DataSpaces

BoF: Enabling Data Services for HPC

Philip Davis (pretending to be Manish Parashar)
Rutgers Discovery Informatics Institute
The Problem Opportunity

• The volume of data produced by analysis workflows is increasing faster than the storage capacity of HPC machines
  – Simulation output for Viz/Analysis
  – Data reduction is not the (whole) answer
  – Storage bottleneck threatens to stall traditional post-hoc analysis workflows

• In-situ analysis to the rescue
  – Keep simulation products in memory
  – Perform analysis concurrent with simulation
  – Live analysis gives us some capabilities we didn’t have before

• Similar for multi-physics workflows
• This is HPC, and we want purpose-driven middleware that does this well
DataSpaces

- In-memory data staging for coupling workflow components
  - “workflow component” can be read “process group”
  - Compare to exchanging data using files
  - Decouple coupling in time
  - Decouple coupling in space (DART communication library)
  - Low-latency, asynchronous data transfer with RDMA

- Optimized for HPC workflows
  - Assumes a shared space abstraction
  - Provides spatially-significant put/get primitives (variable x bounding box x version)
  - Indexing optimized for spatial locality of access using SFC. This permits scalability using a DHT
DataSpaces Project History

• Product of Rutgers University (Center for Autonomic Computing -> RDI2)
• Originally developed to meet coupling challenges in multi-physics fusion simulation and analysis workflows
• Builds on the DART communication layer for RDMA communication
  – Originally supported Cray Portals, has been extended to DCMF, InfiniBand, OmniPath, GNI, and TCP
• A publically available standalone software product since 2012
• Has been used to accelerate applications in a wide variety of research domains (e.g. fusion, combustion, seismic, material science)
• Framework Extensions:
  – Stacker: deep-memory hierarchy support
  – ActiveSpaces: flexible code execution in staging nodes
  – DataSpaces-as-a-Service (DSaaS): persistent staging service
  – Corec

Stacker: An Autonomic Data Movement Engine for Extreme-Scale Data Staging-Based In Situ Workflows
Thursday, 4:00 C141/143/149
Deployment

• DSaaS
  – Deploy dataspaces as a persistent staging service, rather than (just) a transient communication facility
  – Allow workflow components to come and go
  – Permit more complex interactions (e.g. an analysis routine is spun-up conditionally, gets simulation output from DataSpaces, the routine completes, nodes are repurposed, etc…)
  – Wire-up is done by common configuration files and bootstrapping sockets

• Cross-job coupling
  – Fun stuff! On-demand analysis jobs, multi-scale simulation, etc.
  – Stage-in / Stage-out: save resources for jobs
  – Scheduler issues: cross-job communication credentials
  – Scheduler issues II: we’d like support to resize a job!

• Security
  – So far an impediment(!), but this is probably a bad attitude
  – We don’t understand this well yet, but it’s an important issue
Resiliency

• Process-level resiliency
  – Code-coupling model presents challenges and opportunities
  – Complex workflows present an opportunity to isolate failures to individual components
  – Persistent services present greater resiliency challenges:
    • Likely to see a process failure
    • Can we survive system restart?
  – CoREC (Duan, 2018) provides survivability features to DataSpaces using a hybrid approach: replication for hot data, error coding for cold data
  – Process group survivability is still a challenge, work on this in ULFM and CAARES

• Data resiliency as a feature
  – Deal with data corruption
  – Localize failures: prevent data corruption from propagating to other workflow components