THE CONFIGURATION DATABASE FOR THE CERN ACCELERATOR CONTROL SYSTEM

J.Cuperus, R.Billen, M.Lelaizant, CERN, Geneva, Switzerland

Abstract

CERN has 9 interconnected accelerators, comprising the PS injector chain, the 6.9km SPS and the future LHC with a circumference of 27km. At the end of 2002, the 3 control groups for these accelerators merged and it was decided to use common control structures as much as possible. For such a large system, it is essential to use generic software wherever appropriate, driven by data in a configuration database.

INTRODUCTION

In total, 8 accelerators have to be controlled. The PS complex and the SPS accelerator each have their own control system and the future LHC has some industrial controls already designed. New controls will follow a standard design pattern but the old controls must also be accommodated. A common description in a ConFiguration DataBase (CFDB) must support the different designs and offer a unified view of the control system to the application programs. Therefore, the database tables are very general, applying to all accelerators, with special attributes dedicated to particular subsystems where appropriate. There is only one table for all computers in the system and one table for all controlled devices. The CFDB is implemented on a dedicated server with an Oracle relational database management system.

CONTROL SYSTEM CONFIGURATION

Fig.1 shows the main components of the common control system. The middleware [1] has a narrow interface (the same for all device classes), well adapted to generic programming. To give an idea of the scale of this system: for the PS complex and the SPS machine alone, there are now 590 Front-End Computers (FECs).

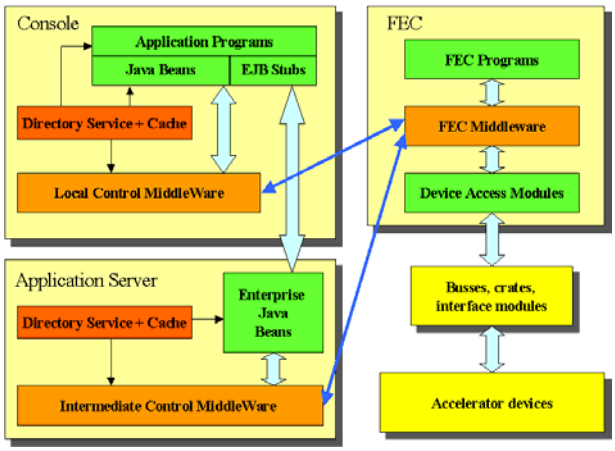

Figure 1: The main components of the control system.

ACCELERATOR CONTROL DEVICES

Accelerator devices are objects that are visible to the control system. They are identified by a *DeviceName*. They can be physical apparatus like power converters, instruments, and function generators, but also more abstact things like scope signals, timing pulses, and measurements at a critical moment. The devices, their attributes, and their relations, are described in the CFDB (see Fig.2).

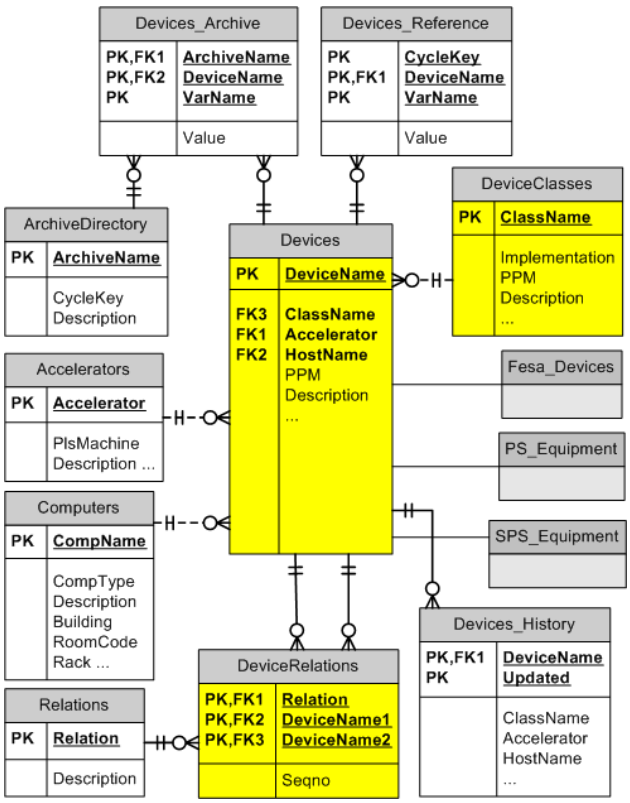

Figure 2: Entity-Relationship Diagram for device-related database tables.

A device is part of an accelerator and is controlled from a FEC Host that is a *Computer*. Similar *Devices* are instances of the same *DeviceClass*; they have settings that can be stored as *References* or *Archives*. Devices can have *Relations* with other devices (e.g. device1 is start_trigger for device2). Table *Devices_History* keeps track of all changes in table *Devices*. Views *Fesa_Devices, PS_Equipment*, *SL_Equip* show table *Devices* in forms compatible with the new and old systems. A class may be PPM (Pulse to Pulse Modulated) which means that its device settings may vary from one accelerator cycle to the next according to a *CycleKey*.

DEVICE ACCESS

Device objects are controlled through pieces of software installed on the FECs. The PS and SPS have each their own device access modules. Future developments will be done with the Front End Software Architecture (FESA). FESA modules will store their class structure and configuration parameters in the CFDB as shown in Fig.3.

Fesa_Devices are instances of *DeviceClasses* which group similar devices. A class defines a set of *FesaFields*, each with many attributes, describing a device property and its characteristics. Configuration constants, like addresses, are stored in table *FesaPersistents*. A *DeviceClass* can refer to another *DeviceClass* as its *Superclass*, and so on, forming a tree of classes. View *FesaFields_Plus* sees also the fields inherited from the superclasses. Different versions of a class definition can be implemented on different FECs.

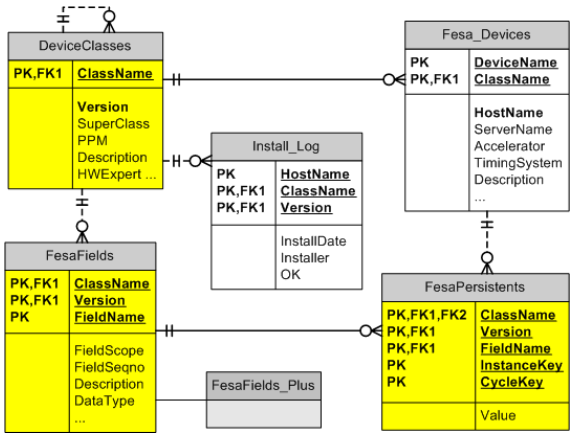

Figure 3: FESA-reated tables in the CFDB.

The environment where these data are used for FESA design is shown in Fig.4. The FESA design and installation tool accesses the database only through a Java interface library. It can create and edit the data and generate configuration files that are read by the FESA framework in the FEC at start-up.

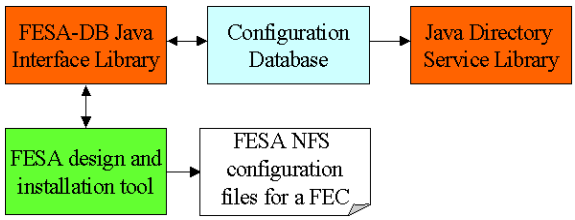

Figure 4: The FESA design and install environment

DIRECTORY SERVICE

The device access module tables, plus some additional tables, provide a complete description of the device access for application programs. These data are accessible to any Java program or component through the Directory Service library [2] as shown in Fig.5.

Devices have *DeviceData* and are instances of a *DeviceClass*. A *DeviceClass* has a set of *DeviceProperty* to access device value channels and these properties have characteristics (type, units, min, max, …). Devices of the same class can be grouped in a *DeviceGroup* and several *DeviceGroup* can be part of a *WorkSet*. A *WorkSet* can be set up in the database or constructed at runtime with a database query. A *Contract* maps to several *DeviceProperty* and allows grouped acquisition or setting of these properties for a device. *DirectoryItem* is the superclass of all these classes. A *DirectoryItem* can be obtained from static calls to the *DirService* methods.

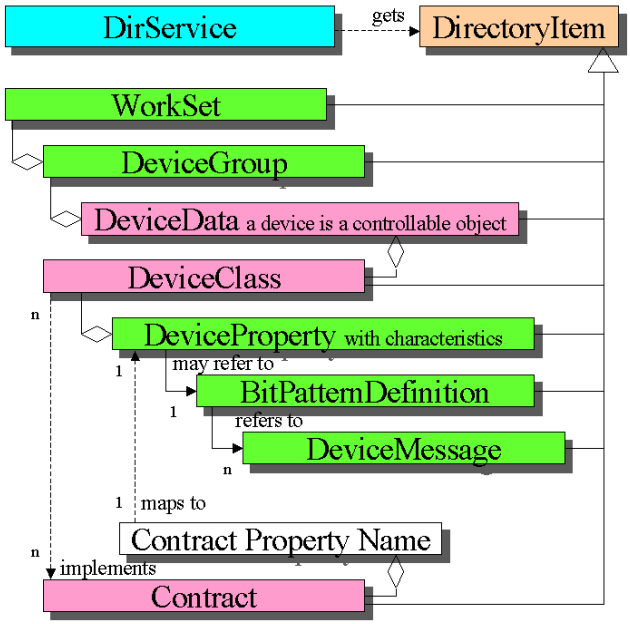

Figure 5: Java class diagram of the Directory Service.

HARDWARE CONFIGURATION

Each FEC drives a number of busses with hardware crates and modules, as shown in Fig.1. Fig.6 shows the CFDB description of these components.

Mapping of FESA address parameters (relative to a FEC) to hardware module channels is done with address strings:

- ll/cc/nn/aa (loop/crate/slot/channel) for CAMAC
- name/nn/aa (moduletype/lun/channel) for VME
- appropriate conventions for any other bus …

A set of routines converts these strings to bit patterns that the hardware drivers in the FEC can understand.

The database contains, for any FEC:

- The list of programs and their parameters
- Data for all FESA classes relevant for the FEC
- The list of drivers and hardware interrupts.

These data, together with program libraries, make it possible to automatically initialize any FEC.

A special case is the interfacing with industrial control systems like the LHC cryogenics. These come generally with their own configuration database, not necessarily in Oracle. How these databases will relate to the CFDB has not yet been decided.

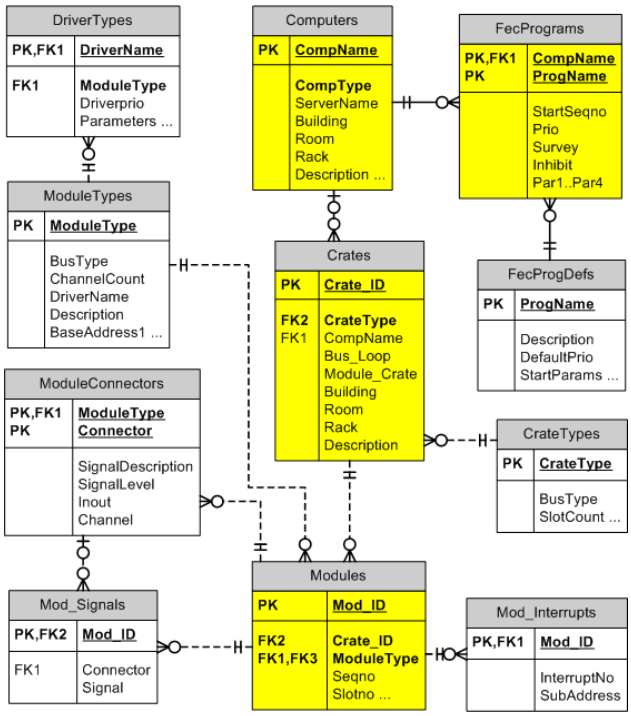

Figure 6: Configuration of FECs, hardware crates and modules.

DATA ENTRY WITH FORMS

Most data can be entered via Oracle Forms, enriched with a home-made template. The template has a query mode and an edit mode (see Fig.7). Each mode has a set of buttons for easy interactivity. The forms are at present implemented on the Linux platform but will be converted to Web forms that are platform independent.

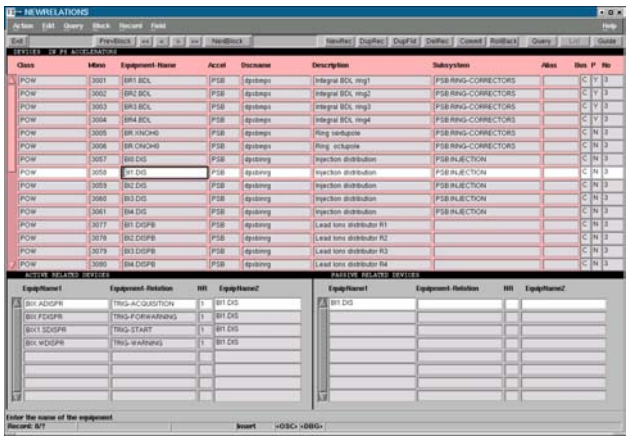

Figure 7: An interactive database entry form in edit mode. The page shows a master block and two dependent blocks. The active block has red background. A forms application can contain several pages.

ACTIVE WEB DOCUMENTATION

All information in the CFDB is accessible through active Web pages, implemented with routines written in Oracle PL/SQL with the help of the OWA (Oracle Web Access) library. Browsing can start with a general home page and any information can be obtained through following the links (see Fig.8).

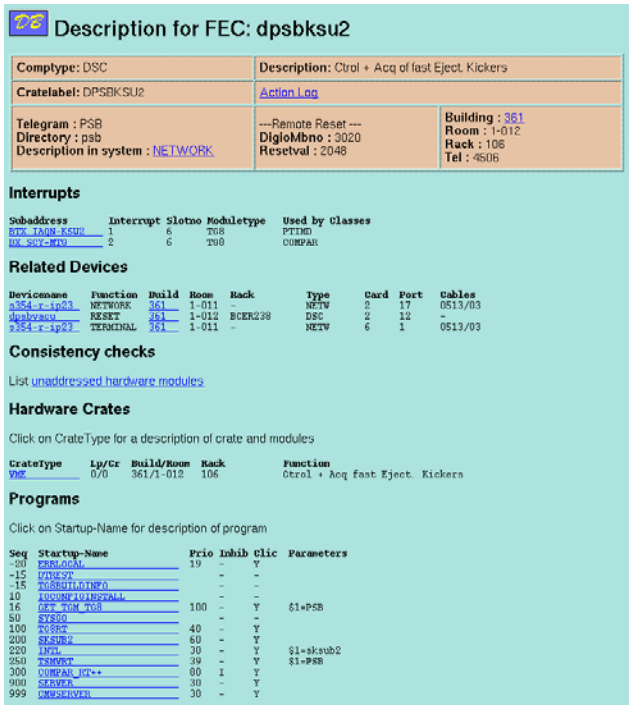

Figure 8: Example of documentation in active Web pages. Clicking on any of the links will result in a new documentation page.

RELATION WITH OTHER DATABASES

Data for the PS complex come from the precursor of this database [3]. SL equipment is defined in a table which is now part of the device table and accessible through the SL_EQUIP view. Precise geographical positions are in the Survey database. Buildings are charted in a location database. Many LHC devices are described in the LHC reference database. Beam optics are described in dedicated tables. Long cables are listed in a CERN-wide cable database. Industrial electronics may have their own dedicated databases.

CONCLUSIONS

A configuration database, implemented on a relational database management system, is a necessary component of any modern control system. It brings unity and conceptual simplicity to a large and diverse system.

REFERENCES

[1]The Controls Middleware at CERN – Status and Usage, K.Kostro, F.Di Maio, S.Jensen, J.Andersson, N.Trofimov, this conference.

[2]A Directory Service for the CERN PS/SL Java Programming Interface, J.Cuperus, P.Charrue, F.Di Maio, K.Kostro, W.Watson, Icalepcs-99, Trieste, Italy.

[3]Integration of a Relational Database in the CERN PS Control System, J.Cuperus, M.Lelaizant, Icalepcs97, Beijing, China.