface

After the operator logs in, he is presented with a menu interface which permits him to select the working environment and the programs to be started.

Generic programs are available for controlling equipment and for displaying values in the desired format, for single pieces of equipment or for various ensembles (Fig.6). This is aided by meta-data which specify the important properties for each class.

All this is data-driven, with data coming from the RDB via DBRT [4].

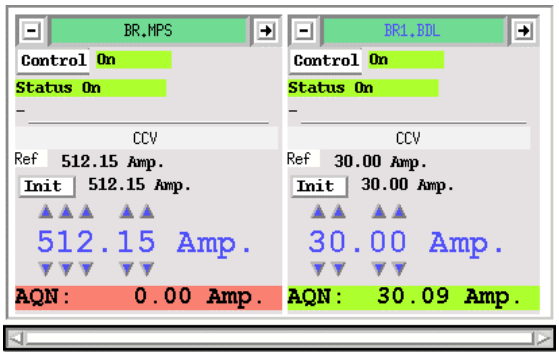

Fig.6: Knobs for setting values of accelerator parameters.

3.6 Supercycle editor

An interactive program permits the operator to set up the various beams and cycles which compose a supercycle. All this is described with tables in the RDB which also contains an archive of useful past cycles, beams and supercycles [5].

3.7 Expert systems

A few expert systems have been built, using the database as a knowledge base:

- An experimental system relating beam data and alarms [6].
- A set-up system for initialising the equipment interface and the equipment with a specially made inference engine [7].
- A system converting rules to SQL statements for checking the compatibility of cycles in a supercycle[8].

These systems work well but there is no unified approach. We think it would be logical to put both facts and rules in a relational database. Some means of indexing the rules would avoid linear searches through huge rule bases. What is missing is an inference engine integrated in the database, perhaps based on an extension of the SQL language. This marriage between RDBMS and expert systems seems natural to us but database designers have other priorities at the moment.

3.8 Data extraction from drawings

A lot of information about the control system is contained in drawings made with the help of Computer Aided Drawing systems. It would be useful to be able to extract the geographical location of a component or the number of a cable between two connectors. There are however no accepted data format standards and little provision is made for extracting information beyond part lists. Some way to query the data with at least a subset of SQL is desirable.

We tried automatic information extraction from lists that could be generated but this was not implemented because

the efforts were out of proportion to the benefits.

3.9 Other applications

Independently developed applications may use the core tables and add tables of their own, integrated with the core data by agreeing on common identifiers. Examples are:

- ABS [9] which describes beam paths, magnets, and monitors in order to calculate operational settings to steer the beam.
- NAOS [10] which controls a system for acquiring and digitising signals for display and reference.

4 Why a relational database ?

We make considerable efforts to reconstruct software objects from data in the database, so you may ask why we do not use an object oriented database. The advantages of a relational database for accelerator controls are:

- The data are in a safe environment, well protected from unintentional changes by faulty programs.
- Data access is with the standard SQL language and is not restricted to a set of closely related programs.
- The rules for modelling complex structures in a relational database are well known and rather easy to apply with some experience.
- The resulting tables are well adapted for filling-in the data through forms.
- Strong data relations are grouped in tables while weaker relations can be left implicit, which keeps the structure simple and versatile.
- Object views adapted to particular applications can be derived with SQL statements. This may be inefficient but it is straightforward and fast enough for the applications which are of interest here.
- Most large and complex control systems will grow, have frequent modifications, and often whole new subsystems have to be integrated without disturbing existing applications. All this is rather easy with a relational database.

Version 8 of Oracle, now becoming available, has many new object-oriented features, implemented on top of the RDBMS. It is not yet clear how we can use these features and whether they will make things simpler.

Of more immediate interest is support for storage, display, and querying, of complex datatypes like text, images, video, and sound. Especially important for us is support for collections such as arrays and lists. These developments fit perfectly in the relational model.

5 Conclusion

We have demonstrated that it is quite possible to store almost all configuration data and many operational data in a relational database. These data can be used to generate interface modules, install programs, and provide programs with data. The database is especially useful if the whole system is integrated and not just a number of unrelated subsystems. This requires close collaboration between the database section, responsible for the structure of the core tables, and the application programmers. An important side effect of a central data store is the availability of the information on the Web, for local use and external documentation.

References

[1] L.Casalegno, J.Cuperus, C.H.Sicard, Process equipment data organisation in CERN PS controls, ICALEPCS-89, Nucl. Instr. & Meth. A293 (1990) 412-415.

[2] J.Cuperus,A.Gagnaire, Automatic Generation of configuration files for a distributed control system, ICALEPCS-95, Nucl. Instr. & Meth. A352 (1996) 148-153.

[3] J-M. Bouche, J.Cuperus, M.Lelaizant, The data driven alarm system for CERN PS accelerators, ICALEPCS-93, Nucl. Instr. & Meth. A352 (1994) 196-198.

[4] J.Cuperus, F.Di Maio, C.H.Sicard, The operator interface to the equipment of the CERN PS accelerators, ICALEPCS-93, Nucl. Instr. and Meth. A352 (1994) 346-349.

[5] J.Lewis, V.Sikolenko, The new CERN PS timing system, ICALEPCS-93, Nucl. Instr. & Meth. A352 (1994) 91-93.

[6] P.Skarek, L.Varga, Multi-agent cooperation for particle accelerator control, Expert Systems with applications Vol.11 No.4 (1996) 481-487 (Pergamon / Elsevier Science Ltd).

[7] G.Daems, V.Filimonov, V.Homutnikov, F.Perriollat, Yu.Riabov, P.Skarek, A knowledge based control method, ICALEPCS-93, Nucl. Instr. & Meth, A352 (1994) 325-328.

[8] J.Lewis, P.Skarek, L.Varga, A rule-based consultant for accelerator beam scheduling used in the CERN PS complex, ICALEPCS-95, Nucl. Instr. & Meth. A352 (1996) 703-706.

[9] B.Autin, F.di Maio, M.Gourber-Pace, M.Lindroos, J.Schinzel, Database for accelerator optics, this conference.

[10] B.Dupuy, P.Fernier, B.Frammery, S.Pasinelli, A VXI system for observation of distributed analog signals. RT93, Eight Conference on Real-Time Computer Applications, June 8-11, 1993, Vancouver, Canada.