Partition configuration maker

Argyris Zardilis

University of Cambridge & CERN

DAQ/HLT meeting, 15 August 2013

Motivation

- Large amount of information in a partition, hard to create its configuration manually
  Solution: Use a tool that automates all or part of the process
- Existing tool depends heavily on system design and schema. After new DF design can't do the job anymore.
- Last Technical Run: need to create partition of different flavours to test different components of the system and their combinations.
- Manual partition configuration generation laborious and time-consuming
- Limited amount of time in a TR setting to perform all necessary tests

Solution

- Created a tool following the new system design that automates the process
- Started as a temporary solution for TR2 but since grew to a more complete and configurable command-line tool akin to the older one.
- By no means as complete or comprehensive but covers basic use cases
- Set of python scripts to create basic segments or complete partitions
- Build on top of config and Python DAL package to get access to base classes defined in OKS schema and OKS ConfigObject(for classes not in standard schema bound to Python DAL)
- Also uses pm.project from the old package to get a convenient handle to the config db

Solution schematically

Include a diagram of the modules

Capabilities

and incapabilities

- Create localhost or multihost partitions in testbed/P1
  - farm description loaded from user provided python dictionary
- Create standard partitions: only DCMs, only HLTPUs
  - customisable through command line parameters
- Create standalone, pluggable HLTSV segment
- Doesn't handle ROS segment generation yet
- Also includes configuration for standard monitoring applications
- Other configurable parameters: extra includes, data networks, repository root

Example usage

Create a DCM only partition with a provided python file 'farm_gen' containing farm description with partition name 'az_test' and repository root my home directory:

- `tdaq_python pm_evo.py -p az_test -f farm_gen --dcm-only -r tbed/user/azardili/installed`

Create a DCM/HLTPU partition with a provided python file 'farm_gen' containing farm description with partition name 'az_test' with PuDummy.data.xml as extra include:

- `tdaq_python pm_evo.py -p az_test -f farm_gen -I PuDummy.data.xml`

Conclusions

and Future Work

- More crude version proved useful for some use-cases in TR2.
- Short term plans: Use this more polished version more extensively in next TR or for tests in testbed
  - Bonus: can be used by anyone now and it doesn't require knowledge of the code anymore!
- Longer term plans: Build upon this based on usage experience and input from more people so that it becomes a complete and mature tool
  - Discover use-cases and extra requirements
- Current version lives in a private git repo on my public afs partition which you can clone:
  - `/afs/cern.ch/user/a/azardili/public/partition_maker`