The Detector Algorithms framework

ALICE detector

ALICE is the heavy-ion detector designed to study physics of strongly interacting matter and the quark-gluon plasma at the CERN Large Hadron Collider.

The detector includes high resolution tracking (silicon detectors, large time-projection chamber), particle identification, and triggering elements. It features two large magnets, a main solenoid and a dipole on the Muon arm.

It primarily targets heavy-ion lead-lead collisions, but it also has a substantial physics program with proton-proton and proton-lead collisions.

Calibration procedures

The 18 ALICE sub-detectors require specific calibration tasks to be performed regularly in order to achieve the most accurate physics measurements. The corresponding set of procedures involves events analysis in a wide range of experimental conditions.

A dedicated framework has been designed and implemented to achieve as much as possible the detector calibration directly online in the DAQ. These calibration tasks may be done either in dedicated runs, or in parallel to physics data taking.

Detector algorithms (DAs)

DAs are provided by the sub-detector teams, using the global framework to develop detector-specific calibration procedures.

Each DA grabs detector data (physics or calibration events) and produces results online. These results can be reused directly online (e.g. to configure the detector electronics or give quality feedback to the Data Quality Monitoring system), or shipped offline (to be post-processed and used in event reconstruction).

A DA consists of some specific detector code (to analyze events and produce results according to a given calibration task), user code (to be post-processed), and a Detector Algorithm (DA) for the DA framework.

Software life cycle

Deployment in production is done after a strict release procedure, including building in a standardized environment and testing on reference data sets. 50 different DAs regularly used online.

328 DA packages upgrades between 2007 and 2012.

Control and bookkeeping

The "DA launcher" starts each DA and follows their execution. It reports to the DAQ runControl the DA status (running, error, exited) and executes commands (stop or abort). It enforces configured resource limitations (e.g. memory used), and if necessary kills the DA process if unresponsive.

The launcher also integrates bookkeeping features. It redirects the DA output messages to the central DAQ logging system, and collects run-time statistics (exit status, execution time, etc) for the experiment e-logbook. These records are remotely accessible by detector experts through a Web interface.

Parallelization

The DA framework provides a set of C++ classes to ease the development of multi-threaded DAs, and to be able to use the full power of multi-core CPUs. It involves independent processing of events in parallel threads and aggregation of partial results.

Detector specific code is effectively implemented by filling a few virtual methods (e.g. processEvent()). The control and data flow between the runtime threads are then transparent to user.

Contact: sylvain.chapeland@cern.ch

Conference on Computing in High Energy Physics, May 2012, New York, USA