DAQ database

Tomoyuki Konno

TRG/DAQ workshop 2016,
Budker Institute of Nuclear Physics, Novosibirsk

DAQ database

- **Configuration DB** explains DAQ initial setups at (re)start of runs
  - Parameters into FEE registers
  - Software settings (ref. # of condDB, HLT scripts, running RC nodes, etc)
  - Power supplies (running channels, voltages etc.)

  => **DAQ configuration DB** except for PXD

  => **PXD configDB** maintained by M. Rizert
- **Logger DB** shows histories of detector status and DAQ activities
  - Text messages with severity from DAQ processes

  => **NSM2 message collector** to DB is running

  => **JMS based framework** for PXD system
  - Monitored values collected via NSM2/EPICS records

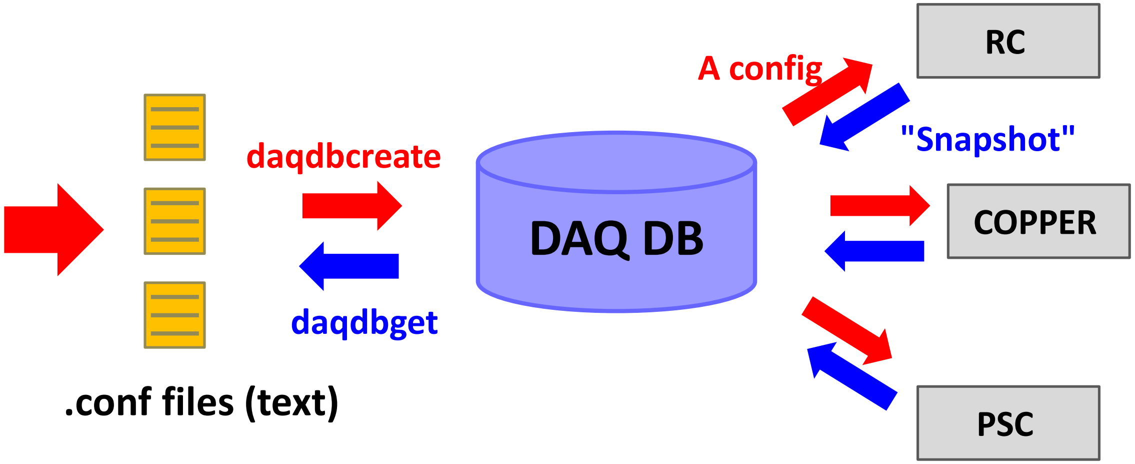
  => **CSS channel archiver** modified M. Rizert

Configuration DB

- **Configuration DB** is based on
  - Text parser into objects with nested structure
  - DB tables to contained objects by spiting variables into table rows
- Conversion between text and DB entries are done by CLT
  - Text -> DB : daqdbcreate <filepath> <tablename>
  - DB -> text (stdout) : daqdbget <tablename> <configname>
- Snapshot of a config entry (edited by manual) is recorded before run start
  - Assigned unique label : <nodename>@<expno>:<runno>:<subno>:<start/end>
  - Same daqdb tools are available
- Python utility is needed but not available yet (my homework)

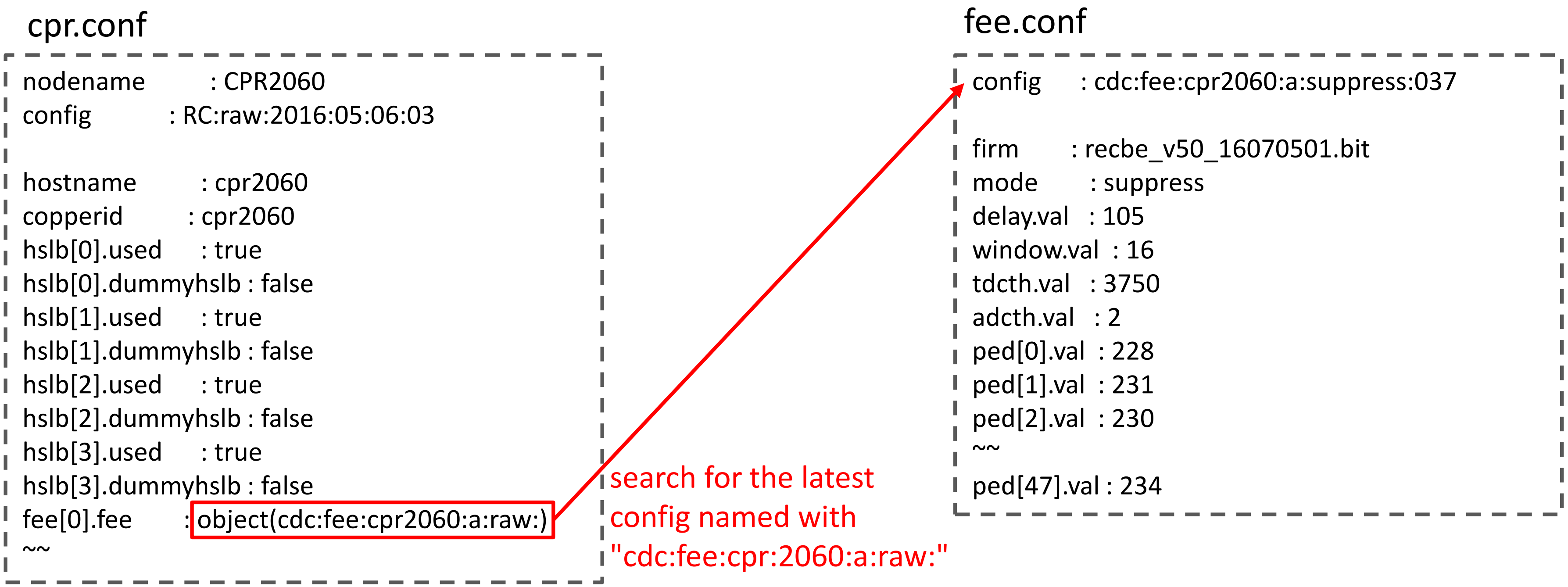
Experts activities

- Measurements
- Calculation
- Condition DB



Creation new configuration

- New configuration can be created from a text file as input:
  - ex.) $ daqdbcreate cpr.conf cdc
    $ daqdbcreate fee.conf cdc
- Each configuration is identified by a configname
  - Assigned a configname as (nodename@)config



cpr.conf

```
nodename         : CPR2060
config           : RC:raw:2016:05:06:03

hostname         : cpr2060
copperid         : cpr2060
hslb[0].used     : true
hslb[0].dummyhslb : false
hslb[1].used     : true
hslb[1].dummyhslb : false
hslb[2].used     : true
hslb[2].dummyhslb : false
hslb[3].used     : true
hslb[3].dummyhslb : false
fee[0].fee       : object(cdc:fee:cpr2060:a:raw:)
~~
```

search for the latest config named with "cdc:fee:cpr:2060:a:raw:"

fee.conf

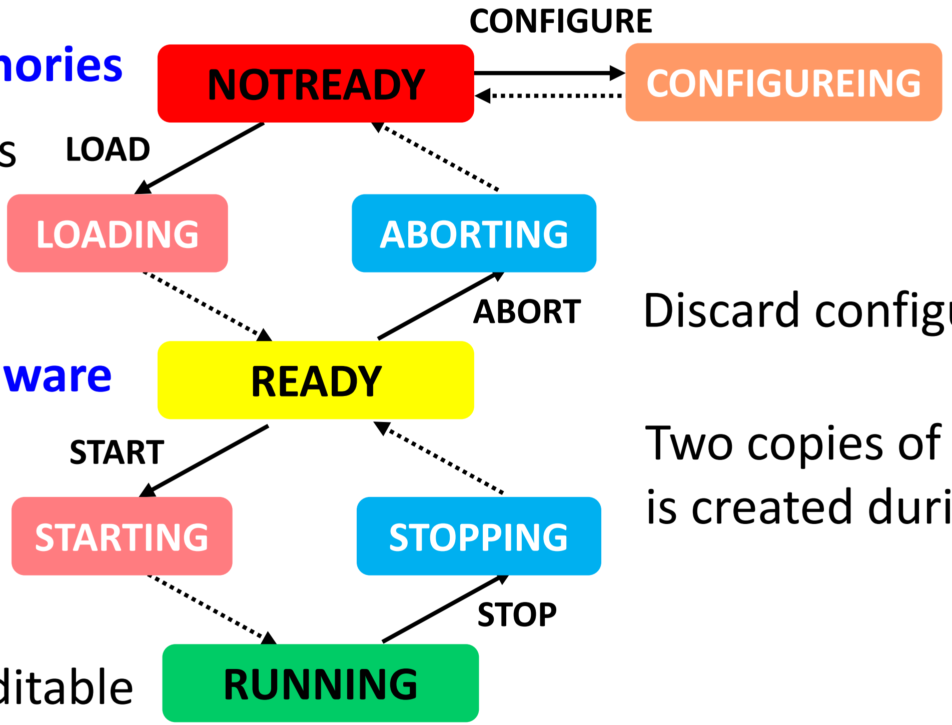
```
config      : cdc:fee:cpr2060:a:suppress:037

firm        : recbe_v50_16070501.bit
mode        : suppress
delay.val   : 105
window.val  : 16
tdcth.val   : 3750
adcth.val   : 2
ped[0].val  : 228
ped[1].val  : 231
ped[2].val  : 230
~~
ped[47].val : 234
```

Configuration and RC state

Select configuration and read from DB

Parameter in memories

Load parameters
to hardware



Discard configuration

Parameter in hardware

Two copies of the configuration
is created during run start/stop

Parameter is not editable

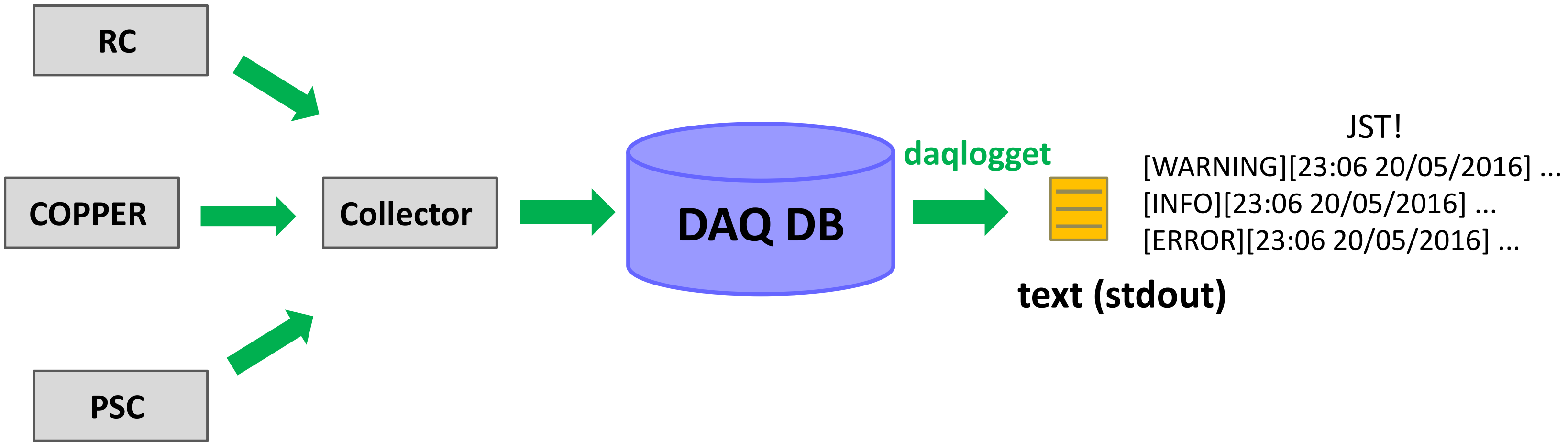
- CONFIGURE carries out replacement of configuration
  - ex) calibration -> physics
  - Manual modification is reset

Configuration of DAQ components (current situation)

- TTD
  - Mapping (ttaddr): hard corded in header files
  - ?? : beyond my mind ...
- COPPER/ROPC : in config. DB
  - FEE parameters : in config. DB
  - Streaming files: in local disk (paths are in DB)
  - eb0 / eb1tx : in config. DB
- HLT : in text files
  - basf2 scripts are called in a python script
    - The scripts are determined in the text files
  - eb1rx : in config. DB
- Storage : in config. DB including eb2rx
- Express reco : in text files (same as HLT)

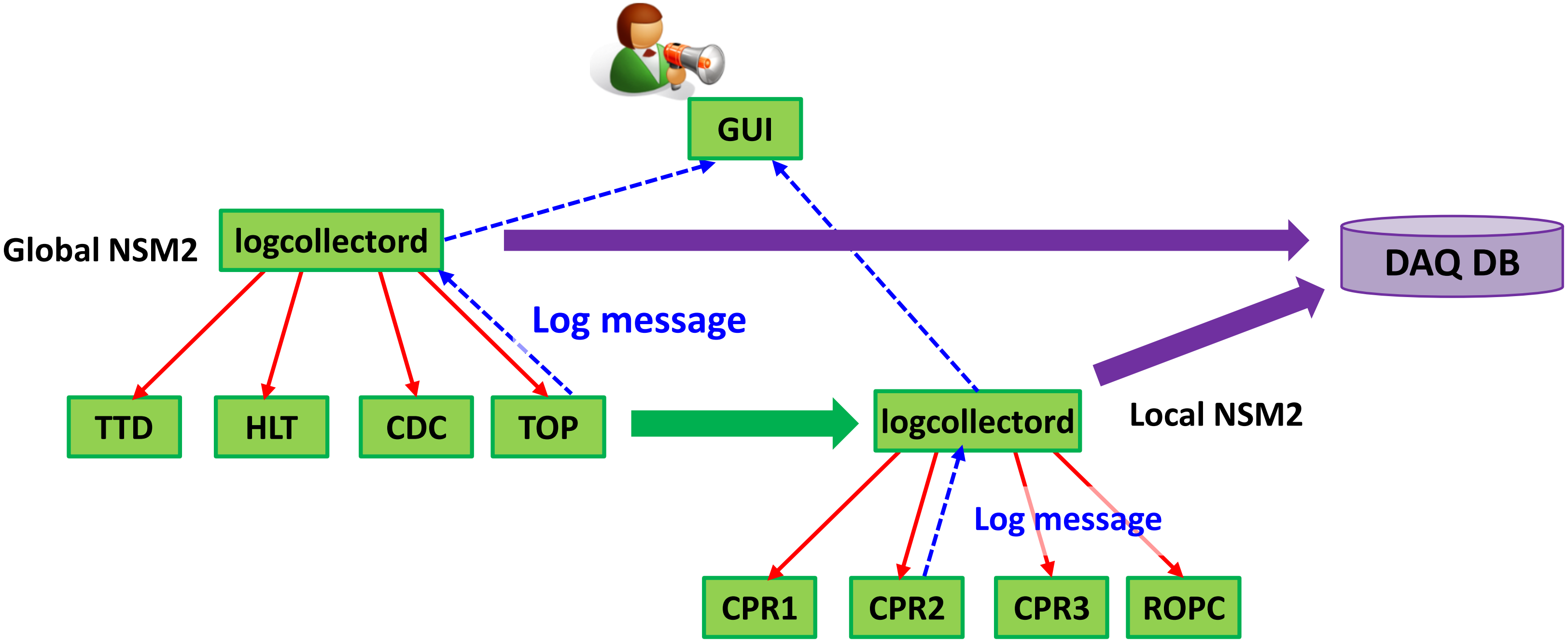
Logger DB (text message)

- **Text Logger DB** is based on
  - NSM2 based message transportation ("LOG" request)
  - Parser and Deparser of text messages
- Command line tools to dump messages into texts are available
  - daqlogget <nodename> [<date>] [severity]
  - Web based view is also available
- Data size estimation is not done yet (homework)



```
[WARNING][23:06 20/05/2016] ...
[INFO][23:06 20/05/2016] ...
[ERROR][23:06 20/05/2016] ...
```

Log collection scheme



- Log message collection via NSM2
- logcollectord : log message collector daemon
  - Unique in a NSM2 segment
    - Each collector has unique NSM2 name to identify the segment
  - Accept NSM message labeled by "LOG"
  - Send to GUI (CSS) and database

How to send log

slow control C/C++ libraries are available to implement codes to send logs

- From HLT basf2 codes
  - B2FATAL, B2ERROR, B2INFO are redirected to HLT slow control
  - HLT Log collection functions are newly developed by Itoh-san
- From slow control programs
  - Implemented in a C++ Class extending NSMCallback
  - log (<priority>, <message>) (a class method of NSMCallback)
- From FEE handler
  - Implement in FEE::monitor(RCCallback& callback, HSLB& hslb)
  - callback.log(<severity>, <message>)
- From generic NSM programs
  - call b2nsm_sendany(node, "LOG", npar, pars, len, msg, NULL)
  - node = NSM name of logcollectord
  - pars = 0 or 1, pars[0] = severity (DEBUG:1-6:FATAL), par[1] = UNIX time
  - msg = C string for text message, len = strlen(msg)

How to handle log severity

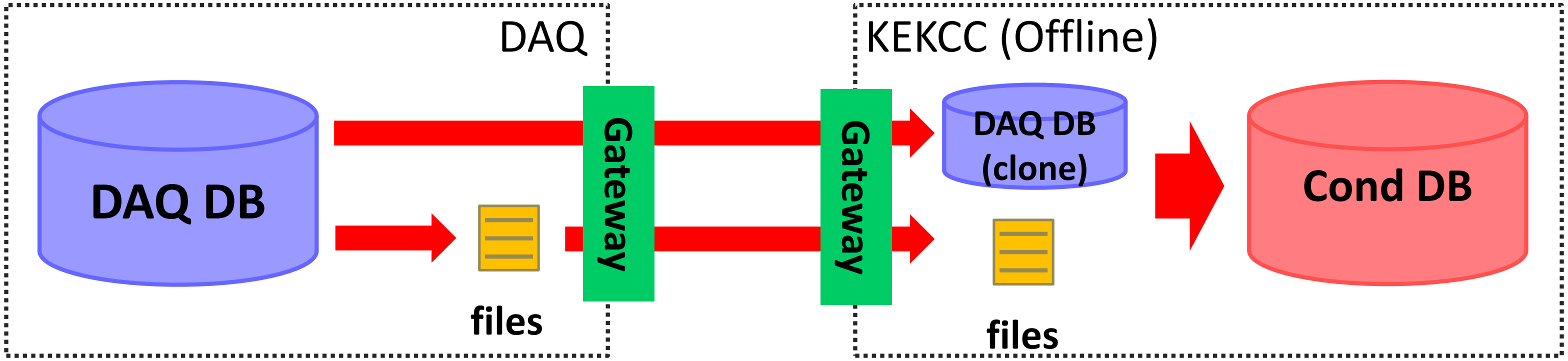
- Log priorities : DEBUG < INFO < WARNING < ERROR < FATAL
  => **Question : What slow control should do for error logs?**
  - DEBUG : (current) nothing shown on GUI but recorded into DB
  - INFO : (current) shown on GUI and recorded into DB
  - WARNING : (current) shown on GUI and recorded into DB
  - ERROR : (current) shown on GUI and recorded into DB
  - FATAL : (current) shown on GUI and recorded into DB

Currently
No difference

- **Should run control suspend / abort runs according to log priorities?**
  - INFO / WARNING : Nothing to do
  - ERROR : Suspend run. Trigger is stopped but others are still running
  - FATAL : Abort run. All subsystems are back to NOTREADY
- Other NSM messages "ERROR" and "FATAL" can be sent to runcontrold
  - ERORR and FATAL abort the current run
    => ALL components are back to be "NOTREADY"
  - Should logcollectord redirect error log messages to runcontrold?

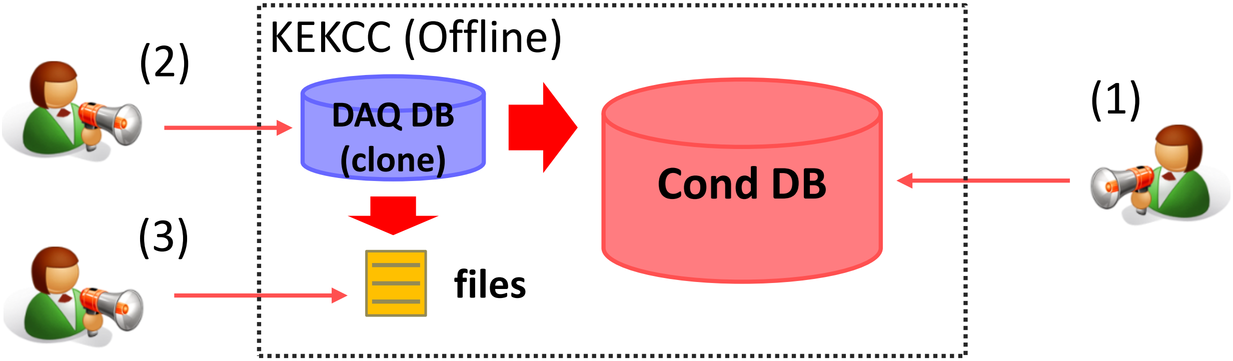
Exporting DAQ DB

- Replication of database itself
  - DAQ accepts postresql (5432) port from KEKCC
  - Native method (or popular) methods of PostgrelSQL replication
  - Config DB is converted into Cond DB for offline usage
- Transport **dumped files** for monitoring
  - http (80) access to get files by wget or curl
  - KEKCC side downloads files periodically (one per day?)
- Yamagata-san and T.Hara-san are negotiating with KEKCC to open connection
  - Much complicated concerns in network security
  - Closed network and restricted ports are available



How to use config. DB in offline

- Copy of configuration is stored in each run start/end
  - Stored in different DB tables : ex) cdc -> cdc_log
- Offline people will convert the replica of config DB to condDB
  - I have no idea who takes care and how they convert yet...
- But config DB itself is available for basf2 software since the codes in basf2
- There are many possibilities to access configuration:
  - (1) Access to Cond DB with converted configurations
  - (2) Direct access to config DB clone
  - (3) Text files dumping the configuration from config DB



Summary

- DAQ use database for configuration and logging
- Configuration DB is used in many parts of DAQ components
- Message logging scheme is also in operation
  - HLT log collection is newly implemented
- Infrastructure of DB replication is designed by Yamagata-san
- Conversion of config DB to cond DB is still unclear
  - Usage in offline is also not clear yet