Testbeam monitoring
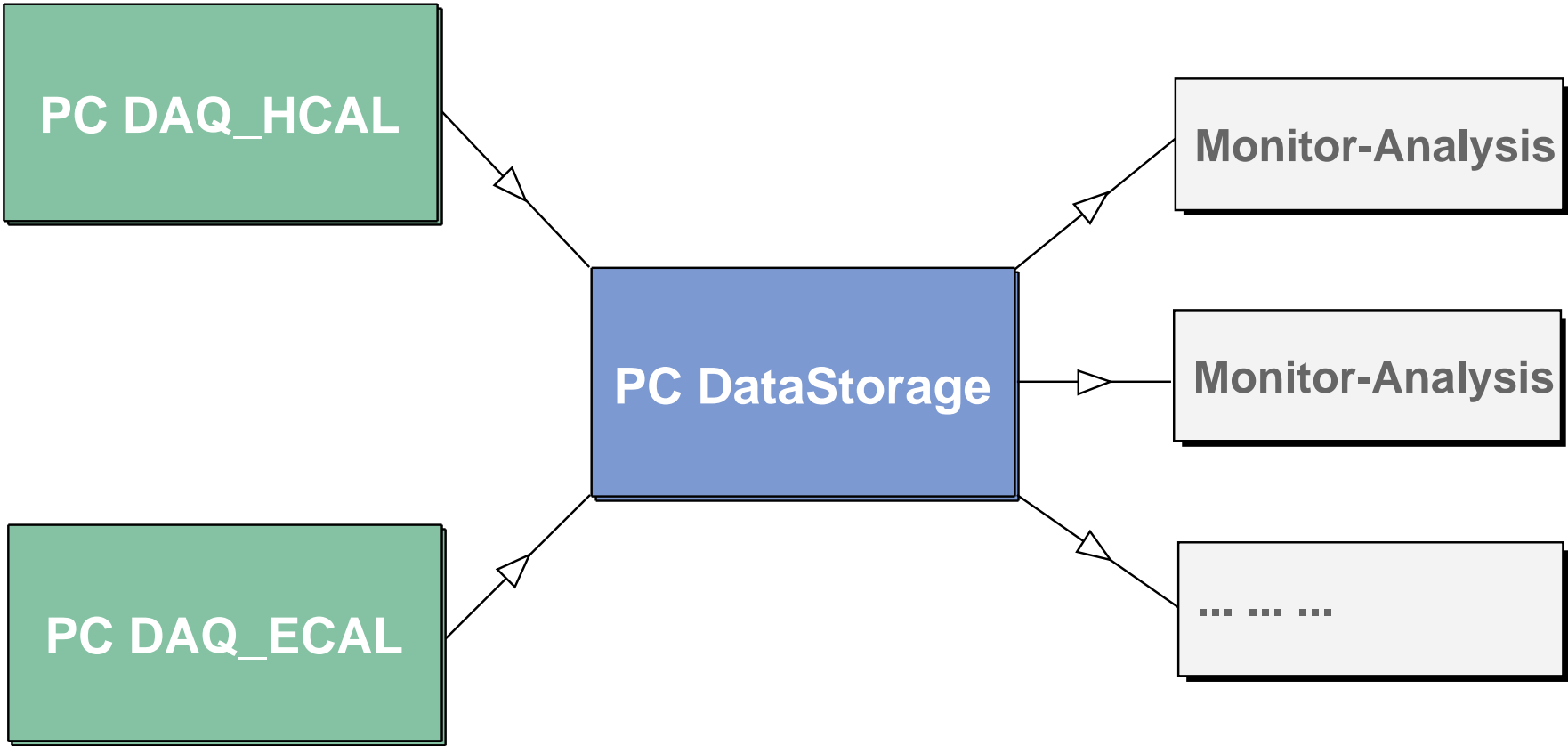
G.Mavromanolakis, University of Cambridge

Outline

► **General**

► **Tasks**

► **Features**

► **Example**

PC DAQ_HCAL

PC DAQ_ECAL

PC DataStorage

Monitor-Analysis

Monitor-Analysis

... ...

General

▶ .

application for realtime or offline monitoring

: code arranged in client-server parts (gui and sender)

: the different parts communicate through a socket
   i.e. can live and run at the same pc or at different pc's

: configuration with single gui - single sender pair
   or single gui - multiple senders
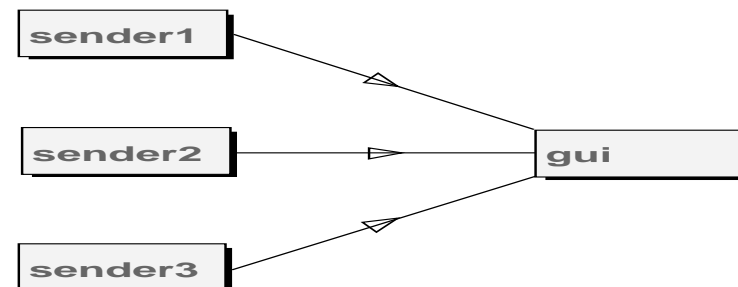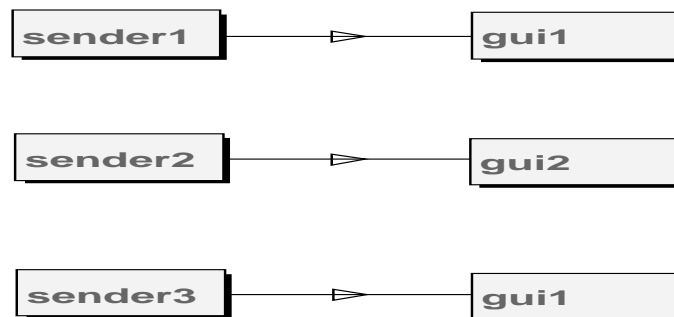
Tasks

► . **gui and sender**

 : gui's task

   ▷ to provide a basic functionality through
     entry fields and action buttons

   ▷ to display the info that the sender sends

 : sender's task

   ▷ to process the data and send plots and numeric info to gui

   ▷ to create a .root file per run with summary plots

   ▷ to create a .slcio file per run with preliminary clean data

General features

► .

: requires ROOT 4.02/04 or higher and LCIO 01-04 or higher

: communication through secure connection using ssh is possible
  i.e. gui and sender running in pc's that are behind different firewalls

: multiple types of transmission can occur,
  currently one light and very frequent (say every 10events) and
  one heavy and less frequent (say every 1000s events or some minutes)

: gui's display pads invoke root interactively

: it is a light application, easy to install/run

General features

► . **input**

    : .bin files produced by daq, mapping files and
        tentative calibration constants

► . **output**

    : .root file with several summary plots per run

    : .slcio file ala mokka containing "preliminary clean" events checked for
                                                           NO TRIGGER
                                                           BAD READOUT
                                                           TDC OVERFLOWN
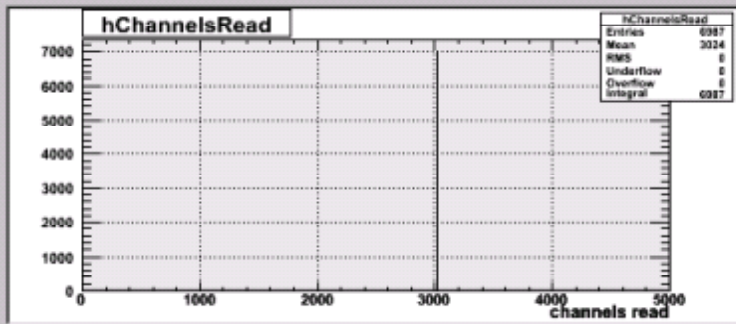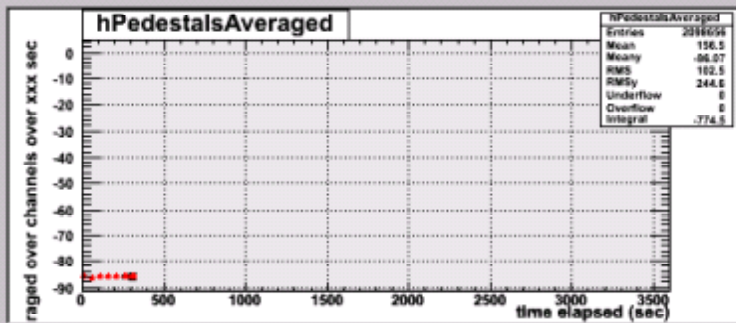   BAD TRACK

    + .ps printout on request and streaming to a webpage possible

Example
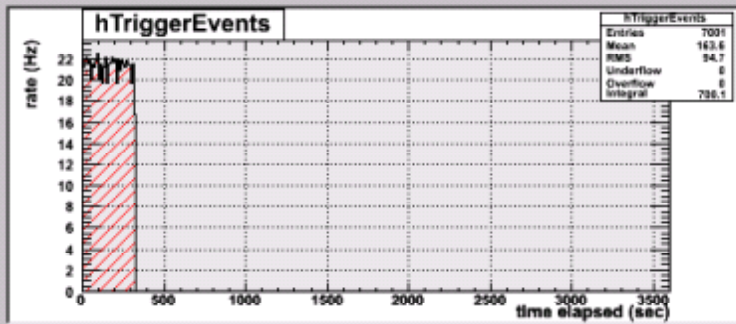
► **.** **info transmitted and displayed**

▷ numerical

: run number, event time, event index

: status counters (#ofevents processed, #ofevents found with "NoTrigger", "BadTrigger", "BadReadout", #of good pedestal and trigger events, #ofevents found with "TdcOverflown", "BadTrack")
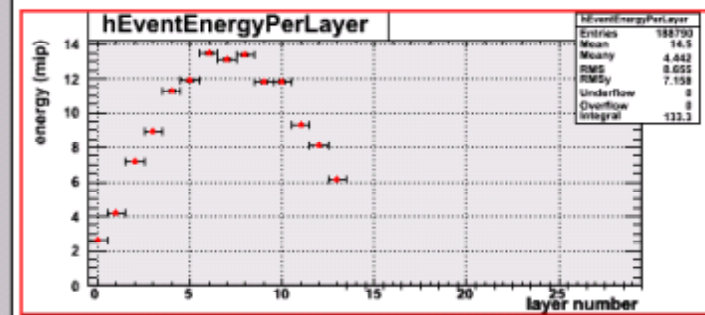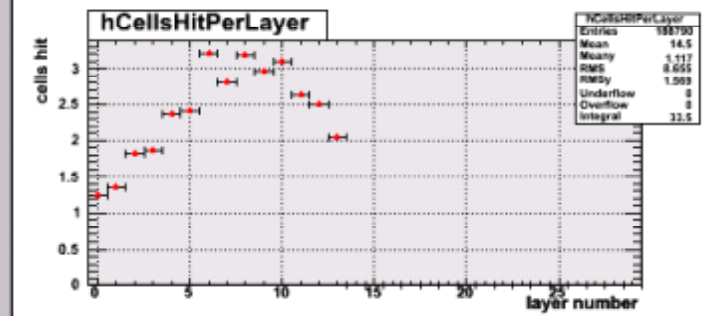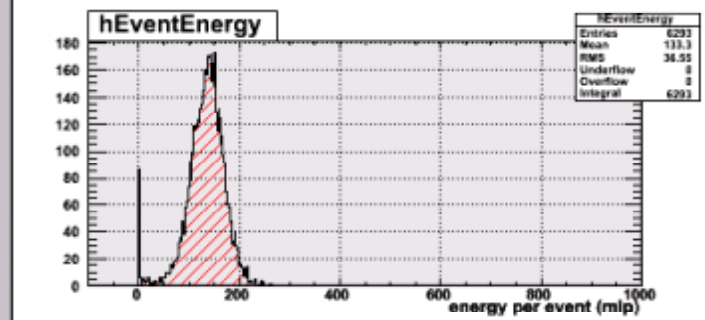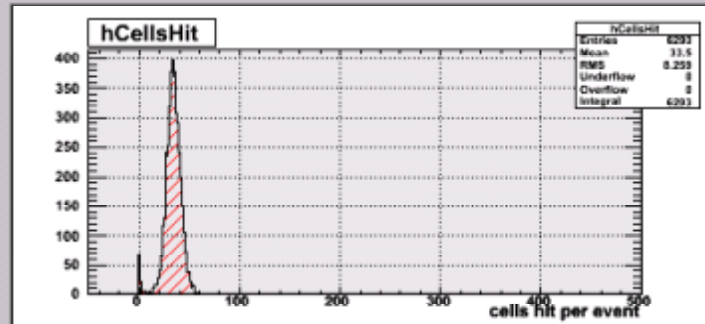
▷ graphical

: 3d event display

: hits per event, energy per event, hits per layer, energy per layer

: daq rate vs time, average pedestal vs time "NoTrigger" vs time, "BadReadout" vs time

: tdc time per chamber, preliminary track coordinates

► **.** **benchmark**

: on a PentiumM 1.7GHz 1GB RAM with linux, running both gui and sender, processing rate achieved is about 170Hz (i.e. about 18 min for 200kevents)

Demonstration ...