"FRAMEWORKING": A COLLABORATIVE APPROACH TO CONTROL SYSTEMS DEVELOPMENT

M. González-Berges, R. Barillère, F. Bernard
CERN, Geneva, Switzerland

ABSTRACT

The use of frameworks in software engineering is a common practice to ease the development and maintenance phases. In our terminology, a framework is a set of practices and software components from which a developer can select a subset for his application. Three frameworks are currently in use at CERN for the development of control systems: the JCOP (Joint COntrols Project) Framework, the UNICOS (UNified Industrial COntrol System) Framework and the LHC GCS (Gas Control System) Framework. The three projects originate from different domains, with different requirements and timescales. Still there are many commonalities and considerable effort has been invested in establishing and maintaining a collaboration between the three projects. The paper will talk first about the reasons for a framework based approach. The different Frameworks will then be described, with their domain of applicability, their scope and a short overview of the technical details. Afterwards, the underlying tools will be presented. The relations and dependencies between these Frameworks will be explained. Finally a summary of our experience and some conclusions will be given.

INTRODUCTION

In view of the construction of the LHC accelerator and experiments, there was a process of technology standardization in the field of control systems. A direction was taken to have common developments within each of the domains instead of reinventing the wheel for each new application as happened in the past when they were developed independently. The natural way for these common developments has been to have a framework that is then used to build several applications.

In particular, the JCOP Framework was setup for the LHC experiments Detector Control Systems (DCS), the UNICOS Framework initially meant for the cryogenic systems of the LHC accelerator has been extended to other systems, and the LHC GCS Framework for the gas systems in the LHC experiments.

A further step has been the collaboration between these frameworks. In our context, a typical control application is composed of a supervision layer (where actions like monitoring, archiving or trending are performed) and a process control layer (where actions like control loops or sensors reading are performed). Although the collaboration affects both layers, the paper will concentrate on the supervision layer.

REASONS FOR WORKING WITH FRAMEWORKS

The first obvious reason for working with Frameworks is to develop software more rapidly. When many applications have to be developed separately and then integrated in a bigger system, having a Framework becomes crucial. Having many developers using the same tools, makes them think in a similar way and have a common vocabulary. Like that they can work together.

DESCRIPTION OF THE FRAMEWORKS

The JCOP Framework

The selection of a common SCADA tool (PVSS) was a big step in the direction of reducing the effort needed to build the control systems for the LHC experiments. However, it was clear that it would not be sufficient. Each of the four experiments will have between 10 and 20 subdetectors that can be themselves organized in subsystems for which the control systems are developed by teams distributed worldwide. The teams are composed of many short term staff and non control experts. The independently developed projects will have to be integrated to form the Detector Control System (DCS) of each experiment. The JCOP Framework provides a set of guidelines and tools to be used by developers and integrators in the experiments. The deliverables cover the supervision layer of the

DCS, although recommendations to interface with front-end software to have a coherent configuration are provided. The Framework is component based. When building an application, the developer will select the appropriate set of components and add his own code and operational user interfaces. To maximize reuse, the different components are organized in layers as can be seen in Figure 1. The Framework Core contains general purpose components for interfacing with the underlying technologies (e.g. PVSS, Finite State Machines). On top on the Core we have the Framework Devices that give access to the hardware or model other logical objects. Examples of typical hardware use in the experiments are high voltage power supplies, electronics crates or input/output modules. Then we have the Framework Tools used to handle the devices to provide the functionality required by the user (e.g. browsing, configuring, trending). For more details see [2] and [3].

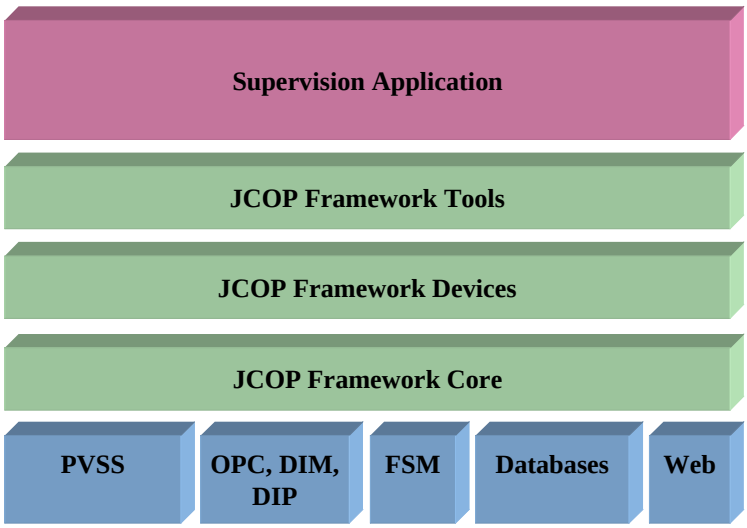

Figure 1. JCOP Framework software architecture.

The UNICOS Framework

The UNICOS Framework was initially developed to control the cryogenics for the LHC accelerator, although currently it is being extended to other projects (for more details see [4], [5] and [6]). It provides a set of tools and devices for the SCADA and the PLC layers (Figure 2) and a methodology to develop a control application. An important point is the automatic generation of PLC software and PVSS devices from a database definition. The Framework "imposes" an application concept, making it very easy for non PVSS experts to develop new applications with the same look and feel for the visualization of the device data. In some cases, the developers are the final operators. In most of the applications it is only required the implementation of user interfaces with a catalog of widgets and some configuration steps, but without any coding. UNICOS is an open framework, new devices and new components can be easily added. The behavior of the UNICOS devices can also be customized to the application needs (e.g. new visualization, access control).

The current users of the Framework are the LHC accelerator cryogenics, the LHC experiments cryogenics, the LHC experiments Gas Control Systems and the supervision of two projects for the accelerator magnets: the Quench Protection System (QPS) and the Powering Interlock Controller (PIC).

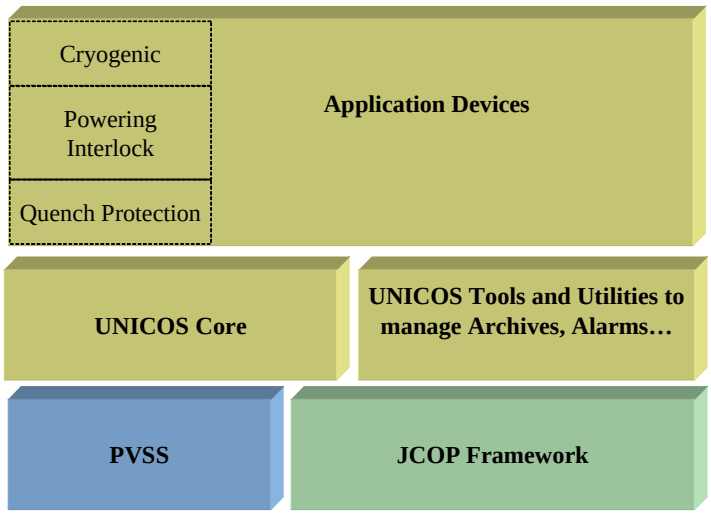

Figure 2. UNICOS Framework software architecture in the supervision layer.

The LHC GCS Framework

The LHC experiment Gas Control System (LHC GCS) [7], [8] aims at providing LHC experiments with homogeneous control systems (supervision and process control layers) for their 23 gas systems. To ease the production of these control systems, it has been decided to develop a library of components, the LHC GCS framework, and to adopt a model-driven automatic code generation approach. In this case the developers of the LHC GCS Framework are as well the developers of the applications. The LHC GCS framework solution consists of: object components to handle the gas specific devices, SCADA components to provide data-driven solutions for the implementation of all functionalities of the supervision layer and PLC components for a data-driven generation of the LHC GCS instance specific process control code. The components are designed in such a way that they can be integrated in a model-based approach [9] which allows the automatic code generation of a complete LHC GCS instance.

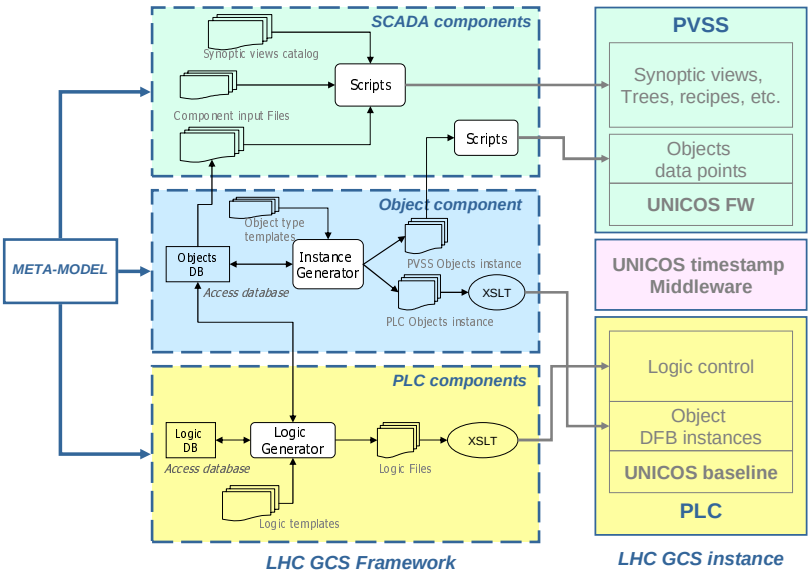

Figure 3. The components of the LHC GCS Framework.

HOW PVSS HELPS

PVSS is a SCADA tool produced by the Austrian company ETM (see [1]). There are some features of the tool that are relevant when talking about collaborative development:

- It is device oriented rather than tag based. As in object oriented programming, one can define a class (datapoint type) that can then be instantiated (datapoints). This makes the reuse of device classes defined by other developers straight forward.
- The tool can be easily customized and extended either in the internal C-like scripting language or directly in C/C++.
- Multiple operating systems are supported. In our case only Windows and Linux systems are important.
- Different configurations can be stored in the internal database, making it easy to customize developments for different projects.
- One can distribute the PVSS processes (e.g. User Interface, Control Manager, Archive Manager) accross several machines. After an initial setup, it is transparent for a developer where the processes are located. With this facility the same software components can run on projects with different hardware architectures.

RELATIONS BETWEEN THE FRAMEWORKS

In the previous points we saw that although the three frameworks come from different domains, there are some commonalities that give an opportunity for collaboration.

As can be seen in Figure 4, we follow a layered approach where each framework reuses parts of the frameworks below it. This is the basic principle, but sometimes a lower level framework reuses from above (e.g. GCS recipe types can be reused in UNICOS) or developers of one contribute to a lower level framework (e.g. extension of address types in JCOP by UNICOS).

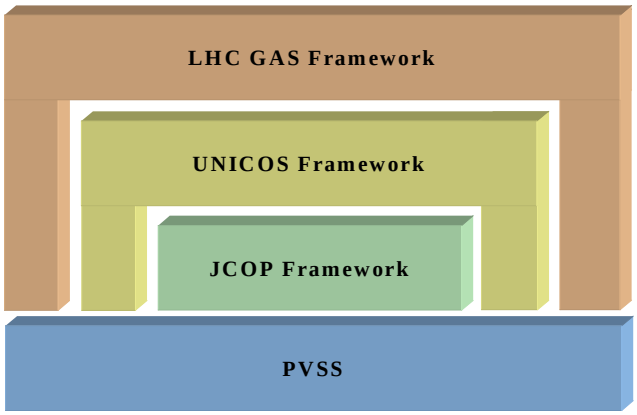

Figure 4. Layered organization of the Frameworks.

In the following paragraphs the different ways in which the frameworks interact are presented, and some examples are given.

Hiding complexity (encapsulation)

This is the case when a library or a group of libraries wrap some functionality, reducing the knowledge required to access that functionality. As an example we can mention the JCOP Framework libraries that make it simpler for developers to use the PVSS configurations associated with datapoints* for addressing, alarms, archiving, etc. Changing a PVSS configuration implies changing the value of several items in the PVSS internal database in the right order. The libraries encapsulate that knowledge and provide a common API.

* In PVSS, a datapoint is a data structure used to group related data items. In object oriented programming terminology, it is the equivalent of an object containing only attributes and no methods.

Using the same concepts (horizontal integration)

We use this term when there is not an explicit reuse of a component but rather the same concept is shared between the frameworks. An example is what we call a device. From the point of view of the SCADA, a device is a combination of datapoints, libraries and a user interface (set of panels for configuration and widgets for operation). The datapoint represents the data related with a piece of equipment (e.g. valve) or some logical entity (e.g. plot). The library contains the logic to manipulate the device. The panels and widgets allow an expert to configure the device or an operator to access it.

Reusing components (vertical integration)

When a component is reused, we talk about vertical integration. In this case, the component can be taken as it is or customized for a particular need. Sometimes when the customization is difficult or impossible, or when the added functionality is of general interest, it is included as an improvement in the component itself. This development of new parts can either be done by the person responsible for the component, or by the person suggesting the new functionality and then integrated and maintained by the person responsible.

The JCOP Framework Installation Tool is used by UNICOS and GCS without modifications. The tool allows transferring PVSS developments packaged as components to other projects. It is based on a set of steps taken automatically for installation and an XML file with the description of all the files in the component. Another example of usage without adaptations is the JCOP Framework Access Control Tool that allows the management of users by giving then access privileges in different parts of the control system.

Some of the UNICOS standard devices are used without modifications in the LHC GCS. In other cases a customization is required by using a set of panels and function hooks (e.g. the LHC GCS access control is different). The LHC GCS team has developed a set of new devices (e.g. xPAR, delayed alarms). These devices were in principle specific for gas applications.

An example where customization was required is the JCOP Framework Trending Tool. The tool extends (e.g. pages of plots, organization in a tree) and simplifies (e.g plot setup) the capabilities of PVSS. The initial version was not enough to cover the needs of UNICOS users. This led to some enhancements (e.g. dynamic addition of curves to a plot, more run time controls, reusable widget to add plots in operation panels) developed by the UNICOS team and later incorporated in the component. On top of that, the UNICOS team has customized the tree view to make it consistent with their look and feel. This has been possible because the internal design follows what was mentioned in the previous point (clear separation of the data model, the logic and the display, see Figure 5).

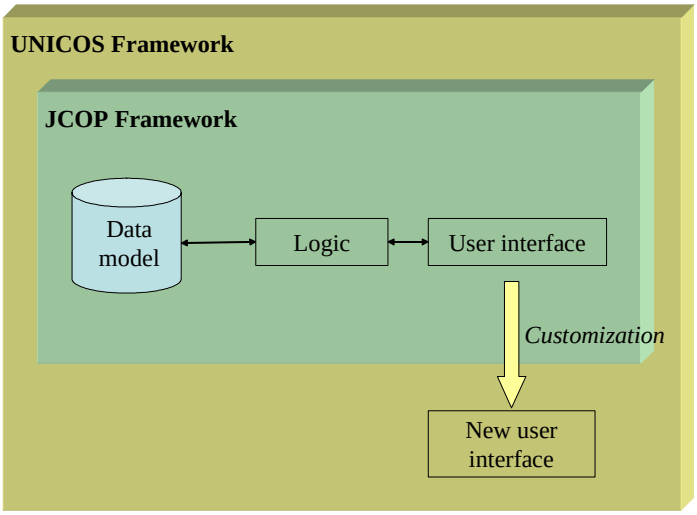

Figure 5. Customization of the Trending tool by UNICOS

Other examples include the usage in the LHC GCS Framework of the UNICOS alert and event screen and the system diagnostics components.

There have been cases where extensions to one of the Framework have been provided by other teams. For example, the LHC GCS team has developed the Device Selector tool that was later included into UNICOS. This tool is meant to find devices in the PVSS database according to some criteria. It can be used to associate devices to user interface widgets, to add curves to a trend plot, etc.

SUMMARY OF OUR EXPERIENCE

Overall we have found that having such collaboration is beneficial for each of the projects, even though there are many obstacles to overcome. On the positive side we have found that not only can we share the development effort, but also a lot of the testing. The Frameworks become more robust as they are tested under different conditions. It is also an important point in the evolution of our software when new versions of the underlying tools appear. Working together has generated enough momentum to create a user community at CERN, where different points of view are exchanged and future enhancements for the Frameworks as well as PVSS can be discussed.

Considering the drawbacks, we saw that sometimes progress seemed slow because of the necessary discussions to reach agreement. The different timescales for each of the projects were also problematic; especially when a team had to develop a component far in advance to the time it was required for their project. However, our view on the long term gain helped in maintaining our collaboration. A supplementary effort in documentation and consultancy was necessary because of the additional external users of the same software. Another point is that it became critical to have a proper process for the release of new versions as well as information dissemination. There has to be some synchronization of the updates, especially when systems in production are involved or new versions are not compatible with old ones. These efforts have shown to be very beneficial also for the standard users of each of the Frameworks.

CONCLUSION

Technology standardization was the first step to reduce the effort in the development of control systems for LHC. Afterwards it was necessary to build specific frameworks for each of the domains. Then more possibilities were opened by reusing parts of these frameworks. Although this has imposed some constraints in each of the projects, we think that there is a clear benefit and we expect that it will even more important in the long term.

REFERENCES

[1] PVSS (Prozeßvisualisierungs- und Steuerungssystem). http://www.pvss.com
[2] The Joint COntrols Project (JCOP). http://cern.ch/itco/Projects-Services/JCOP
[3] The JCOP Framework. ICALEPCS 2005, Geneva, Switzerland. [WE2.1-60]
[4] The UNified Industrial COntrol System (UNICOS) Framework. http://cern.ch/ab-project-unicos
[5] UNICOS. A Framework to build industry-like control systems. Principles and Methodology. P. Gayet, R.Barillère. ICALEPCS 2005, Geneva, Switzerland. [WE2.2-6I]
[6] Deploying the UNICOS Industrial Controls framework in multiple projects and architectures. C.H. Sicard et al. ICALEPCS 2005, Geneva, Switzerland. [WE3A.2-6O]
[7] The LHC Gas Control System (GCS) Framework. http://cern.ch/itcofe/Projects/LHC-GCS
[8] LHC GCS: A framework for the production of 23 homogeneous control systems. G. Thomas et al., ICALEPCS 2005, Geneva, Switzerland.
[9] LHC GCS: A model-driven approach for the automatic PLC and SCADA code generation. G. Thomas et al. ICALEPCS 2005, Geneva, Switzerland. [FR2.3-6O]