Cluster deployment tools for CBM online

DDS

Bartosz Soból

Deployment tools

What we have now?

- Set of bash script for managing slurm jobs and configuration

Why we need a dedicated tool?

- Current solution is not flexible, error prone, etc.
- The larger the deployment, the harder to maintain a scripted solution
    - VT will be more complex setup than mCBM

Dynamic Deployment System (DDS)

A tool-set that automates and significantly simplifies a deployment of user defined processes and their dependencies on any resource management system using a given topology.

- Developed at GSI
  - Part of the Alice stack (AliceO2, FairRoot, FairMQ, ODC, DDS)
- Configuration via XML files, management via CLI
- Supports local, remote (SSH) and cluster (Slurm) deployment
- No modifications to task code required *except one optional feature
- Alternative software from outside of GSI: HyperQueue

DDS topology definition

- XML-based schema
- Set of tasks (executables) to be deployed
  - Tasks can be organized into groups and collections
    - Collection ≈ group to be executed on the same agent (compute node)
  - Environment setup for tasks supported
- Variables
- Triggers (restart on crash)
- Requirements (specific hostname, potentially GPU availability, etc)

DDS topology definition

Simple example:

```
<topology name="cbmreco_minimal">
  <var name="cbmreco_cmd" value="cbmreco . . ."/>
  <var name="reco_env" value="run_env.sh"/>

  <decltask name="cbmreco_worker">
    <exe reachable="true">${cbmreco_cmd}</exe>
    <env reachable="true">${reco_env}</env>
  </decltask>

  <main name="main">
    <group name="reco" n="4">
      <task>cbmreco_worker</task>
    </group>
  </main>
</topology>
```

Other features

- Properties - mechanism for simple communication between tasks
  - Key-value pairs
  - i.e. exchanging tcp ports during startup
  - Requires code changes (C++ API)
- C++ API for session management
  - In addition to CLI
  - (I’d be great if it has Python wrappers)

Basic usage

```
$ dds-session start

$ dds-submit -r localhost --slots 1

$ dds-topology --activate cbmreco_minimal.xml

. . .

$ dds-topology --stop

$ dds-session stop

$ dds-session clean
```

mCBM replay

- Up to 9 compute nodes, 3 types of tasks, ~50 LOC topology definition
  - >=1 nodes running collection of tasks
    - *1 tsclient* task reading .tsa files from 1 of 8 fles datastreams, shm output
    - Multiple *cbmreco* tasks
  - Single histogram server task on separate QA node
- Common env file with config
- Each task executes simple wrapper constructs command based on env and DDS context
- Auto restart task after crash

```
i=$DDS_COLLECTION_INDEX
# N, I, J = f(i)

INPUT="file://$DATA_ROOT/${RUN}_node${N}_${J}_%n.tsa"
OUTPUT="shm://127.0.0.1/flesnet_$I...."

TSCLIENT_ARGS=()
TSCLIENT_ARGS+=("-a")
TSCLIENT_ARGS+=("-i" "$INPUT")
TSCLIENT_ARGS+=("-o" "$OUTPUT")

tsclient "${TSCLIENT_ARGS[@]}"
```

mCBM replay - test on run 3516 (20 minutes of data)

Slurm configuration file (addition to DDS-generated base Slurm options)

```
#SBATCH --account=cbm
#SBATCH --partition=main
#SBATCH --time=04:00:00   // it actually finished in ~100 minutes on 64 single-threaded workers
#SBATCH --mem-per-cpu=12GB
#SBATCH --nodelist=lxbk0900  // only slurm_histserv.sfg
export SLURM_SINGULARITY_CONTAINER=$LUSTRE/dev_04_2025.sif
```

Execution

```
$ dds-session start
$ dds-submit -r slurm -n 8 --slots 9 -c slurm1.cfg //  for 8 x (8 x cbmreco + tsclient)
$ dds-submit -r slurm -n 1 --slots 1 -c slurm_histserv.cfg //  for histserv
$ dds-topology --activate online_replay.xml
```

Issues

- There's no *proper* way to reserve >1 CPU ter task slot
  - Required for multithreading
- I found two *bugs* (incompatibilities with Slurm version on Virgo)
  - Reported https://github.com/FairRootGroup/DDS/issues/490
- Some things are not documented (basic functionality is well documented)
- Requires rather recent versions of dependencies
  - Not really a bad thing, but can be painful
- I had to build DDS locally on Virgo because Slurm is not available on custom containers
  - It is on VAE containers, but I don't yet know how to make it work

Observations/ summary

- DDS is designed for set of long-running tasks that fit into the topology at once
  - Not an issue for online setup
- Some features are not flexible, probably developed for a specific need in Alice
- At the same time, other things like Slurm integration can be highly customized
- It was easy to reproduce mCBM setup as DDS topology
  - But with challenges with running on Virgo
- A bit of scripting is still needed for command building (arguments)
  - Minimal and clean
- I believe we can use DDS for complete mCBM and VT
- We have devs on-site