Homogenous Supervisor and Control Software Infrastructure for the CMS Experiment at SLHC

Ildefons Magrans de Abril, Claudia-Elisabeth Wulz

Institute for High Energy Physics, Austrian Academy of Sciences, Nikolsdorfergasse 18, A-1050 Vienna, Austria
Marc Magrans de Abril
CERN, Geneva, Switzerland

Abstract

A project proposal to homogenize the supervisory control, data acquisition, and control software infrastructure for an upgraded CMS experiment at the SLHC is presented. Its advantage is a unique, modular development platform enabling an efficient use of manpower and resources.
1 Introduction

This project aims to develop the CMS Experiment Control System (ECS) based on a new supervisory and control software framework. We propose a homogeneous technologic solution for the CMS infrastructure of supervisory control, data acquisition, and automatic control (SCADA [1]). The current CMS software control system consists of the Run Control and Monitoring System (RCMS), the Detector Control System (DCS), the Trigger Supervisor System (TS), and the Tracker, ECAL, HCAL, DT and RPC sub-detector supervisory systems. This infrastructure is based on three major supervisor and control software frameworks: JCOP [2], RCMS [3] and TS [4]. In addition, each subdetector has created its own SCADA software.

A single SCADA software framework used by all CMS subsystems would have advantages concerning the maintenance, support and operation tasks during the experiment life-cycle:

a) Overall design strategy optimization: There is an evident similarity in technical requirements for controls amongst the different levels of the experiment control system. A common SCADA framework will allow an overall optimization of requirements, design and implementation.

b) Support and maintenance resources: The project should enable an efficient use of resources. A common SCADA infrastructure for CMS will manage the increasing complexity of the experiment control and reduce the effects of current and future constraints on manpower.

c) Accelerated learning curve: Operators and developers will benefit from a common SCADA infrastructure due to: 1) One-time learning cost, 2) Moving between CMS control levels and subsystems will not imply a change in the technology.

This project is based on the evolution of the software infrastructure used to integrate the Level-1 Trigger subsystems. Section 2 presents the project technology baseline and the criteria for its selection. Section 3 presents an overview of the project road map. Finally, section 4 outlines the project schedule and the required human resources.

2 Technology baseline

The design and development of the unique underlying supervisory and control infrastructure should initially start from the software framework currently used to implement the Level-1 Trigger (L1T) control software system. This framework is called Trigger Supervisory Framework (TSF). The following paragraphs describe the principal objective criteria for which this technological baseline has been chosen:

a) Proven technology. It is used in the implementation of a supervisory and control system that coordinates the operation of all L1T subsystems, the TTC system, the Luminosity Monitoring System and to some extent the ECAL, HCAL, DT and RPC subdetectors. This solution was successfully used during the second phase of the Magnet Test and Cosmic Challenge and is being used in the monthly commissioning exercises of the CMS Global Runs.

b) Homogeneous TrIDAS infrastructure and support. The TSF is based on XDAQ [5], which is the same middleware used by the DAQ event builder. This component is a key part of the DAQ system and as such it is not likely to evolve towards a different underlying middleware. Therefore, a supervisory and control software framework based on the XDAQ middleware could profit from a long term, in-house supported solution. In addition, a SCADA infrastructure based on the XDAQ middleware would homogenize the underlying technologies for the DAQ and for the supervisory control infrastructure that would automatically reduce the overall support and maintenance effort.

c) Simplified coordination and support tasks. The TSF is designed to reduce the gap between software experts and experimental physicists and to reduce the learning curve. Examples are the usage of well known models in HEP control systems like finite state machines [6][7] or homogenous integration methodologies independent of the concrete subsystem Online SoftWare Infrastructure (OSWI) and hardware set-up, or the automatic creation of graphical user interfaces. The latter is a development methodology characterized by a modular upgrading process and one single visible software framework.

d) C++. The OSWI of all subsystems is mainly formed by libraries written in C++ running on x86/Linux platforms. These are intended to hide hardware complexity to software experts. Therefore, a SCADA infrastructure based on C++, like the TSF, would simplify the complexity of the integration architecture.
3 Road map

This project aims to reach the technologic homogenization of the CMS Experiment Control System following a progressive and non-disruptive strategy. This shall allow a gradual and smooth transition from the current SCADA infrastructure to the proposed one. An adequate approach could have the following project tasks:

a) Level-1 Trigger incremental development. Continue with the current development and maintenance process in the LIT using the proposed framework.

b) Subdetector control and supervisory software integration. This task involves the incremental adoption of a common software framework for all subdetectors in order to homogenize the control and supervisory software of CMS. The participating subdetectors are ECAL, HCAL, DT, CSC, RPC, and Tracker. Currently, this step is partially achieved because all subdetectors except the Tracker are partially integrated with the TS system in order to: 1) Automate the pattern tests between the sub-detector trigger primitive generators and the regional trigger systems, 2) Check configuration consistency between LIT and the trigger primitive generators.

c) Level-1 Trigger Emulators supervisory system. This task involves the upgrade of the supervisory software of the LIT emulators to the proposed common framework. The hardware emulators of the LIT have been deployed as components of the CMSSW framework [8]. This task does not involve any change in the emulator code or in the CMSSW framework.

d) High Level Trigger supervisory system. This task involves the upgrade of the supervisory software of the HLT to the proposed common framework. In this way the components of the HLT (filter units, slice supervisors, and storage managers) will be launched, configured and monitored as the other software components of the CMS online software [9]. This task does not involve any change on the supervised components.

e) Event Builder supervisory system. This task involves the deployment of the Event Builder supervisory system as nodes of the proposed framework. The Event Builder supervisory software will launch all software components, will configure and will monitor the Front-End Readout Links (FRL), the Front-End Driver Network (FED Builder Network), and the different slices of Event Managers (EVM), Builder Units (BU) and Readout Units (RU) [10]. This task does not involve the modification of the Event Builder components.

f) Experiment Control System feasibility study and final homogenization step. This is the last stage of the homogenization process. This task involves the feasibility study to change the top layer of the ECS and, afterwards, its substitution by components of the proposed framework. This means the substitution of the Function Managers by the nodes of the proposed SCADA software. This task also involves the feasibility study and homogenization of the top software layer of the DCS in order to be supervised, controlled and monitored by the ECS [11].

4 Schedule and resource estimates

Schedule and resource estimates have been approximated according to the COCOMO II model [12] assuming the delivery of 50000 new SLOC\(^1\), the modification of 10000 SLOC and reusing 30000 SLOC, with the model parameters rated as a project with an average complexity. The SLOC effort has been estimated using the development experience with the Trigger Supervisor and Run Control and Monitoring frameworks. Additional assumptions are a development team of people working in an in-house environment with extensive experience with related systems, and having a thorough understanding of how the system under development will contribute to the objectives of CMS.

The four project phases are: 1) Inception: This phase includes the analysis of requirements, system definitions, specification and prototyping of user interfaces, cost estimation; 2) Elaboration: This period is meant to define the software architecture and test plan; 3) Construction: this includes the coding and testing phases 4) Transition: this last phase includes the final release delivery and setup of support and maintenance infrastructure.

\(^1\) SLOC: Source Lines Of Code. Only Source lines that are delivered as part of the product are included -- test drivers and other support software is excluded. Source lines are created by the project staff -- code created by application generators is excluded. One SLOC is one logical line of code. Declarations are counted as SLOC. Comments are not counted as SLOC.
Table 4.1 shows the schedule for the project phases and the required resources per phase in person-moths. This estimate includes the resources to deliver the infrastructure stated in section 3: all templates, standard elements and functions required to achieve a homogenous system and to reduce as much as possible the development effort for the subsystem integration developers. This estimate does not include the subsystem integration, which follows the Transition phase.

<table>
<thead>
<tr>
<th>Phase</th>
<th>Phase effort (Person-months)</th>
<th>Schedule (Months)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception</td>
<td>16</td>
<td>3</td>
</tr>
<tr>
<td>Elaboration</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td>Construction</td>
<td>199</td>
<td>14</td>
</tr>
<tr>
<td>Transition</td>
<td>32</td>
<td>13</td>
</tr>
</tbody>
</table>

Table 4.1: Project phases schedule and associated effort in person-month

We summarize in Table 4.2 the top-level resource and schedule estimate of the project.

<table>
<thead>
<tr>
<th>Total effort (Person-months)</th>
<th>311</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schedule (months)</td>
<td>38</td>
</tr>
</tbody>
</table>

Table 4.2: Top-level Estimate for Elaboration and Construction

The project is open to interested CMS collaborators. It could also include developers outside CMS such as from the CERN IT division or from Associated Institutes.

References