Command line arguments and environmental variables

Pierre Lasorak, Marco Roda

The problem

Changes to the environment variables last week:

- “name” and “session” are provided by the environment variable, not through the Command Line Arguments (CLA) of the daq_application.

This means that one cannot know which application is running via top, ps and looking at the arguments

- Breaks the thread pinning that uses a similar mechanism

The quick fix

- 2 revert-PRs merged earlier this week on appfwk/confmodel
- Add the name of the application and session back to the CLA *as well as* the environment variables
- Add a check to make sure these are the same

The proposed solution

We remove the application name and session from the environment

- Exact opposite direction of what we wanted to do with last week's PR
- session and application name are coming from the command line

Logging.setup() is responsible for setting ERS up

- In practice, it means that logging setenv("DUNEDAQ_APPLICATION_NAME") and setenv("DUNEDAQ_SESSION")
- No more setenv by appfwk nor by the run control
- Logging.setup() happens after the CLA have been parse, daq_application now cannot throw an ers::exception if that failed.

Question: Do you use DUNEDAQ_APPLICATION_NAME or DUNEDAQ_SESSION environment variables in appfwk? We cannot guarantee these are set properly once this will be merged.

Code Status

- 3 PRs all in morda/logging branches
  - logging: https://github.com/DUNE-DAQ/logging/pull/37
  - appmodel: https://github.com/DUNE-DAQ/appmodel/pull/140
  - appfwk: https://github.com/DUNE-DAQ/appfwk/pull/301
- unittests are working
- Tested with config/daqsystemtest/example-configs.data.xml
  - local-1x1-config
  - ehn1-local-1x1-config
  - ERS messages are flowing correctly, data as well
- Iomanager should change as well
  - But because it's not strictly necessary for the functionality