Communication between Trigger/DAQ and DCS in ATLAS

H.Burckhart[1], R.Hart[2], R.Jones[1], V.Khomoutnikov[3], Y.Ryabov[3]

1) CERN, Geneva, Switzerland
2) NIKHEF, Amsterdam, Netherlands
3) PNPI, Gatchina, St.Petersburg, Russia

Abstract

Within the ATLAS experiment Trigger/DAQ and DCS are both logically and physically separated. Nevertheless there is a need to communicate. The initial problem definition and analysis suggested three subsystems the Trigger/DAQ DCS Communication (DDC) project should support the ability to:

1. exchange data between Trigger/DAQ and DCS;
2. send alarm messages from DCS to Trigger/DAQ;
3. issue commands to DCS from Trigger/DAQ.

Each subsystem is developed and implemented independently using a common software infrastructure. Among the various subsystems of the ATLAS Trigger/DAQ the Online is responsible for the control and configuration. It is the glue connecting the different systems such as data flow, level 1 and high-level triggers. The DDC uses the various Online components as an interface point on the Trigger/DAQ side with the PVSS II SCADA system on the DCS side and addresses issues such as partitioning, time stamps, event numbers, hierarchy, authorization and security. PVSS II is a commercial product chosen by CERN to be the SCADA system for all LHC experiments. Its API provides full access to its database, which is sufficient to implement the 3 subsystems of the DDC software. The DDC project adopted the Online Software Process, which recommends a basic software life-cycle: problem statement, analysis, design, implementation and testing. Each phase results in a corresponding document or in the case of the implementation and testing, a piece of code. Inspection and review take a major role in the Online software process. The DDC documents have been inspected to detect flaws and resulted in a improved quality. A first prototype of the DDC is ready and foreseen to be used at the test-beam during summer 2001.

Keywords: ATLAS, Trigger/DAQ, DCS, Online, SCADA, PVSS II, Software process.

1. Introduction

The fact that Trigger/DAQ (shortened as DAQ) and DCS of ATLAS are physically and logically separated, does not mean they do not have to communicate. There has to be some kind of communication between the two systems. Especially during physics data taking (but also during calibration) the run control of the DAQ should be able to issue commands to DCS, being informed on errors and warnings and the state of the (sub)detector(s) in general. A general utility to exchange data, like states and parameters, permits the DCS to get the state of the DAQ. The DAQ-DCS Communication software (DDC) is dedicated to support the communication. For the design and implementation the DDC software has to deal with a couple of prerequisites and boundaries:

- Any manipulation with the physics data is beyond the scope of the DDC.
- *Partitioning*: the DAQ and DCS of ATLAS use a different concept of partitioning. The DDC software is aware of it, but is not responsible in case the partitions do not match in terms of boundaries and resource allocation.
- The DCS is expected to be operational at all times, but the DDC is available as soon as DAQ becomes operational.

1.1 Software Process

The DDC project adopted the Online Software Process with its software life-cycle, containing phases like Analysis, Design, Implementation, Testing, etc. For each phase a document or code is produced. At the moment there is a User Requirement Document (URD), a High Level Design (HLD), a Test Plan and a draft User Guide. An integral part of the Online Software Process is *inspection.* After each phase in the software life-cycle the document/code should be inspected or reviewed. Currently the URD and HLD were inspected and the Test-plan reviewed. Inspection is greatly appreciated and some of the benefits are: detecting faults as soon as possible, sharing knowledge and getting to know the other developers.

2. Context of the DDC

The DDC software behaves like an interface between DAQ and DCS and has been mentioned in the HLT/DAQ/DCS Technical Proposal [1]. The DCS uses a commercial SCADA (Supervisory Control And Data Acquisition) system, called PVSS II [2]. The ATLAS DAQ system consists of several subsystems: Level 1, Event Filter, Data Flow, etc, including the Online software which encompasses the software needed to configure, control and monitor the DAQ. The DDC uses the PVSS II and Online software for its interface points with the DCS and DAQ respectively. Figure 1 shows the basic context diagram of the DDC software.

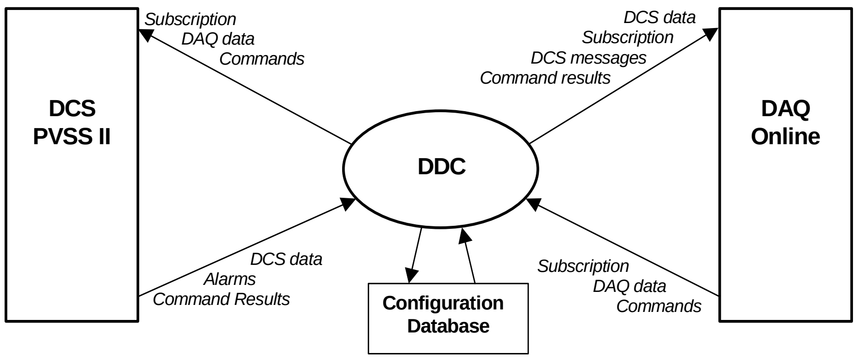

Figure 1. DDC context diagram.

The term *subscription* in figure 1 is used to denote asynchronous reading. The caller gets the initial value and afterwards it will be notified on all updates. The DDC uses its own configuration database, containing the information it needs for its functionality.

2.1 PVSS II

PVSS II is a commercial SCADA system manufactured by the Austrian company ETM [2]. It is chosen by CERN to be the SCADA system for all LHC experiments. Besides the typical SCADA functionality, like data acquisition, logging and archiving, alarm handling, access control mechanism and a man machine interface, PVSS II has the following additional capabilities: it can run in a highly distributed manner, it has multi-platform support (WNT and Linux), it is device oriented and it has an advanced scripting facility. PVSS II contains an internal 'real-time' database, based on the object oriented *datapoint* concept. The datapoint is the basic data container of a variable and could be everything from being a simple type (integer, float, etc) or a complex type like an array or a reference to another datapoint. The API of PVSS II consists of a C++ class library and provides an interface to the external world, to access the datapoints. Furthermore, PVSS II is capable of running scripts, which are triggered by an update of a datapoint. They are interpreted and have a C/C++ like syntax.

2.2 Online Software

The Online subsystem can be regarded as the supervisor of the DAQ. It is essentially the 'glue' that holds the various subsystems together. The Online runs on various platforms running some kind of Unix flavour, including Linux. At the moment WNT is not supported, nor will it be in the near future. C++ is used as the primary programming language and the internal communication is based on CORBA (ILU). The Online software consists of a group of inter-operating components, each supporting a specific function.

3. DDC Subsystems

During the analysis phase of the DDC project the domain decomposition defined 3 functions the DDC should provide:

1. Bi-directional exchange of data like parameters and status values;
2. Transmission of DCS messages, like alarms, to DAQ;
3. Ability for DAQ to issue commands to DCS.

The functions listed above are independent and are implemented as separate subsystems. For each subsystem an Online component is involved. It is expected that on each PVSS II system (a set of PC's forming a logical system) at least one runs Linux. This machine acts as a bridge between DAQ and DCS, because both the Online software, as well as PVSS II (with its API), is available on Linux, which enables it to implement the 3 DDC subsystems.

3.1 DDC Data Transfer

The subsystem to provide bi-directional exchange of data, both read and write, is called DDC Data Transfer (DDC-DT). The term data is regarded as a value belonging to a DAQ or DCS variable. The DCS variables reside in the SCADA database. On the Online side the *Information Service* (IS) is used, because it is logically the counterpart of the SCADA database. The configuration database for the DDC-DT subsystem contains two lists. The first contains a list of variables maintained by the PVSS II system and put at the disposal of any DAQ application by means of the IS. The second contains a list of variables, maintained by DAQ applications, which by means of the IS and DDC-DT, is exported to the SCADA's database. The two lists express the two use cases the DDC-DT supports: data transfer from DAQ to DCS and vice versa. Both cases are almost symmetrical, but the main difference is that DCS is supposed to be available at all times, while the DAQ applications are not. On both sides the variables are subscribed, in order to capture any change or update. An overview of the collaboration between the subsystems participating in the DDC-DT for the two cases is shown in figure 2a and 2b.

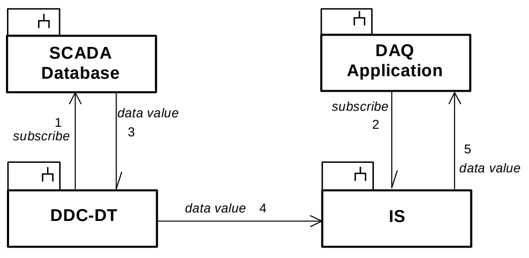

Fig 2a. DCS to DAQ data transfer

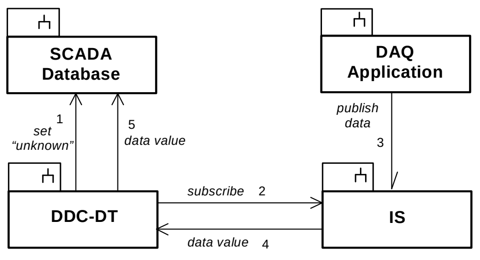

Fig 2b. DAQ to DCS data transfer

3.2 DDC Message Transport

The message transfer system of the DDC is known as DDC-MT and it is responsible to transport alarm messages from DCS to DAQ. A message is regarded as a piece of information packed into a string. The *Message Reporting System* (MRS) of the Online software is responsible to make messages available on the DAQ side. The alarms reside as datapoints in the SCADA database and are subscribed by the DDC-MT, where each change/update is caught and a message is formatted according to the MRS convention and then sent to the MRS. The configuration database of the DDC-MT contains a list of alarms. A typical collaboration diagram is almost identical to figure 2a.

3.3 DDC Command Transfer

The DDC Command Transfer subsystem (DDC-CT) provides the ability for DAQ to issue commands to DCS. The *Run Control* (RC) component of the Online software is used as the interface point on the DAQ side, what implies that a command is regarded as a controller transition. Examples of such transitions are: *load*, *configure* and *start*. Only dedicated DAQ applications, called *controllers*, are enabled to use the DDC-CT. The commands have to be implemented on the DCS side and are beyond the responsibility of the DDC. The commands are stored as scripts and triggered by setting a dedicated datapoint. The result of the command is written into another datapoint. The configuration database of the DDC-CT subsystem contains a list of commands, together with the trigger-datapoint and result-datapoint. Furthermore it contains a timeout value, indicating the amount of time within a command has to finish. A typical collaboration diagram of the DDC-CT is shown in figure 3.

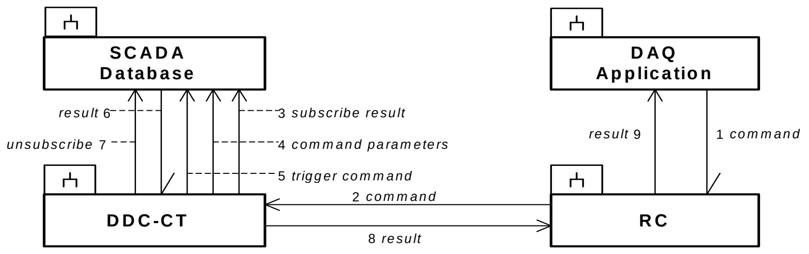

Figure 3. DDC-CT Collaboration Diagram

4. Status and Acknowledgement

Release 0.2 of the DDC software is available and supports the DDC-DT and DDC-CT subsystem. The DDC-MT is expected in the next release. The DDC is now part of the Online software as a full component and available in release 0.0.14. Release 0.2 will be used at the Tile Calorimeter test-beam at CERN during summer 2001. All documentation is available on the Online homepage [3] or DCS homepage [4]. The authors would like to thank all those people who participated in the DDC reviews, the TDAQ and DCS community in general, as well as anyone who has provided valuable feedback, in particular Doris Burckhart, Mihai Caprini, Jim Cook, Serguei Kolos, Serguei Malioukov, Igor Soloviev and Fernando Varela Rodriguez.

5. References

[1] ATLAS high level Triggers, DAQ and DCS Technical Proposal, CERN/LHCC/2000-17, http:://atlasinfo.cern.ch/Atlas/GROUPS/DAQTRIG/SG/TP/tp_doc.html
[2] PVSS II von ETM, http://www.pvss.com
[3] Atlas Online Homepage, http://atddoc.cern.ch/Atlas/DaqSoft/Welcome.hml
[4] Atlas DCS Homepage, http://atlasinfo.cern.ch/ATLAS/GROUPS/DAQTRIG/DCS/dcshome.html