DATABASE APPLICATION FOR CONTROL SYSTEM HARDWARE MANAGEMENT

R. Broomfield, M. Heiniger, T. Korhonen, T.Pal, Paul Scherrer Institute, 5232 Villigen, Switzerland

Abstract

The Paul Scherrer Institute (PSI) is operating at present two accelerator facilities, and a third one is under construction. The control system hardware for these facilities is managed by a single group. To be able to keep track of the hardware (more than 300 VME and CAMAC systems), database applications are being developed. The set of applications provide features to track the stock status as well as the life cycle of components. It will also optimise the spare parts stock and improve the overall quality of service. The paper presents the conceptual design and implementation, using object relational methods, of the Oracle RDBMS and web application server.

QUANTIFYING THE CORE DATABASE REQUIREMENTS

With the large number of hardware deployed in the control systems of PSI accelerators, the logistics of handling all the hardware becomes a burden. Without an overview of the status of hardware stock, it is almost impossible to know when new modules should be ordered, where they existing ones are installed and so on. For that reason a database application (called Controls Inventory Database or CIDB) has been developed. The purpose of it is to keep track of what hardware modules are installed, what is in stock, what hardware is in repair, on loan or otherwise not deployed. By recording the history of moving the modules, their fault history and repair data (=reasons for the fault) we will be able to accumulate a wealth of empirical reliability data, which will aid in estimating needed stock levels, finding out weak modules, doing preventive maintenance and so on. The database includes the notion of installed subsystems, so it is possible to find out which hardware modules are used in a subsystem.

Some functionality that should be relatively easy to add to the core database system would also enable us to handle and track requests for additional hardware (new subsystems), include financial information to easily find out what the costs of systems are and to store hardware details like card firmware versions and so on.

At a later stage it expected to couple this database with the control system so that the control system hardware configuration for the low-level controllers could be generated from the database.

The SLS Controls group uses predominantly VME and in particular VME64x. The rest of the Controls group is also moving towards VME. Thus the design was started using the normal hierarchy of a VME system as a guide to implementing the database, i.e. Racks can contain VME Crates, VME Crates can contain modules and VME Modules can contain Mezzanine Cards. Thus the bulk of equipment would be VME or related products. Initially this would be the case but provision would be made to incorporate all types of hardware.

TABLE DESIGN AND IMPLEMENTATION

Based on the requirements described in the previous sections, the relational model, in terms of the entities and their relationships has been built. Oracle tools [1] have been used, both to produce the server model as well as for reverse engineering during the design process. The implementation is on version 8.1.7 of the (Oracle) server engine. The entities are a set of modular repetitive pair structure, for the different hardware (Crates, Power Supplies, VME cards, Mezzanine cards), comprising of a part which describes the type of hardware (description tables) and the other the actual hardware component details (master tables).

Referential integrity is imposed, in the usual way, with primary key and foreign key constraints. The primary keys for the hardware components are the site-specific, unique identity tags attached to a particular piece of hardware, and a description identifier for the description tables. The cardinality is one-to-many, and the optionality, depending on the case, is imposed via *not null* constraints. Additional restrictions on the values are enforced by *check constraints*. Downward denormalization has been used on some tables for convenience, bearing in mind the requirement to maintain the appropriate attributes after inserts and updates.

Figure 1 shows the server model. The 'crate master' is the 'driving' table. Each crate and its contents has an association to one or more 'system names', according to a naming convention [2], which represents the accelerator or beamline devices controlled by the IOC(s) in the crate.

For the initial data capture, the *sql*loader* utility was used with a set of comma separated variable files, mapping to the individual tables, after a process of consolidation from legacy data. The table loading sequence was according to the foreign key constraints.

Inconsistencies in the data sets, with respect to the constraints, were written to a log file, and after correction, the respective tables updated as a subsequent step.

The total number of rows, in all tables, for the initial data is approximately 4000. The timestamp and identity of the user performing inserts and updates is recorded, for each table, as is common practice. In addition, however, a complete set of ‘history’ tables (not shown in figure 1) were created, which mirror the actual tables themselves, and record the Data Manipulation Language (DML) action, as well as the old/new values of the attributes as appropriate, via triggers.

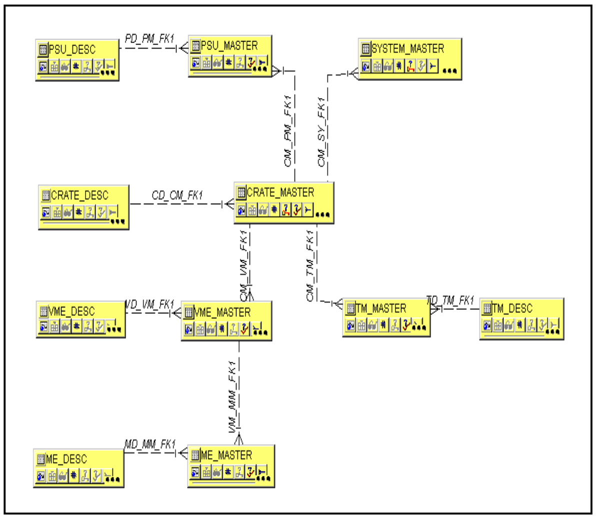


Figure 1: Server model diagram (abbreviated).

WEB APPLICATIONS

One of the objectives is to provide a versatile set of interactive web applications to access the data, and display the results. Oracle WebDB (Portal) is used, and provides thin client HTML interfaces, via the Oracle application Server (9iAS), respectively [1]. The application component wizards of the tool allow for rapid application development, in the case of simple forms and reports. However, due to the limitations of the wizards, as well as the inherent limitations of the web, such as its stateless nature, it is difficult to store and maintain variables. To overcome these, concerning the more complex applications, with interactive feedback, it is necessary to use *pl/sql* stored procedures and *JavaScript*.

The web applications have been structured in a menu, according to the functions, such as deploying equipment to a system, returning it to stock or to ascertain the availability. Access security and privileges to execute components and modify the data is granted via (database) schemas and roles.

The applications require between five to eight interactive steps via forms and reports components to insert or update equipment. In order to optimize performance, (Oracle) *collection* constructs are used to bulk-bind the output of queries against tables, in the *pl/sql* stored procedures associated to the application components.

As an illustrative example, Figure 2 shows a generic flow diagram corresponding to moving equipment to a system. The basic form and report templates are, in this way, reusable, and the *sql* query is customized for the specific type of hardware component table, depending on the application.

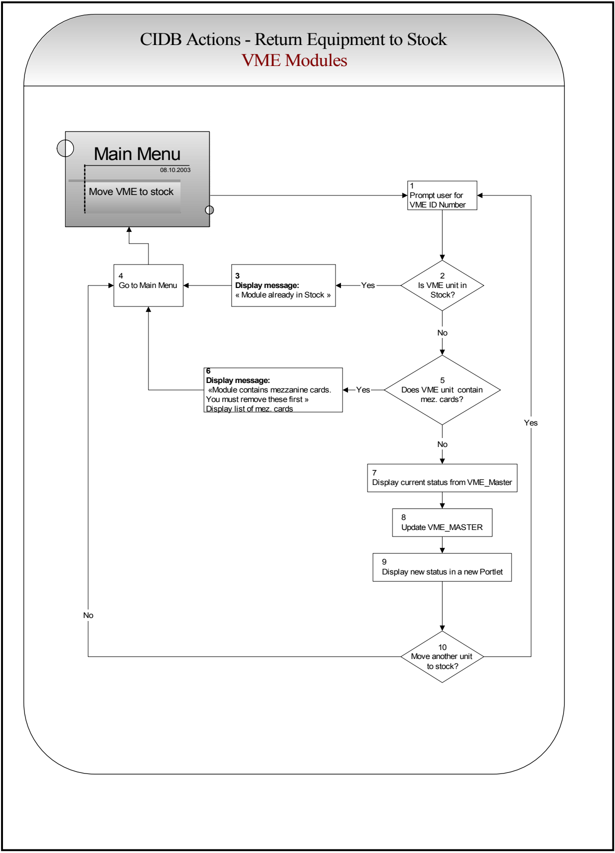
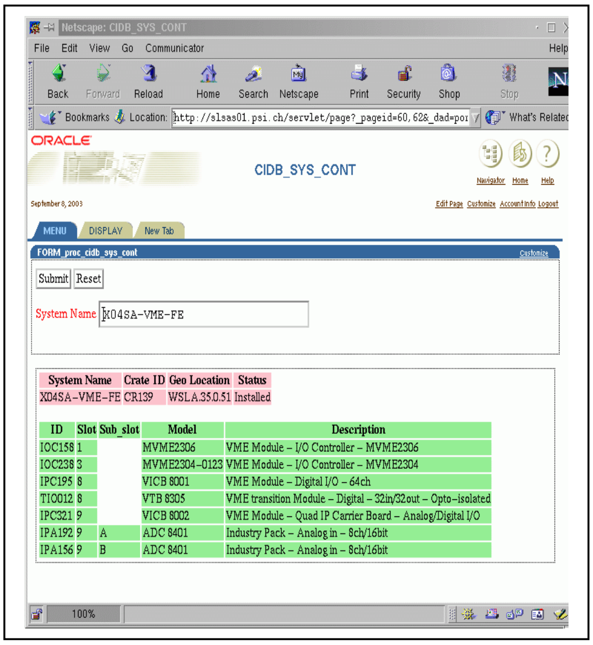

CIDB Actions - Return Equipment to Stock
VME Modules
Main Menu
08.10.2003
Move VME to stock
1 Prompt user for VME ID Number
2 Is VME unit in Stock?
Yes
No
3 Display message: « Module already in Stock »
4 Go to Main Menu
5 Does VME unit contain mez. cards?
6 Display message: «Module contains mezzanine cards. You must remove these first » Display list of mez. cards
7 Display current status from VME_Master
8 Update VME_MASTER
9 Display new status in a new Portlet
10 Move another unit to stock?

Figure 2: Flow diagram – move VME module to Stock


Netscape: CIDB_SYS_CONT
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Location: http://slsas01.psi.ch/servlet/page?_pageid=60,62&_dad=por
What's Related
ORACLE
CIDB_SYS_CONT
Navigator Home Help
September 8, 2003
Edit Page Customize Account Info Logout
MENU DISPLAY New Tab
FORM_proc_cidb_sys_cont
Customize
Submit Reset
100%

System Name X04SA-VME-FE

System Name	Crate ID	Geo Location	Status
X04SA-VME-FE	CR139	WSLA.35.0.51	Installed

ID	Slot	Sub_slot	Model	Description
IOC158	1		MVME2306	VME Module – I/O Controller – MVME2306
IOC238	3		MVME2304–0123	VME Module – I/O Controller – MVME2304
IPC195	8		VICB 8001	VME Module – Digital I/O – 64ch
TIO012	8		VTB 8305	VME transition Module – Digital – 32in/32out – Opto–isolated
IPC321	9		VICB 8002	VME Module – Quad IP Carrier Board – Analog/Digital I/O
IPA192	9	A	ADC 8401	Industry Pack – Analog in – 8ch/16bit
IPA156	9	B	ADC 8401	Industry Pack – Analog in – 8ch/16bit

Figure 3: Web Portal application components.

At the time of writing this article, approximately 30 applications exist, built using webDB components. These are in the process of being transformed to portal components and *portlets*, which offer added value to the end user, such as to reduce the number of navigation steps, to different pages, for a specific task. Figure 3 shows an example of a page, comprising of two portlets: the upper is a form component, and the lower a report component.

On click of the form's submit button, an event handler passes the input parameter(s) to the report component, and after processing, the results are displayed, after a page refresh, on the same HTML page. Page tabs, also visible in figure 3, are used to group sets of applications, again with the goal to reduce the time needed to search for the appropriate application component.

The development of the web applications has been the majority fraction (~ 2/3) of the total time resource. From our experience, it is unlikely that this fraction can be reduced significantly in the future, as it is dominated by the development time for the customized requirements, albeit taking into account the reuse of *pl/sql* library packages, common to the applications.

SUMMARY AND OUTLOOK

A core set of functions for the database application has been put into operation, and is in use for the control system hardware management. This has enabled to overcome legacy implementation, and provide a common repository for the accelerator facilities at the PSI.

Oracle *object types* are being progressively introduced, with the existing tables, which enable to encapsulate the underlying structures of the hardware components (e.g. VME modules contained in a crate), with a corresponding simplification in the *sql* and *pl/sql* constructs for the DML statements.

Extensions to the core functions, such as, to keep track of the information concerning the reliability and life cycle of the components, although available in an indirect manner from the existing tables ('history' tables), have to be fully integrated into the set of applications. It is also envisaged to implement functionality in order to store time-dependant calibration data for the appropriate hardware components.

REFERENCES

[1] Oracle Corporation, http://www.oracle.com

[2] A. Streun, SLS Functional Device Naming Convention, http://slsbd.psi.ch/pub/slsnotes/naming