EXTENDING THE CAPABILITIES OF SCADA - DEVICE MODELLING FOR THE LHC EXPERIMENTS

L. Del Cano on behalf of the JCOP Framework Working Group*, CERN, Geneva, Switzerland

Abstract
The JCOP Framework is one of the subprojects of the Joint Controls Project (JCOP), which is a collaboration between the four LHC experiments and CERN. By sharing development, a reduction in the overall effort required to build and maintain the experiment control systems can be obtained. As such, the main aim of the Framework is to deliver a common set of software components, tools and guidelines that can be used by the LHC experiments to build their control systems. The Framework is based on the industrial Supervisory Control And Data Acquisition (SCADA) tool PVSS [1] for the supervisory layer; it extends the functionality provided by customizing it for the specifics of the LHC experiments. A part of this customization is the inclusion of devices commonly used in the experiments (e.g. high, low voltage supplies, and PLC devices). Access to an external configuration database is a further extension provided by the Framework. This paper first gives an overview of the PVSS capabilities, which are needed to understand the Framework development, described afterwards, and then introduces the Framework itself, its philosophy and its scope. The remainder of the paper concentrates on the device modelling (abstraction layer) added by the JCOP Framework as well as the configuration database subproject.

INTRODUCTION
The JCOP [2] project was started to address common aspects of the LHC experiments’ Detector Control Systems (DCS). As a JCOP subproject, the Framework [3] was set up to identify and develop these common components, reducing the effort to be put into the development of the final control systems. The main advantages of this approach are to avoid duplication of work, achieve homogeneity, and ease the integration, operation and maintenance of the resulting systems.

The Framework is currently undergoing a redesign based on feedback from the experiments’ teams after usage of early versions over a period of two years.

PVSS CAPABILITIES
PVSS provides a development environment with a Human Machine Interface (HMI) and scripting/API (Application Programming Interface) capabilities. It also supplies standard SCADA mechanisms for alarm handling, access control, archiving and trending as well as various interfaces to external hardware and software.

PVSS can run in a distributed manner (either as one single system or as various interconnected systems) and it has multi-platform support (Linux and Windows).

A key PVSS concept is the data point. Every object is modelled using data point types, which are a collection of data point elements. Each element can be configured to be archived, to trigger alarms, etc. Data points are particular instances of a data point type. A data point type is somewhat analogous to an object oriented class in the sense of a collection of attributes that provides inheritance.

In Figure 1 we see a data point type representing a simple high voltage channel. This has a set of read (readings) and write (settings) parameters, represented as data point elements associated with it as well as related display information, in this case the name of the panel to be used.

Figure 1: Example of a PVSS data point type representing a high voltage channel.

Another important PVSS feature is the so-called reference panel. The same panel or graphic symbol may be required many times in a controls application. To ease development, PVSS provides the possibility to create a single symbol or panel that can be instantiated each time with different parameters. This can be done at development or run time.

THE FRAMEWORK
The Framework mandate is to develop components that can be used by the experiments to create their own control systems. It should also compile a set of guidelines and recommendations that, if followed, will ensure homogeneity in the DCSs.

Framework components are divided in two groups, devices and tools.

* Major contributors: Belinda Chan, Jim Cook, Clara Gaspar, Frank Glege, Diana Gonzalez, Manuel Gonzalez, Philippe Gras, Oliver Holme, Hervé Milcent, Sorina Popescu, Sascha Schmeling, Fernando Varela


**Devices**

A device is an entity that represents a piece of hardware or a logical group of those entities. Detailed information on how the Framework represents and manages devices is given later in the paper.

The following devices have already been integrated in the Framework:

- Generic analog and digital channels
- CAEN Power Supplies
- Wiener Power Supplies
- Wiener Fan Tray
- ELMB [4]
- Accelerator (beam line data)
- ISEG Power Supply (currently under development)

To facilitate the integration of new devices into the Framework, a document containing the detailed procedure is provided.

**Tools**

The following tools are currently available in the Framework:

- Device Editor and Navigator: Allows one to browse the control system. It can be used to build device hierarchies and to manage devices.
- Component Installation: This tool facilitates the installation of the different Framework components.
- Controls Hierarchy: Allows the definition and operation of hierarchies of devices behaving as Finite State Machines.
- Mass Configuration Tool: Can be used to create and configure a large number of similar devices.
- External Alarm Handler: Alarms generated by a process outside PVSS can be injected into PVSS with this tool.
- Trending Tool: This tool extends the PVSS trending capabilities by providing pages of plots, templates and other extra features.
- Generic External Handler: Permits the integration of existing C/C++ code into PVSS panels and scripts.
- PC Monitoring Tool: Allows the monitoring of (remote) PCs running either Windows or Linux.

The list below shows other tools currently under development:

- Access Control: The aim of this tool is to provide a coherent access control mechanism for Framework components.
- Configuration Database Tool: This tool should be used to store and retrieve different sets of configuration data for the components of a control system in and from an external database.
- Data Storage and Retrieval: This utility will make possible the storage of detector conditions in a dedicated external database for later offline use.

The Framework is developed and distributed in a modular way. There is a core component that includes general data and libraries used by other components. Both the Device Editor and Navigator and the Mass Configuration tool are delivered as part of the core. On top of this, the user can independently install each of the other devices and tools.

It is important to mention that every Framework component is delivered with a user guide and a development guide to ease maintenance.

**Other Framework deliverables**

Besides components, the Framework provides a guidelines document [5] that includes recommendations for development and integration, as well as conventions for naming and look-and-feel issues.

Since 2002 a Framework tutorial [6] is available to be followed as a self-training or as a one day course that complements the PVSS training given by ETM.

**DEVICE MODELLING USING THE FRAMEWORK**

As explained above, part of the scope of the Framework is to model the most commonly used devices in a coherent way. From an abstract point of view a device is an object with a number of properties and certain behaviour. The first approach to model a device in PVSS could be to consider a device type as a data point type in PVSS. Thus, a device instance would map to a data point, whose data point elements correspond to the device properties.

PVSS functionality lacks however some necessary features, such as default values and information on which properties can have a certain configuration; to fill this gap, the Framework also stores in the PVSS database the default and potential configuration for each of the device properties.

Regarding device behaviour, this is encapsulated in function libraries, but to facilitate the management of devices instances, the Framework also supplies a set of configuration panels. These panels provide the user with functionality to create, delete and configure devices; they integrate generic Framework mechanisms for setting up the connection to the hardware (through the industrial standard OPC or DIM [7], a CERN made protocol), configuring archiving and alarms, as well as other Framework tools like the Mass Configuration Tool that allows one to manage large number of devices.

The next Framework version is scheduled to incorporate a Configuration Database tool that will allow the user to save an existing configuration for a device into an external Database (and later restore it into the PVSS system).

The Framework also provides simple operation panels for diagnostic and debugging purposes. As the main operations panels will be experiment specific these should be developed by the sub-detector teams themselves and are therefore not provided by the Framework.

Both configuration and operation panels are integrated in the main Framework tool, the Device Editor and Navigator. This utility displays the devices in an Explorer-like structure that allows a user to manage...
device instances and build simple hierarchies. More complex hierarchies that include FSM behaviour can be created using the Controls Hierarchy tool.

Figure 2: Framework Device Editor and Navigator panel.

Framework panels make wide use of PVSS reference panels to implement common panels for different device types. Both the use of these types of panels and the data-driven configuration model ease the integration of new devices and reduce the development effort.

CONFIGURATION DATABASE

Each experiment will have a central configuration database which will hold the configuration information for the complete experiment and not just that for the DCS. As a consequence of using PVSS for the DCS, the DCS configuration data is stored in the PVSS internal database. To enable this to be stored in, and retrieved from, the experiment’s central configuration database, the Framework is developing an interface to such an external configuration database.

DCS configuration data can be grouped into three categories:

- System Static Configuration data: This is the data that corresponds to the overall control system, such as PVSS project settings and various configuration files.
- Device Static Configuration data: Data directly related to devices, e.g. device structure, instances of a device, address, archive and alarm configuration, etc…
- Device Dynamic Configuration data: data related to devices that might change frequently, such as hardware settings. Alarm configuration could also be considered as dynamic depending on how often a user wanted to change it. A set of dynamic configuration data for a particular device is called a recipe.

PVSS data is stored in an internal database that is accessible through a proprietary API. To move this data to an external database PVSS provides a set of functions based on the ADO standard (ActiveX Data Objects) that can be used to connect to an external ODBC-compliant database. Although this feature was originally only available for Windows, the latest PVSS version, 3.0, provides an implementation that runs on Linux.

Other mechanisms have also been considered for the interface between PVSS and the external database, such as intermediate ASCII or XML files, but they have been discarded due to the difficulty of integration or poor performance.

A prototyping exercise is currently under way using the ADO library as the communication mechanism between PVSS and Oracle and MySQL databases. This first implementation is integrated in the Framework Device Editor and Navigator and will allow a user to store the configuration of existing devices into the database and load it back into the PVSS system. The database model is device oriented and follows the same structure as the Framework device model in PVSS. The configuration data is grouped into the previously mentioned categories and is kept in versions in the database.

This prototype is also intended to validate the feasibility of the approach in terms of performance.

CONCLUSIONS

The Framework has proved to be not only a useful tool but a good environment for the experiments to address common issues. Thanks to continuous feedback from the users, the Framework has been constantly revised and improved. Experience gained with previous versions led to a core redesign, currently being implemented, that will ensure a robust and flexible core for the experiment control systems.

REFERENCES

[1] Prozeß-Visualisierungs- und Steuerungs-System made by ETM AG, Eisenstadt, Austria.

Proceedings of ICALEPCS2003, Gyeongju, Korea