

Command line control of runs

But, not run-control

Brett Viren

February 23, 2021

Interim command line interface proposal

We want to **run** some command line program to **control** the DAQ

- Call it generically “cli” to avoid any implication of grandeur.
- Not `run-control`, but `cli` may talk to something of that name.
- It is an **interim** thing, we will likely throw it away at next scale-up.
 - ▶ Ultimately we want some nice web based human-DAQ UI

cli's CLI

```
cli -c run-NNN.json [cmd] [cmdargs]
```

The config file

The `run-NNN.json` holds **all** command information for run NNN.
Produced by script, not directly by humans (unless masochist).
Composed of top attributes:

- `.boot` the boot command from Gio's proposal, discussed last week.
Includes information needed to run a `daq_application` command line to create a running process.
- `.inits` all *init* command objects correlated to each process
- `.confs` all *conf* command objects correlated to each process
- `etc` potentially other collections of commands

ie, what we discussed last week.

cli's CLI - a full startup

Full, monolithic startup of a (mini)DAQ:

```
$ cli -c run-42.json
```

No qualifying options so it applies:

- 1 the *boot* command to start all processes then
- 2 the *init* commands to each process and then
- 3 the *conf* commands to each process and then maybe
- 4 the *start* commands to each process.

And then exits because we'd not want to immediately stop.

cli's CLI - fine grained

Piecemeal operations for experts/developers:

```
$ cli -c run-42.json boot # 1  
$ cli -c run-42.json boot --procs proc1,proc2 # 2
```

- 1 Would apply just *boot* to all processes in the run config file.
- 2 Would apply just *boot* to just the identified processes.

Same patterns would be available for *init*, *conf* and the other “standard” (and custom?) commands.

What about *start*, *stop*, *scrap*, etc

These are trivial command (*start* has additional run number).
Despite the redundancy, we might as well provide them in `run-NNN.json` in the same way as *boot*, *init* and *conf*. The `cli` may then become **agnostic** to the command semantics
In any case, same CLI pattern:

```
$ cli -c run-42.json start
$ cli -c run-42.json start --procs proc1,proc2
```

What about expert commands

Given command agnosticism, an expert may add their own expert commands to a `run-NNN.json` file to enable issuing like:

```
$ cli -c run-42.json mycmd --proc myproc
```

Questions

- Can `cli`, as described, be the human-DAQ UI to drive Glenn's system?
- What desired human-DAQ interactions are not captured by these examples?