ANTARES SLOW CONTROL STATUS

J.M. Gallone[1] on behalf of the ANTARES Collaboration
[1]*Institut Pluridisciplinaire Hubert Curien, CNRS/IN2P3*
23, rue du Loess BP 28, F-67037 Strasbourg Cedex 02
Jean-Michel.Gallone@IReS.in2p3.fr

Abstract

ANTARES is a neutrino telescope project based on strings of Cerenkov detectors in deep sea. These detectors are spread over a volume of 1 km3 at a depth of about 2 km in the Mediterranean Sea near Toulon. About 400 of such detectors are now operational, as well as a large variety of instruments that need a reliable and accurate embedded slow control system. Based on Commodity Off-the-Shelf (COTS) low-power integrated processors and industry standards such as Ethernet and Modbus [1], the foreseen system is expected to run for 3 years without any direct access to the hardware. We present the system architecture and some performance figures. The slow control system stores the state of the system at any time in a database. This state may be analyzed by technical staff in charge of the maintenance, physicists to check the setup of the experiment or the data acquisition system for saving experimental conditions. To give a record of the state of the whole system, to set the value of a parameter of a physical device, to modify the value of a parameter of a physical device and to set up the initial values of the physical devices are the main functions of the slow control system.

INTRODUCTION

The ANTARES telescope is designed to serve as a cosmic neutrino detector for astroparticle research [2]. The idea is to detect Cerenkov light emitted by the relativistic muon that is created through the interaction of a high-energy neutrino with matter and reconstruct the energy and the trajectory of the neutrino. Detector must be large enough to be able to reconstruct trajectories with sub-degree accuracy. As other neutrino experiments, the ANTARES detector must reside in a transparent material medium which refraction index allows for the emission of Cerenkov light, namely solid or liquid water. At the same time, in order for the medium to shield the system from atmospheric muons, the ANTARES detector resides at a depth of 2000 meters on the bottom of the Mediterranean Sea. The detection of the Cerenkov light is achieved by the acquisition system (DAQ) [3] in charge of reading out the digitized data and transmitting them to a computing farm on the shore through fibres inside a submarine cable. All the physical parameters are controlled by the slow control system (SC). SC is to be able to provide the state of the system at each time. This state may be requested by technical staff in charge of the maintenance, physicists to check the set-up of the experiment or the data acquisition system (DAQ) for saving experimental conditions. SC also allows users to send commands to the on shore and off shore physical devices.

The offshore system presents performance challenges together with specific constraints due to its located 2000 m bellow the sea level.

General Layout section is devoted to the description of the main requirements and constraints of the S.C. and a brief description of all the modules that have been implemented. The following section describes the Data Monitoring system; it deals with the route used by data to read the value of parameters. Last section deals with Data Polling list generation, i.e. how the system ensures that all electronic devices are scanned.

GENERAL LAYOUT

As most of the bandwidth on the submarine cable must be dedicated to the data acquisition system, slow control data traffic has been optimised. Therefore, the embedded processors must not be overloaded by the external requests. Thus the SC stores an offshore record of all parameters in a database (DB), and all the external requests will be sent to this database.

So the SC relies on a DB that contains all the parameters with their current value and a history of their values taken during the experiment. It also contains the set-up parameter of each physical device. This DB is accessed by SC monitors, by the DAQ, by users via graphical user interface (GUI), or by sending direct requests to the base via the interface provided.

The SC is interfaced with the Run Control (RC) System used by shifter in Control Room to setup, start and stop the acquisition on the telescope. The SC is in fact a client of the RC, it shares the same internal state machine (figure 1), and transition are sent from RC to SC to ensure synchronisation of both systems.

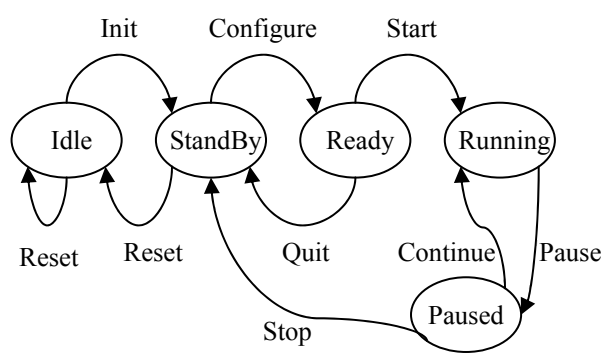

Figure 1: Finite state machine.

The SC takes into account the request from the users, the DAQ, GUI, or DataPolling (an automatic control program that scan the DB). All the results of the command are checked by looking at the resulting modifications in the DB. This check can be done by graphical monitoring system that displays windows on computer screen.

The DB is accessed in write mode by the embedded system via the DBWriter module, which is a simple process that just listens to the Dispatcher module and writes data into a RawData table.

The dispatcher is a module that implements ControlHost [4] message passing interface. It acts as a message server: it receives some messages from processes; each message is identified with a tag. Then, via a subscription mechanism, each process may request messages be sent.

Off Shore, each electronic device is connected via a serial link to an embedded DAQ/SC Board, which runs a ControlHost client to get messages. Then each message is parsed to extract the Modbus request and the link to be used, in order to set up an electronic device or to read a sensor value. Once Modbus response arrived, it is sent to DBWriter via the Dispatcher, and then written as RawData in database.

Another process, attach to the database, is in charge of parsing RawData, that is to decode it, depending on the electronic device that generate the RawData and write it as a set of monitored data, i.e. simple values with a reference to the parameter that has been monitored.

DATA MONITORING

To monitor a specific electronic device, people have to fill a Modbus request table. In this table, each type of device that can appear in the detector is described by a given set of Modbus requests, associated with a frequency and eventually an order when needed. This request is the exact sequence of bytes that is to be sent on the serial link for monitoring the device. Once the Modbus request table is filled, depending on the setup, a view is automatically generated in the database to get the complete set of Modbus request to be sent to get the detector status.

This set is given has input of the DataPolling module, which acts has a scheduler to regularly send request message to the Dispatcher, with a tag that identifies the DAQ/SC board used.

The Dispatcher then forwards the message to the DAQ/SC board. This board is a processor board, running VxWorks [5] where a task named ScHarness is fetching ControlHost messages to extract the Modbus message. The Modbus message is then sent to the electronic device via the serial link indicated in the ControlHost message. When the electronic device replies to the Modbus request, the response is timestamped by the SCHarness task and sent to the Dispatcher.

The DBWriter fetches messages containing Modbus replies and writes them into the database in a RawData table. Then the parser process reads the table, decodes the parameter values encapsulated in the Modbus reply and fill a monitored data table. The monitored data table is then scanned by a SC monitor to check detector status.

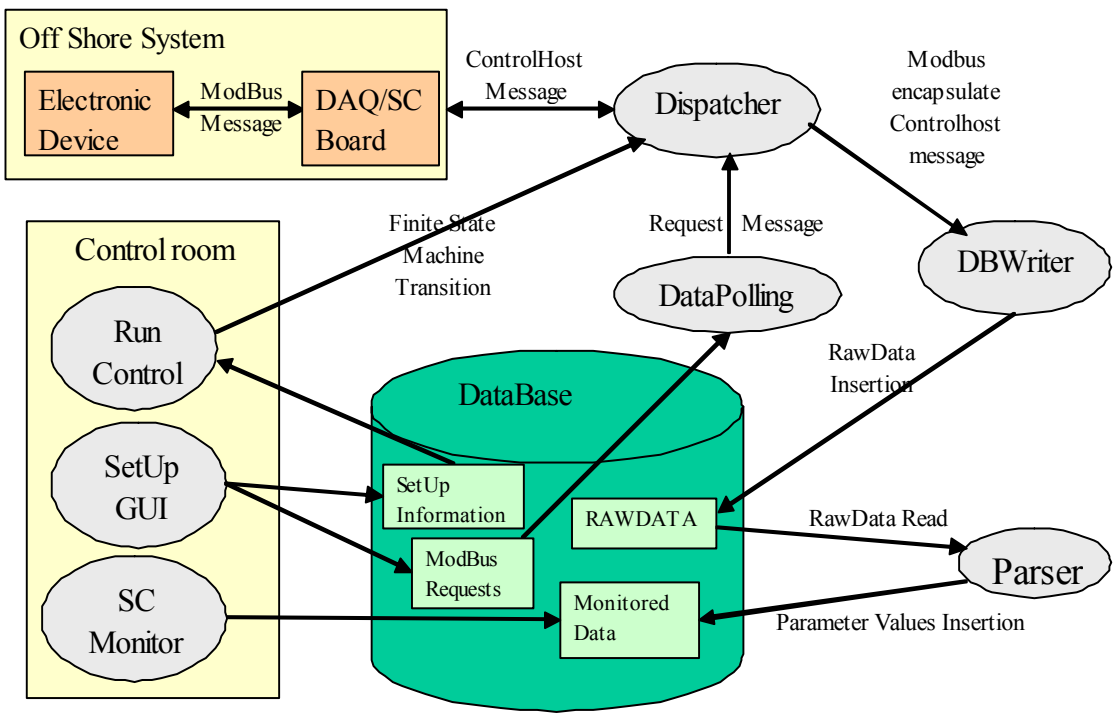

Figure 2: Slow Control General Architecture

POLLING LIST GENERATION

Shifter in the control room mainly acts on Run Control Graphical User Interface to start, initialize, configure and stop acquisition. At the initialisation step, a setup is chosen. Each setup defines the part of the detector to be started, which will give a list of DAQ/SC boards to be polled.

As each board has been associated during the integration phase of the project to a set of electronic types, from the list of DAQ/SC boards, a list of electronic devices can be generated. Then this list is converted into an ordered list of Modbus requests which will be given has input of the DataPolling module.

This dynamic generation ensures that polling is always matching the actual running configuration.

Once raw data have been parsed and written into Monitored Data table, data have to be calibrated. Each parameter has its own calibration function, which is defined into calibration tables of the database.

These calibration functions can be executed either by DAQ programs, by expert programs that check some specific parameters or by standard GUI monitoring programs. Those GUI are available in control room and are always running and used by shifter to check running conditions.

Figure 3 and 4 show two examples of regularly checked monitoring windows. The first one is from Aquadopp monitoring system which scans sea current velocity and direction. The second one shows the compass monitoring system which checks orientation of all the local control modules (boxes that contain DAQ/SC board) of the line number 4.

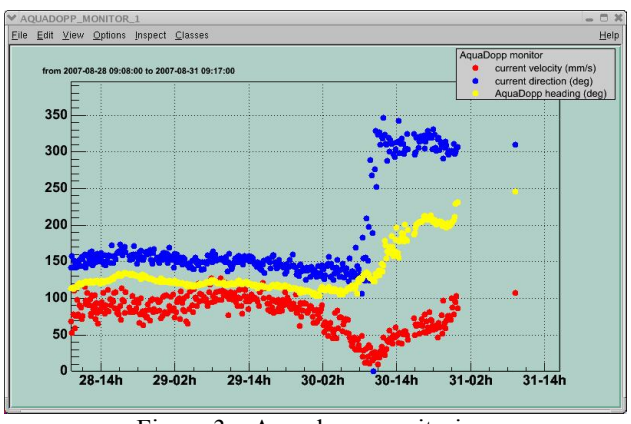

Figure 3 : Aquadopp monitoring.

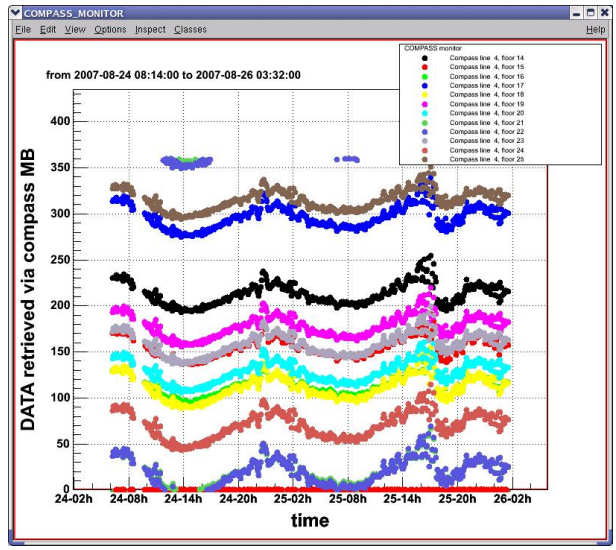

Figure 4 : Compass monitoring.

CONCLUSION

Currently five lines are connected; five new lines are under water and should be connected in December 2007, the whole detector should be completed in 2008.

Slow Control system runs for two years. Since 2006, RawData table has more than 38 millions entries, and Acoustic Monitored Values (the biggest Monitored table) has 147 millions entries.

REFERENCES

[1] Modicom, Inc., Industrial Automation Systems "Modbus Protocol Reference Guide" PI-MBUS-300, June 1996, http://www.Modbus.org

[2] The ANTARES Collaboration, "Technical Design Report of the ANTARES 0.1 km^2 project", V1.0, july 2001, http://antares.in2p3.fr

[3] Shebli Anvar, Hervé Le Provost, Frédéric Louis "The ANTARES Offshore Data Acquisition : A Highly Distributed, Embeded and COTS based System", IEEE - Nuclear Science Symposium Conference Record, October 2000, pp 12/103 – 12/106, vol.2.

[4] R.Gurin, A. Maslennikov, "Controlhost – Distributed Data Handling Package" http://www.nikhef.nl/~ruud/HTML/choo_manual.pdf

[5] Wind River Real Time Operating System http://www.windriver.com/vxworks/.