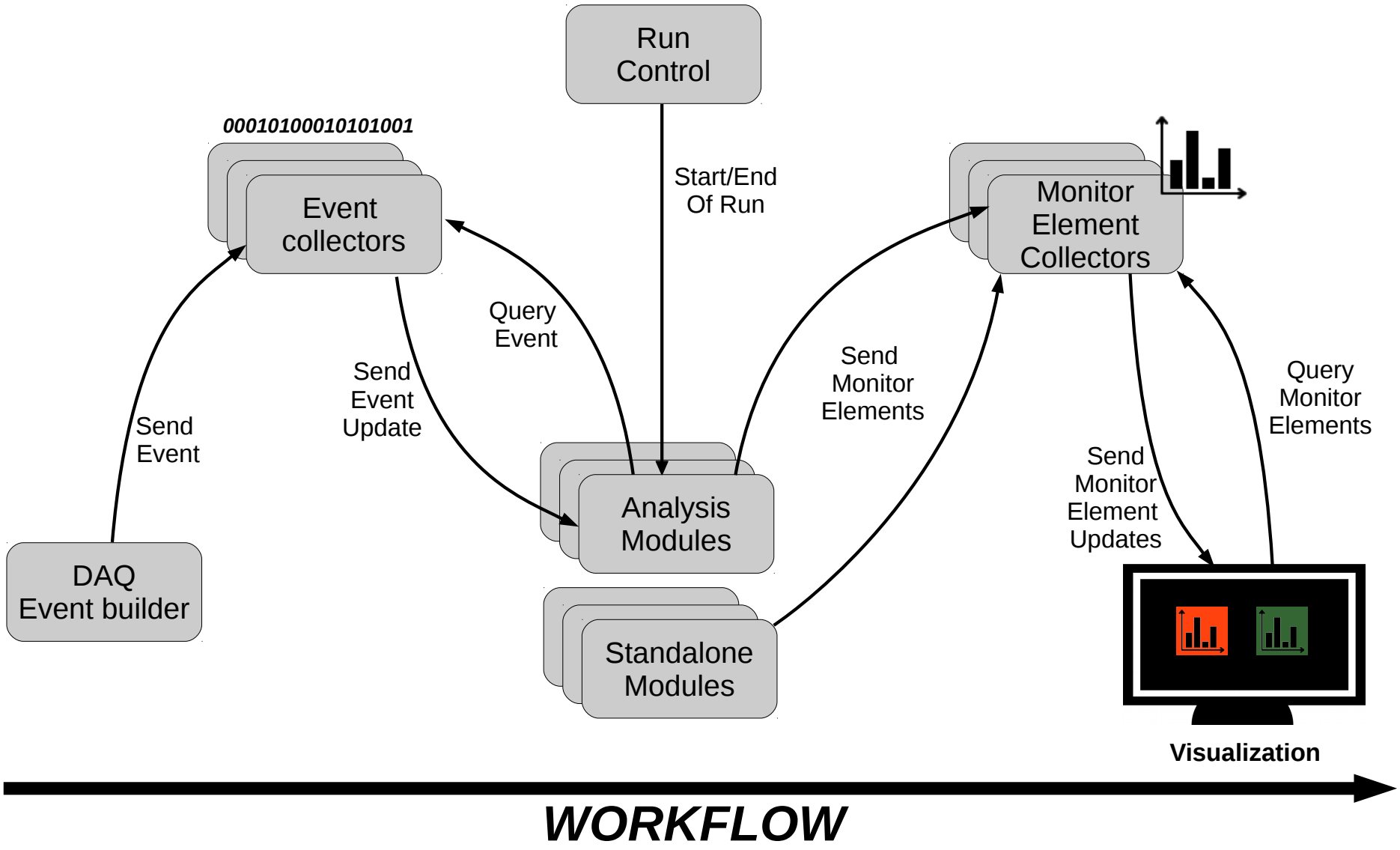
Data Quality Monitoring for High Energy Physics (DQM4HEP)
Module interfaces

R. Été, A. Pingault, L. Mirabito

Université Claude Bernard Lyon 1 - Institut de Physique Nucléaire de Lyon / Ghent University

10 février 2016

Global workflow



Module applications - analysis module

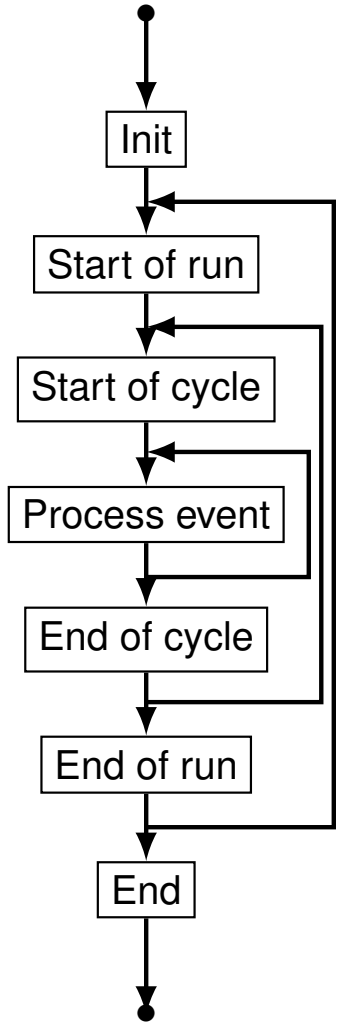
Purpose

- Receive events from a collector server and process them
- Produce monitor elements (histograms, scalars, generic TObject)
- Follow the run control signals (SOR, EOR)

- **Init** : Initialize the application : load dlls, declare services, etc ... Wait for a SOR
- **Start of run** : start cycles loop, open archive
- **Start of cycle** : start a cycle of '*process event*'
- **Process event** : Process incoming event, fill monitor elements, etc ...
- **End of cycle** : send subscribed monitor elements, update archive (opt).
- **End of run** : Wait for SOR, close archive (opt).
- **End** : Clean and exit module.

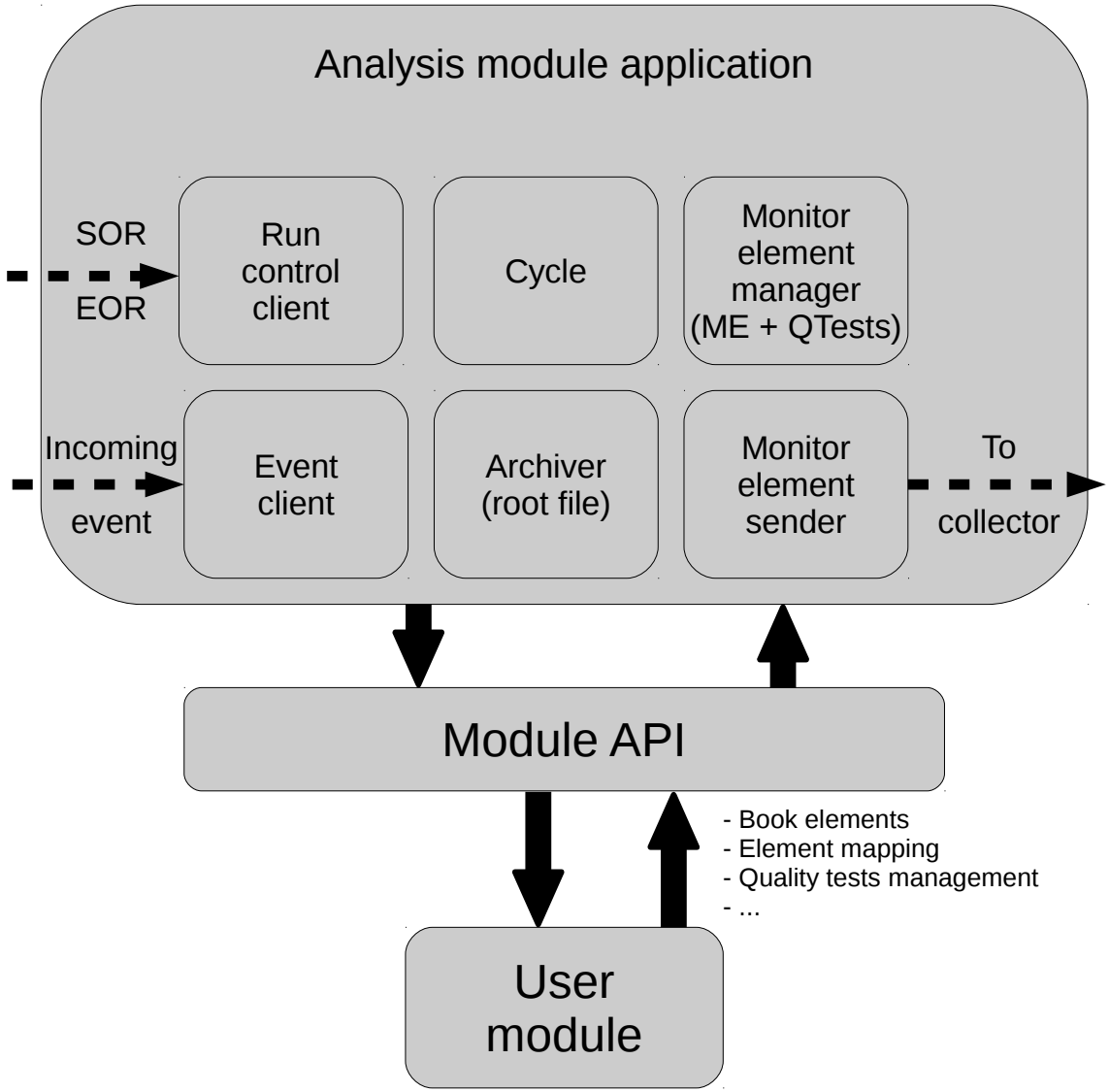
To implement online DQM analysis, user must implement the `DQMAnalysisModule` interface. A shared library must be build and loaded in the application using the plugin system (see next slides).

Use `dqm4hep_start_analysis_module` to start an analysis module.



Analysis module application flow

Module API


Analysis module application
SOR
EOR
Run control client
Cycle
Monitor element manager (ME + QTests)
Incoming event
Event client
Archiver (root file)
Monitor element sender
To collector
Module API
- Book elements
- Element mapping
- Quality tests management
- ...
User module

Module applications - standalone module

Purpose

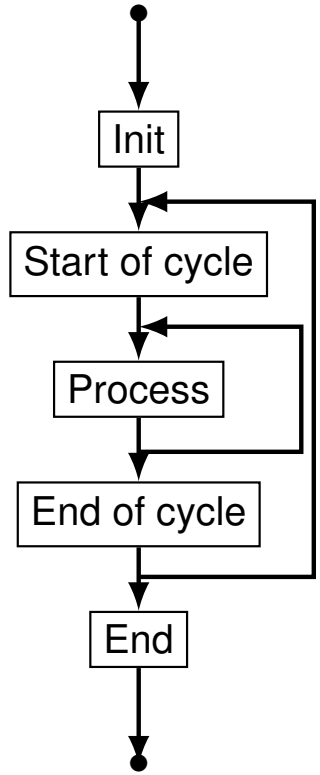
- No event reception
- No run signals
- Produce monitor elements (histograms, scalars, generic TObject)

- **Init** : Load dlls, init the module.
- **Start of cycle** : start a timer cycle of n seconds
- **Process** : call back function.
- **End of cycle** : collect monitor elements and send
- **End** : The application has received a signal to exit and the process ends.

To implement online standalone analysis, user must implement the `DQMStandaloneModule` interface. A shared library must be build and loaded in the application using the plugin system (see next slides).

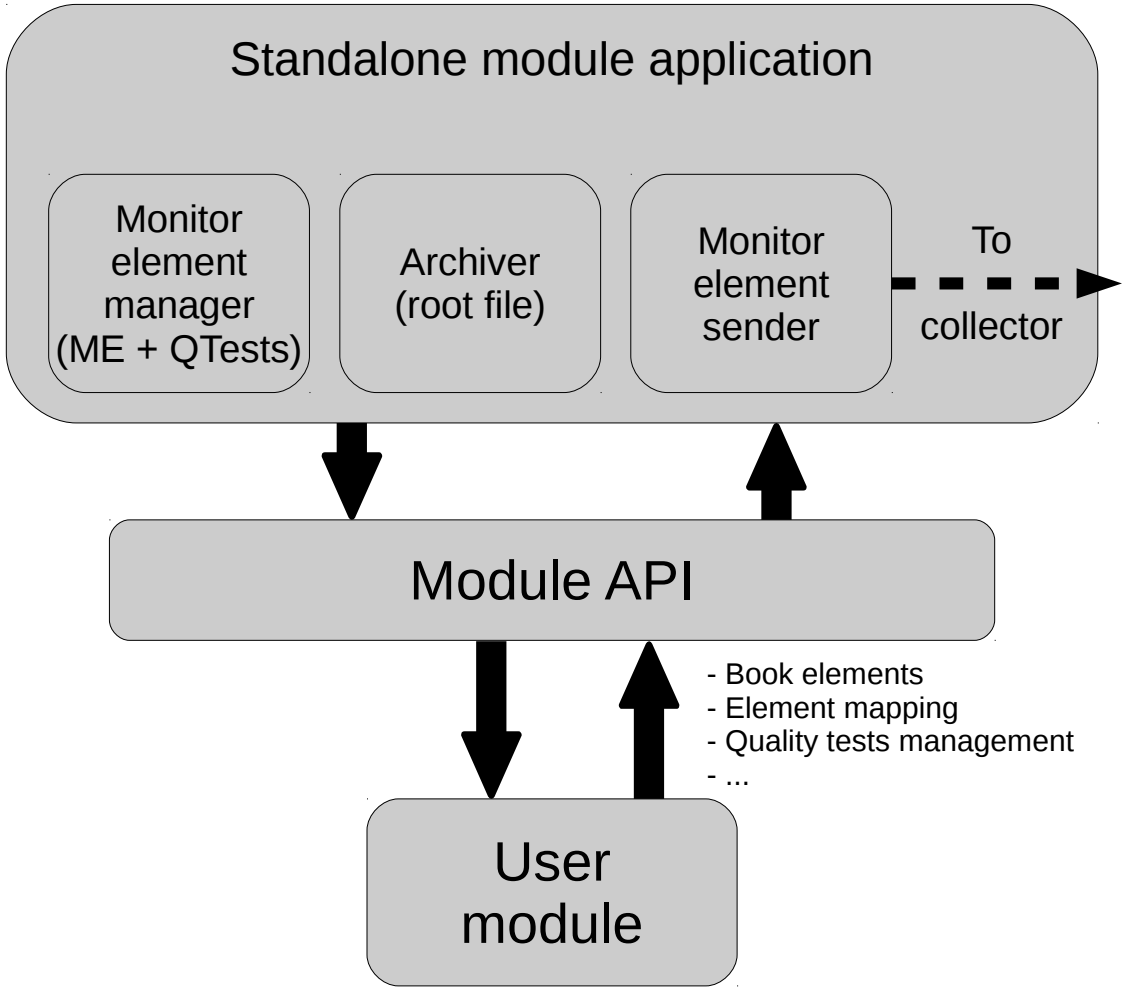
Designed for *slow control - like* data processing.

Use `dqm4hep_start_standalone_module` to start a standalone module.



Standalone module application flow

Module API


Standalone module application
Monitor element manager (ME + QTests)
Archiver (root file)
Monitor element sender
To collector
Module API
- Book elements
- Element mapping
- Quality tests management
- ...
User module

Module API

Data processing performed in **modules** (standalone or analysis).
Modules **book** monitor elements, **fill** them and **publish** them to a single collector.

A monitor element is a wrapper around a ROOT TObject with some additional attributes :
$\rightarrow$ Type, name, path (i.e "/Efficiency/Layer2/"), collector name, quality flag, reset policy, title, description, run number, quality test results.

The `DQMModuleApi` class provide a static interface to perform operations within the application :

- Monitor elements management (book, delete, reset, from xml)
- Directory structure management (mkdir, cd, ls, rmdir, pwd)
- Quality test management (register, add, remove, run, from xml)

Quality tests can be run on a particular monitor element to test the quality of the processed data (chi2, Kolmogorov, user defined).

Note that **QTest results are sent to the collector together with the monitor element** !

Example modules

An LCIO example module can be found on the github page of DQMCore package `https://github.com/DQM4HEP/DQMCore` :

CaloHitModule (lcio, analysis) :

- source/examples/module/lcio/CaloHitModule.h
- source/examples/module/lcio/CaloHitModule.cc
- conf/lcioCaloHit.xml
- `dqm4hep_start_analysis_module` executable