CMS DCS DESIGN CONCEPTS

R. Arcidiacono[8], V. Brigljevic[9], E. Cano[4], S. Cittolin[4], S. Erhan[4,5], D.Gigi[4], F.Glege[4], R.Gomez-Reino[3,4], M. Gulmini[1,4], J. Gutleber[4], C. Jacobs[4], G. Lo Presti[4], G. Maron[1], F. Meijers[4], E. Meschi[4], S. Murray[7], A. Oh[4], L. Orsini[4], M. Pieri[6], L. Pollet[4], A. Racz[4], P. Rosinsky[4], C. Schwick[4], I. Suzuki[7], J. Varela[2,4]

[1]*INFN - Laboratori Nazionali di Legnaro, Legnaro, Italy;* [2]*LIP, Lisbon, Portugal;* [3]*Universidad de Santiago de Compostela, Santiago, Spain;* [4]*CERN, Geneva, Switzerland;* [5]*University of California, Los Angeles, Los Angeles, California, USA;* [6]*University of California, San Diego, San Diego, California, USA;* [7]*FNAL, Chicago, Illinois, USA;* [8]*Massachusetts Institute of Technology, Cambridge, Massachusetts, USA;* [9]*Rudjer Boskovic Institute, Zagreb, Croatia;*

ABSTRACT

CMS uses PVSSII [1] and the Joint Controls Project (JCOP) framework [2] to develop the Detector Control System (DCS) of its experiment. PVSSII is a commercial SCADA system and the JCOP framework is an in-house software project of CERN. These tools enable developers to create control applications for a supported set of hardware devices. In addition, other commercial and CERN-made hardware/software solutions are used where necessary. The JCOP framework contains a Finite State Machine (FSM) toolkit. Sub-detector and service experts use this toolkit to build control trees based on FSMs. CMS has put policies into place to ensure a homogeneous and coherent use of its DCS. The different control systems of the experiment have been integrated into a single control tree, whose top node is referred to as the CMS DCS Supervisor. This supervisor manages the control systems of all sub-detectors and support services. These control systems are distributed among different machines and run on different operating systems and platforms. CMS developed new software and concepts in order to integrate its DCS. A plug-in system was developed to facilitate the load balancing of control processes across DCS PCs. Software maintenance was eased through the introduction of a central software repository. A monitoring system was written to provide a 3D view of the experiment that shows errors and their geometric correlations. A rack control and monitoring application was also developed for the experiment, and this application has been included in the JCOP framework to be used by all LHC experiments.

INTRODUCTION

The CMS collaboration comprises thirty six countries and more than two thousand scientist and engineers. Like most of CMS experiment's tasks, the Detector Control Systems design and implementation is distributed among collaborators. Members of such a big collaboration are not only CERN based but also spread all over the globe, working in multiple universities and research institutes. The integration process of putting together all the controls developed by the different collaborators is a challenge for the DCS developers of the CMS experiment. The CMS DCS central team has put into place a set of policies managing and governing this activity. These policies will ensure the homogeneity and coherence of the final integration result: the overall CMS control system.

PVSSII AND JCOP FRAMEWORK

PVSSII was selected in 1999 by the Joint Controls Project team (JCOP), which results from a collaboration between the four LHC experiments and the CERN IT/CO group. This selection [3] was done after an evaluation in which PVSSII competed with forty other commercial SCADA systems. SCADA products were tested against criteria [4] such as: scalability, extensibility, stability, cross-platform integration, graphical user interface flexibility, archiving and trending capabilities, remote access, security, alarm handling and documentation. After applying these selection criterions, PVSSII, from the ETM company, appeared to be the best solution. Based on PVSSII the JCOP team built a controls framework integrating the requirements of the LHC experiments. The so-called JCOP framework, today extensively used among Large Hadrons Collider (LHC) experiments, allows to create controls for a set of commonly used hardware devices in an automated way. The framework

also includes a Finite State Machine toolkit to create control hierarchies. This FSM toolkit, based on SMI++ [5], combines finite state machines behavior with expert system like rules. The Delphi experiment in collaboration with the DD/OC CERN group developed the State Management Interface (SMI) and demonstrated its potential during the LEP era. SMI++ models a control system using two different types of objects: a device object type whose state is connected to hardware devices and a logical object type that behaves as a finite state machine. Using these tools, DCS developers have sufficient flexibility to create control trees for their equipment. DCS policies do guarantee that control applications are developed in such a way that their integration into the global DCS is done in a smooth and coherent way and future maintenance effort is reduced as much as possible.

INTEGRATION POLICIES

DCS policies exist in the format of an Integration Guidelines [6] that was provided to all CMS sub-detectors DCS groups. These guidelines are the outcome of several months of investigation involving sub-detector control groups and a central CMS DCS group.

CMS FSM hierarchy

FSM trees are created using logical FSM nodes to model the control logic plus FSM device leaf nodes connected to hardware. Each sub-detector DCS group has the duty of building an FSM node tree control layer modeling their system. CMS will integrate all these (sub)trees into a single FSM tree. A set of naming and structuring rules make sure that the different branches of the overall tree look similar. Unlike in industry, people that come to research institutes usually stay for short periods of time. Homogeneity reduces the learning curve for future DCS maintainers, users and operators. Naming rules have been made not only for tree node names but also for FSM node command and state names. DCS groups are required to match their command and state concepts with the command and state names provided by the integration guidelines. The most top node of the hierarchical tree-like structure is called the DCS Supervisor node. The Supervisor node summarizes the overall state of the experiment's services and sub-detectors. Since it is the main DCS entry point, the Supervisor node is in charge of receiving commands, reporting states and forwarding alarms to Run Control during a physics data taking run. The top node of each sub-detector's FSM tree is directly under the control of the Supervisor node. Command and state lists for these nodes are similar. A sub-detector can be in four predefined states:

- *'ON' - Sub-detector is ready for data taking*
- *'STANDBY' - High voltages are switch-off but low voltages are present*
- *'OFF' - Both high low voltages are switched-off (used for long shut down periods)*
- *'ERROR' - A manual intervention is required and no data taking is possible*

This naming homogeneity tends to disappear in the deeper layers of sub-detector FSM trees. This is natural, because the deeper layers of the FSM trees are closer to the hardware, and the hardware of each detector is different.

Detector Finite State Machine Trees

Detectors trees are divided in two different FSM sub-trees. One provides a detector oriented view of system electronics. It is a view that follows the structure of a detector's partition in timing and trigger zones. The other sub-tree provides a hardware oriented view of the detector equipment, grouping detector hardware by subsystems (High Voltage, Low Voltage, Cooling, Gas, etc.). Both trees will present the same data and information about the electronics equipment; however a user in different contexts will find one of the views much more useful than the other. The detector oriented view can be more efficient when identifying a problem related to a certain detector partition and it physics related state. The hardware oriented view can be more advantageous when investigating a subsystem, such as the High Voltage system, that may be affecting several detector partitions.

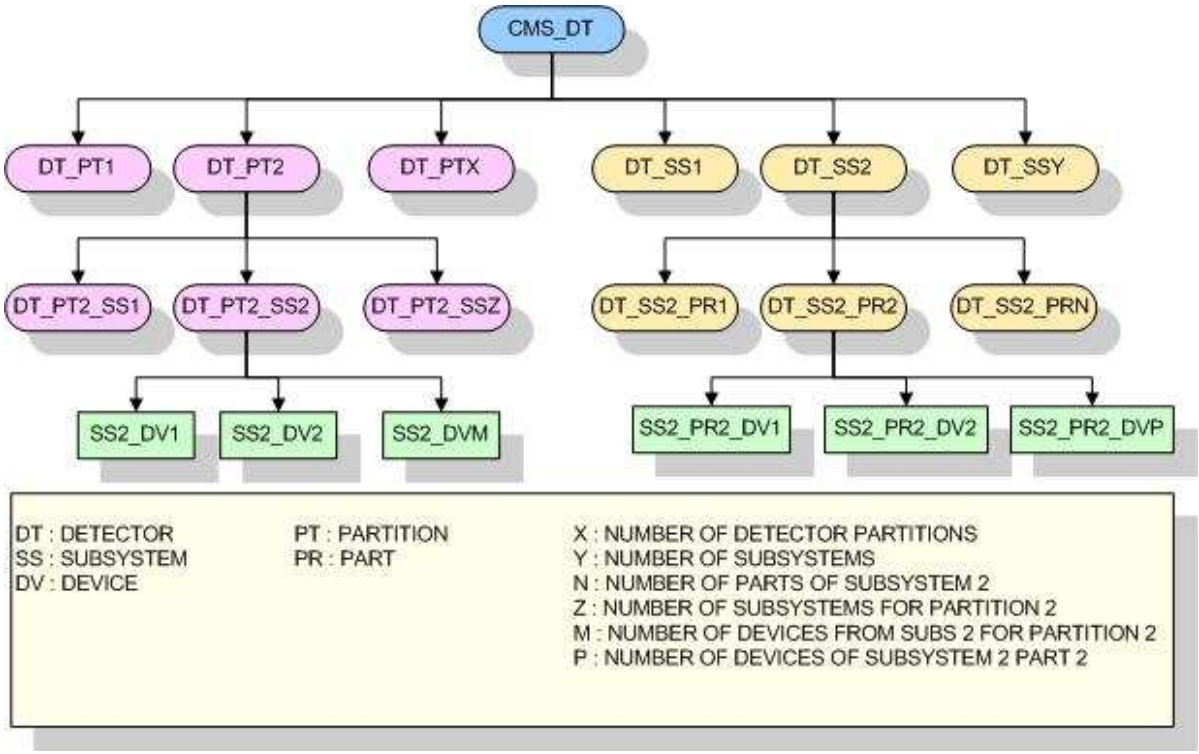

Figure1. Detector Control Trees

The example on figure 1 shows a generic detector control tree with *X* main partitions and *Y* subsystems. Detector partitions could have other sub-partitions as children nodes. '*DT_PT2*' could be the parent of '*DT_PT21*',' *DT_PT22*', etc. Subsystems could also have sub-parts as children of system parts. '*DT_SS2_PR2*' could be the parent of '*DT_SS2_PR21*', '*DT_SS2_PR22*', etc. The number of node layers depends on the detectors complexity. The figure shows one layer level of detector partitions and one layer level of subsystem parts. This simplification does not affect the illustration of the different functionality of each tree. The detector partition '*PT2*' on the figure has a children node for each of the '*Z*' subsystems. This number '*Z*' does not necessary have to be the number of subsystems '*Y*' existing for the detector. '*Z*' could eventually be smaller than '*Y*' since some of the detector subsystems might not concern a particular detector partition. Devices in detector partitions tree are references to devices in the detector subsystems tree. For the partition '*PT2*' and subsystem '*SS2*', '*M*' devices are connected as children. These 'M' devices belong to subsystem '*SS2*' tree but they can be children of any '*SS2*' part node in the subsystem tree. This means from the hardware tree point of view that device children of a particular subsystem part can be referenced by different detector partitions. Then, a problem in the device nodes of a particular subsystem part '*SS2_PR2*' could affect more than one detector partition nodes ('*DT_PT3*', '*DT_PT4*', etc.).

Services Controls

Some global services controls also implement an FSM layer. Example services include: gas control, magnet control, cooling system, environmental monitoring, rack control, and safety systems. The implementation of some of the services is provided centrally by CERN groups to the LHC collaborations. Their FSM trees are also integrated under the CMS Supervisor management.

A rack control application, controlling the services offered by the rack system for the operation of detector electronics, was developed by CMS. This rack control software is included in the JCOP framework and therefore used by all LHC experiments. This software was created in the frame of a joint Rack Control Project in which LHC experiments and technical CERN groups (ST-EL) collaborated. The rack control system renders racks in their respective experiment zones and geometrical positions. For each rack, different user configurable properties are presented via colored LEDs and commands can be sent via different popup menus. Rack properties can represent

measurements acquired by any mean. Archiving and alert configuration is possible for individual rack properties. Trends can also be seen for any rack property.

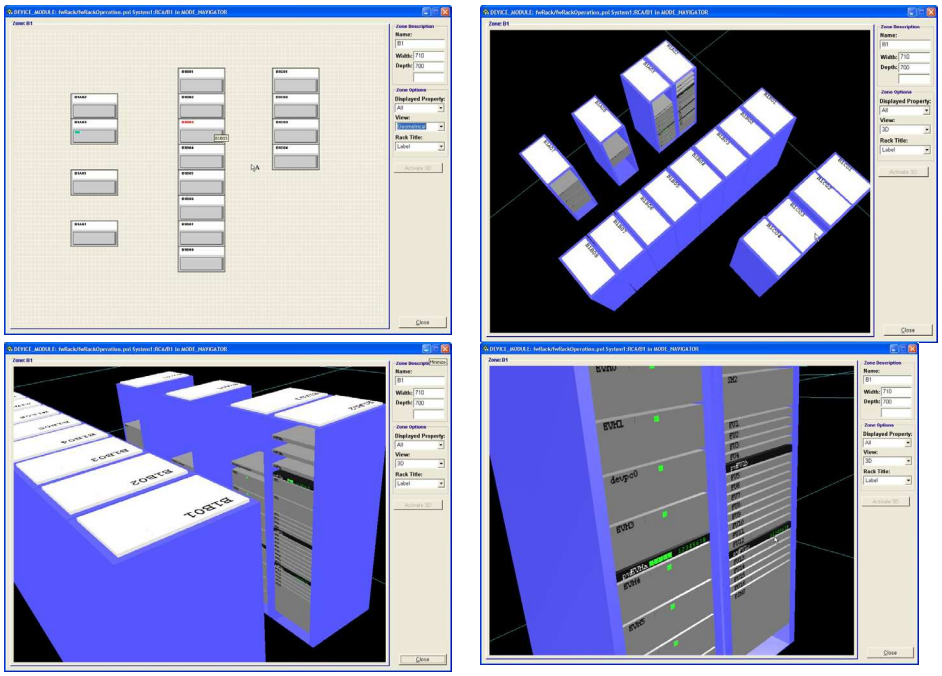

Figure 2. Rack Control software view panels.

A multilayered architecture enhances the extensibility of the rack software. New modules controlling and/or monitoring hardware can be inserted into this package using a provided interface layer. Modules publish rack properties and commands via this interface layer. Once properties and commands are published they are handled and configured by the rack software core. Included in this core, a database interface to the Equipment Management Database (EMDB) automatically loads the experiment zones and racks with their respective geometrical positions. Two rack modules already exist for experiment use. A module to control the CANALIS power distribution system enables the possibility of controlling the power of individual experiment racks and displays the status of individual rack power breakers. Alarm configuration is automatically done for these properties. It enables the monitoring of breaker trips and opening and closing failures among other possible technical problems. Another module connects with the monitoring units installed in every rack. These monitoring units are based on a generic monitoring interface board called ELMB (Embedded Local Monitor Board) [7]. Monitored parameters are: temperature, humidity, rack door status (open or closed), ELMB card status (in situ or not), thermo-switch status (open or closed), air flow, current trough turbine's transformers and water leaks. Alarm configuration is also automatically done for all of these properties. The rack control software package also includes an FSM layer. All modules inserted into the package are automatically integrated in FSM rack state evaluation. This layer is used to define actions to handle different situations and report problems to the DCS Supervisor.

CMS DCS FRAMEWORK

Plug-in system and central repository

A plug-in system was designed to facilitate the installation of different detector software components. Detector DCS software is installed on different PCs. These PCs run different operating systems. Very often, pieces of DCS software designed by different developers are put together in the same PVSS project. A mechanism ensuring proper and conflict free installation is therefore essential. More important, this installation system facilitates load balancing of the control processes across DCS PCs. Detector parts can be de-installed from overloaded projects and reinstalled in others with lower load. Ideas and tools from the JCOP framework have been used to design this system. The JCOP framework is structured in a core complemented by software components. Both the core and the components can

be installed in a PC using a provided installation tool. Each framework component deals with different hardware or control system topics. For a given PVSSII project, users can decide which components they need to use and then install them using the framework installation tool. This tool also enables the user to remove a component that is no longer needed or to update it with a newer version. A similar tool is used to install different parts of the CMS detector controls. Even it is possible for a PC to run multiple PVSSII projects, in CMS only one project runs in each DCS machine. Even more, PVSSII is installed as a system service and configured to restart after a power cut. It also allows the pre-selection of a project so that PVSSII automatically start it. This configuration allows CMS to safely run one PVSSII project in each DCS machine. In CMS, projects belonging to a sub-detector, share a single JCOP framework installation located in a network drive. Also, and following the concepts of the JCOP group, sub-detectors have a sub-detector framework installation. This sub-detector framework is made up of the different pieces of the detector control software. These pieces can be installed in a PVSSII project using the JCOP framework installation tool. The CMS DCS detector framework is the sum of all the sub-detector frameworks. DCS developers have the responsibility of implementing their control software in the format of an installable detector framework component. Instructions concerning how to do this are included in the Integration Guidelines. Sub-detector framework software is also installed in a shared network drive. This ensures that at any moment in time, DCS projects are using the same software versions simplifying maintenance and software updates. Using this approach, every project in the system is potentially able to display pieces of information belonging to other parts of its detector. Furthermore, since the CMS Supervisor will map all sub-detector frameworks, the Supervisor will be able to display and access information concerning all of them.

Alarms with geometrical correlations

The inclusion of 3D information in the DCS systems can provide geometrical alarm correlations that could not be easily obtained by other means. As an example, unstable particle beams in the accelerator could produce several errors (voltage trips, etc.) in various parts of the sub-detectors. Identifying this beam problem in a classical 2D view may be difficult. Most probably, this visual information would need to be accessed via different panels, and then correlated. On the other hand, a 3D view can show that alarms are located around a certain region of the detector. This helps identify the problem faster.

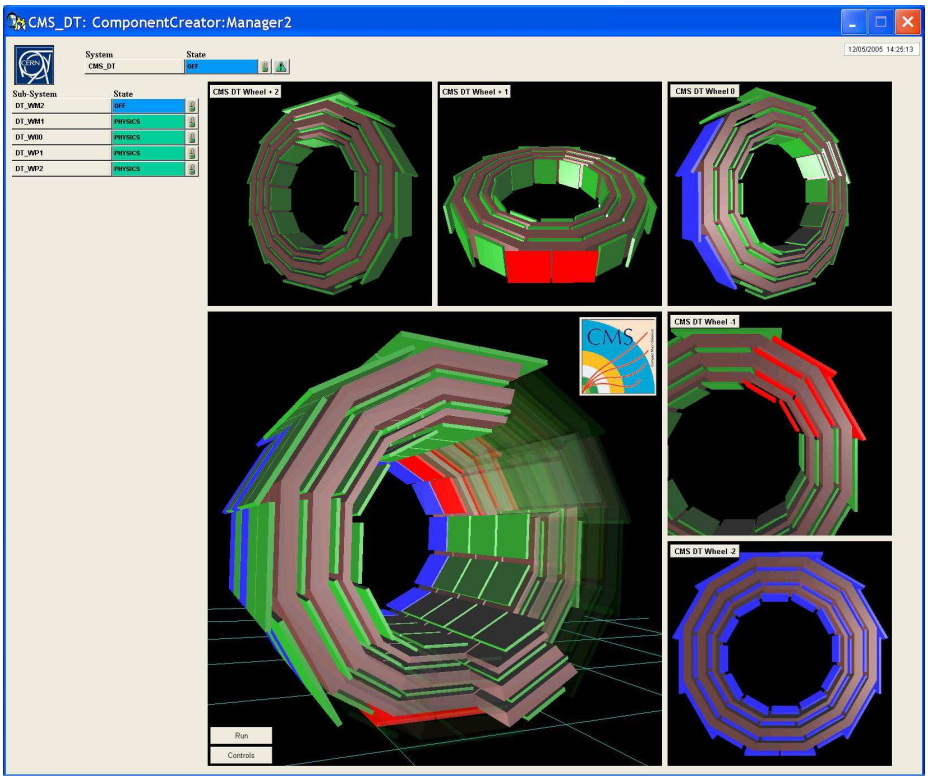

Figure 3. CMS Drift Tubes detector user interface panel example.

The 3D visualization of sub-detectors not only facilitates error correlation within a sub-detector but also across different sub-detectors. An alarm triggered by a sub-detector control system could be caused by a problem in another sub-detector. A 3D view of both detectors together would help in the error correlation. A 3D framework has also been provided to the CMS sub-detector DCS groups. Using this framework, developers can link FSM states to 3D visual detector objects. Detector partitions are represented by these 3D objects. Objects are colored in order to display partition states transitions. These colors follow the JCOP framework design standards. The 3D panels are completely interactive and provide any desired view, from any position and angle. Object transparency and invisibility is also possible. This allows any possible section view of one or more detectors. By clicking on a certain 3D object a user can get detailed information related with the linked detector partition. The geometrical description and position of detector partitions is loaded from the EMDB. This description is based in the Geant4 [8] detector description used for physics reconstruction. PVSSII does not provide a Java interface. However, it does support ActiveX objects. An ActiveX bridge [9] provided by Sun is used to integrate Java3D technology into PVSSII panels. Once again, homogeneity is gained by means of this visually powerful 3D framework used by all CMS sub-detectors.

SUMMARY AND CONCLUSION

Standard controls development tools have been provided to CMS DCS developers. Naming and structural rules homogenize and enhance the versatility of the FSM system layer. Global services are provided within the CERN JCOP; the Rack Control software package is a CMS contribution to this joint project. A plug-in system and a central repository do ensure an easy maintenance and software tidiness. A powerful 3D tool allows for a real time detector error correlation. CMS has built already a solid DCS skeleton of its final experiment DCS.

REFERENCES

[1] ETM, PVSSII - http://www.etm.at
[2] JCOP framework - http://itcobe.web.cern.ch/itcobe/Projects/Framework/welcome.html
[3] “Selection and Evaluation of Commercial SCADA Systems for the Controls of the CERN LHC Experiments.” A. Daneels, W. Salter, CERN, Geneva, Switzerland
http://www.elettra.trieste.it/ICALEPCS99/proceedings/papers/ta2o01.pdf
[4] “JCOP experience with a commercial scada product, PVSS” P. C. Burkimsher
http://itcobe.web.cern.ch/itcobe/Services/Pvss/ScadaLab/ConferencesPresentationsEtc/2003Icalepcs/JCOPExperienceWithACommercialScadaProductFinal.pdf
[5] “SMI++ - Object Oriented Framework for Designing Control Systems for HEP Experiments” C. Gaspar, B. Franek. http://smi.web.cern.ch/smi/papers/CHEP97.PS
[6] CMS DCS Integration Guidelines document -
http://rgreino.home.cern.ch/rgreino/Archives/Work/DCSIntegration1.2.ppt
[7] ELMB related information -
http://atlas.web.cern.ch/Atlas/GROUPS/DAQTRIG/DCS/ELMB/DIST/ELMBdoc.html
[8] Geant4. http://wwwasd.web.cern.ch/wwwasd/geant4/geant4.html
[9] ActiveX bridge. http://java.sun.com/j2se/1.4.2/docs/guide/beans/axbridge/developerguide