Software tools at the Rome CMS/ECAL Regional Center

Giovanni Organtini

Università di Roma “La Sapienza” and INFN - Sez. di Roma

Abstract

The construction of the CMS electromagnetic calorimeter is under way in Rome and at CERN. To this purpose, two Regional Centers were set up in both sites. In Rome, the project was entirely carried out using new software technologies such as object oriented programming, object databases, CORBA programming and web tools. It can be regarded as a Use Case for the evaluation of the benefits of new software technologies in high energy physics. Our experience is positive and encouraging for the future.

1 Introduction

The electromagnetic calorimeter of the CMS[1] experiment is being built in CERN and in Rome. Those two sites are referred in the following as Regional Centers. The calorimeter is composed of about 80000 $PbWO_4$ scintillating crystals, each of which must be carefully characterized before being mounted in the supporting structure.

The ECAL CMS collaboration has developed suitable systems to make the whole process as automatic as possible, reducing the manipulation of parts and related data by Centers operators.

In Rome an automatic machine (ACCOR [2]) was built to fully characterize crystals in terms of size, transmission and light yield uniformity. We took this opportunity to test, on a limited scale, but on a real project, the possible benefits of the new software technologies that are becoming popular in high energy physics. In fact the realization of the Regional Center for the production of the CMS ECAL is the first real LHC project in Italy realized using entirely new software technologies.

2 Object Oriented Design

ACCOR is built in a modular way: a crystal server provides movements of the parts and positioning of them in front of the instruments. Three devices actually do the measurements, namely, a 3D machine for mechanical measurements, a compact spectrophotometer for transmission and a photomultiplier tube with its DAQ chain for light yield determination. Each part is controlled by a separate piece of software in order to maintain the same modularity of the hardware. Object Oriented technology is was used in this work since it is perfectly adapted to model modular pieces of hardware.

Both the crystal server and the instruments have suitable classes to represent them in the measurements control programs. The server is represented by a `Motor` class whose design follows the Singleton [3] pattern to ensure that only one instance is running at any time. This class provides methods to manipulate parts inside the machine.

Each instrument along the measuring chain is represented essentially in the same way thanks to the fact that they were designed to be controlled by a Workflow Management System developed by the CRISTAL collaboration[4]. In this framework instruments are defined as machines able to send characters over a TCP/IP network, connected to CRISTAL via a socket connection with the so called CRISTAL Instrument Agent.

Instruments requests jobs to be done on crystals (parts) sending strings to CRISTAL via network and receive commands using the same protocol. The Instrument Agent computes the jobs to be executed from the status of the part for which they were requested according to its workflow stored in the CRISTAL Data Base (based on Objectivity^{TM}). Both commands and outcomes exchanged between Instrument Agent and Instruments are encoded in a way similar to XML in order to be able to exchange data virtually with any kind of Instrument, while the complex dialog between the CRISTAL components, based on CORBA, is left to the Instrument Agent.

Instruments and parts are represented, in our software, by C++ classes: an `Instrument` class and a `PartOnInstrument` class. The first encapsulates the functionalities for connecting to CRISTAL and registering parts to be measured. Such parts are modeled by the `PartOnInstrument` class who provides methods for requesting jobs to CRISTAL, decoding them and sending outcomes. Both classes inherit from the `XMLParser` class, a class to decode any possible combination of XML-like elements foreseen for CRISTAL-Instrument communication. This design was identified to belong to the well known Composite pattern.

During the project development we found that a careful design at the beginning of the project is of capital importance for future maintainance, even if it has not been carried using formal tools and methods.

However, we also found, after a subsequent deeper analysis made using formal methods and patterns, that a further optimization of the model can still be achieved as well as a further simplification of program structures. To this purpose commercially available design tools are found to be very useful.

In our class definitions we tried to avoid as much as possible the use of pointers. When resizable containers are needed Standard Template Libraries (STL) are used, instead. Actually no explicitly declared pointers appear in our programs.

In Real Time applications performances are quite important. We compared the performances of our C++ programs for data acquisition with the ones of our old *traditional* programs written for the same objects in the past. At least we did not found any significant difference in performances. In few cases C++ provides tools such that some operation can be even faster. Only for few special needs we had to provide *shortcuts* to make the execution of methods faster.

3 Operating Systems integration

In the Rome Regional Center there are both computers running Windows NT and Linux. The CRISTAL system runs on Windows NT, while instrument programs run on Linux computers. Linux was chosen for instruments in Rome to provide remote access, full control and multi-user and multi-tasking capabilities. Only one of the instruments (the 3D machine for dimensional measurements) runs its software (written by the company who produced the machine) on Windows NT.

The CRISTAL software is quite independent and requires interaction only with instruments: the solution is provided by the Instrument Agent so, in fact, no integration problem exists for it.

The 3D machine, instead, can only be operated using a windows interface. In order to integrate it in the system its control program gets input and puts output on text files belonging to Linux machines, exported using Samba [5]. CRISTAL-Instrument dialog is provided by a C++ program running on Linux that uses the `Instrument` and `PartOnInstrument` classes described above, then writes and reads those files.

The 3D machine being mounted along the crystal server, is inside a dark room and is not accessible during the measurement. A simple solution to control the machine from outside was found in VNC [6], a freeware product initially developed by an Olivetti-Oracle consortium and now being maintained by AT&T. VNC allows to view a Windows Desktop, exported via the network, inside an X-window on a Unix computer.

4 Web Tools

The ACCOR machine can be operated both using CRISTAL and as a standalone machine. It always operates in the same way, but in the standalone mode the Instrument Agent simply gets commands from dedicated input text files. The only change required in the system in order to run the machine in standalone mode is the setting of an environment variable holding the computer name running the Instrument Agent. In order to simplify the user interaction nearly all the operations that can be done on the machine can be accessed via the web.

The simplest job for a web server is to provide documents. In particular the hardware documentation of the Center is entirely available via Web.

Web tools are also used, in the Rome Regional Center, to control the machine. Using forms, authorized users, can interact with the system providing information about the measurements to be done, i.e., the parts to be measured, the type of measurements and the parameters to do them, instead of manually editing configuration files.

Data provided in the form are then collected by Perl [7] scripts that automatically generates them for the machine operation, according to the user requests.

Another important benefit of this approach is that the ACCOR machine can be operated even remotely, without the need of operators near to it, but when the machine has to be loaded with parts.

Off-line monitoring is also available via web: all the data recently collected by the machine are stored on disks accessible by the web server computer. The main access to Regional Center data is CRISTAL, but during measurements it is useful to be able to make fast analysis on recently collected data. Enough space is provided in order to store data coming from the measurement of about 300 parts. Perl scripts looks for available data inside the default directories and dynamically generates lists of hyper-links for each part for which a measurement has been done. Clicking on them other scripts are run who read those data and provides plots, histograms and combined numerical data to the user via the browser window. No plot is stored on disks, saving disk space, and no maintenance is required to the Web master to the pages providing this kind of data since they are generated on the fly using the scripts cited above.

Many other variables concerning both the instruments and computers status, as well as environmental data, are kept under control using monitoring tools who provide information to be displayed using a web browser.

4.0.1 Functional decomposition

In order to be able to be operated by the Web, instrument control programs have no graphical interface. When needed, output is given in ASCII form, regardless of its type. Rather than being a disadvantage, such a choice has been proven to be very effective. Program structure is simpler and several programs can interact between them getting inputs from character pipes created between them, resulting in an improved collaboration between the various components. Thanks to this *functional decomposition* operations can be splitted in sub-programs who can collaborate in a versatile way, reducing both the number of them and their complexity, since each *functional component* is just responsible for part of the global task to be executed. Whenever needed graphical

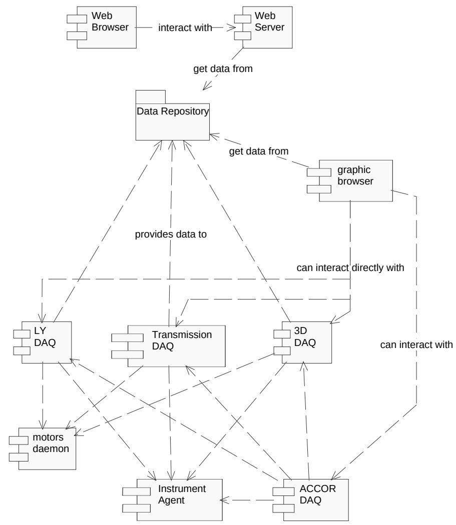

Figure 1: Relationships between software components in the Rome Regional Center. Where not shown, the name of the relationship, must be read as "uses".

interface is provided either by a Web browser or by Perl scripts that starts other graphic browsers. As the diagram in fig. 1 shows, the relationships between the various functional components are quite a lot. Nevertheless, the system is extremely flexible, while, at the same time, each component remains quite simple in its structure.

5 Conclusions

5.1 General Remarks

No more doubts should exist, in high energy physics, that Object Oriented Programming is the best solution for large projects. In this work we experienced the benefits of this technology even at a smaller scale. Both the development and the maintenance of a medium size experiment can be greatly improved using OO programming.

A very practical design strategy we have found is the splitting of a project into smaller independent pieces collaborating between them. We call this approach the functional decomposition of the project.

Explicit use of pointers arithmetics is discouraged: it weaken the robustness of the software and is frequently against the OO approach, to our opinion.

5.2 On the use of Commercial Products

A lot of discussion has been made in the recent past about the use of the so called commercial products. The current strategy is to push towards the use of them as much as possible. There are several advantages in this strategy:

- timescales for the development of projects are greatly reduced;
- documentation is usually of high quality;
- the use of the resources are optimized, both in terms of hardware and of personnel;
- support is guaranteed on large time scales, the project being commercially supported and not depending on people's scientific interests.

Nevertheless care must be taken in choosing a commercial product for physics needs, taking care of the following points:

- commercially available software is tuned for the needs of the largest possible part of potential users: with this respect physicists usually represent a very small part of the possible market, so companies have no big interests in providing specialized software tools;
- the support from companies is not always such as expected;
- the risk to become company-dependent is high: any software company will try to sell its own product, so the efforts of consortia aiming to standardization, actually are non completely effective.

All of these pros and cons have been experienced during our work.

In particular, using Objectivity we experienced some problems in the integration between our software and the Objectivity Data Base as well as some failures of its optional components for data replication. Even if these problems are expected to disappear with the next versions of the product, the lack of knowledge due to industrial secrets makes the understaing of the problems and the identification of causes very difficult.

CORBA implementations, however, are becoming to be available for several platforms, even for Linux. Commercially available, open source software, is quite attractive to solve the kind of problems outlined above. We plan to test few of them to evaluate their usefulness in physics projects and robustness.

References

1 The Compact Muon Solenoid, Technical Proposal, CERN/LHCC 94-38
CMS ECAL Technical Design Report, CERN/LHCC 97-33

2 Dafinei, I et al.," Measuring techniques for large scale production of PWO scintillators", Proc. of Int. Conf. on Inorg. Scintillators and their Appl., SCINT99 (1999) Moscow, Russia.

3 Gamma, Helm, Johnson, Vlissides,"Design Patterns: Elements of Reusable Object-Oriented Software", Addison-Wesley.

4 J-M Le Goff et al., "C. R. I. S. T. A. L./ Concurrent Repository & Information System for Tracking Assembly and production Lifecycles - A data capture and production management tool for the assembly and construction of the CMS ECAL detector "
From ftp://cmsdoc.cern.ch/documents/96/note96_003

5 To find information about Samba see http://www.samba.org.

6 See the home page of the project: http://www.uk.research.att.com/vnc/index.html

7 L. Wall, T. Christiansen and R.L. Schwartz, "Programming Perl", O'Reilly