DataSpaces as a Service and the EPSi Workflow

Melissa Romanus, Fan Zhang, Tong Jin, Qian Sun, Hoang Bui, Manish Parashar
Rutgers University

dataspaces.org

Introduction

- Scientific workflows running at exascale produce large amounts of data
- This data must be effectively managed in order to accelerate the time it takes to glean insight about scientific phenomena
- Accelerating the time it takes to process this data can be achieved using data staging, i.e., using a set of dedicated compute nodes strictly for I/O purposes during the execution of a scientific workflow

Motivation

- Current data staging techniques require that all applications in the workflow connect to the staging area at initial runtime
- Current methods cannot support dynamism in workflows
- DataSpaces can be modified to support these types of workflows, by offering a persistent data staging service
  - Applications connect and disconnect from the space
  - Optimizes resource utilization
  - Improves overall workflow performance

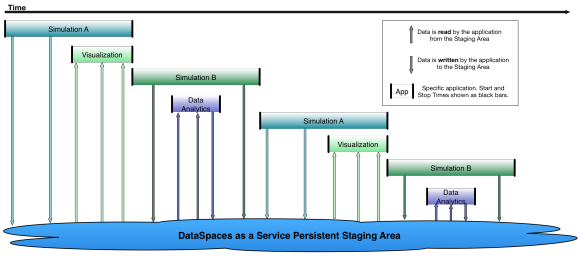

Figure 1, Illustration of scientific workflow interacting with DataSpaces as a Service. The staging area persists over time as different applications connect, read and/or write data, and disconnect to/from it.

Key Features

- **Dynamic**: Coupled applications can join and leave the staging area without affecting other applications
- **Persistent**: The staging area (consisting of DataSpaces *servers*) remain active on the infrastructure as a service, instead of shutting down when the workflow completes
- **Efficient:** Optimizes the write performance by routing data from a requesting client application to the closest staging servers
- **Resilient:** DSaaS is capable of backing up the data stored on staging servers and restarting, i.e., after failure or if workflow wants to resume from a certain stage

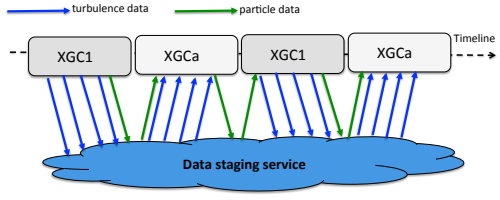

Figure 3, EPSi+DSaaS Workflow. XGC1 runs on a number of cores and generates particle and turbulence data which it writes to the staging service. When XGC1 code terminates, XGCa code runs on the same resources and executes code that refines the plasma parameters. When XGC1 runs again, it reads the new parameters from the staging service and continues execution with these new parameters.

Conceptual Architecture

- DSaaS offers staging areas both on-node and in the traditional manner, i.e., via dedicated compute nodes.
- Applications write to the closest DS server when space is available. Apps read from closest server when applicable.
- When applications shut down, their data persists in the DS. Other applications can utilize the released cores and access this data.
- Application checkpoint data is periodically written to more permanent storage

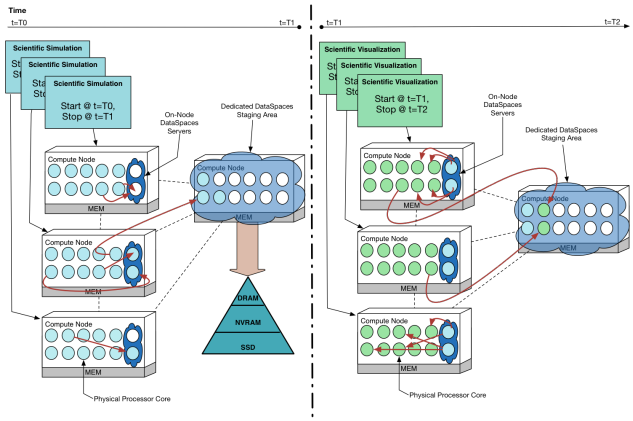

Figure 2, DSaaS Architecture. Scientific Simulation runs from T0-T1. Next, a Visualization code runs (from T1-T2) on the same cores and can utilize the Simulation data stored on-node or in the dedicated area with having to go to further levels of the storage hierarchy.

EPSi + DataSpaces as Service

- Fig. 3 shows the EPSi workflow. XGC1 code runs to compute particle and turbulence data before shutting down. Next, XGCa runs to refine the plasma parameters and shuts down, then XGC1 code runs again, and so on.
- In previous DS implementation, although only 1 code was running at a time, 2x the resources were connected at startup because of the static requirements. At any time, ½ the resources were idling.
- XGC1 and XGCa codes can read/write from node-local staging when applicable instead of the traditional staging area to improve performance.

Experimental Parameter	Setup 1	Setup 2	Setup 3
Num. of processor cores	1024	4096	16384
Num. of coupling iteration	2	2	2
XGC1 num. of steps (per iteration)	20	20	20
XGCa num. of steps (per iteration)	20	20	20
Size of particle data written/read by XGC1 (per iteration)	4.05 GB	16.21 GB	64.85 GB
Size of particle data written/read by XGCa (per iteration)	4.05 GB	16.21 GB	64.85 GB
Size of turbulence data write by XGC1 (per iteration)	0.19 GB	0.19 GB	0.19 GB
Size of turbulence data read by XGCa (per iteration)	12.63 GB	50.52 GB	202.09 GB

Table 1, Experimental setup of EPSi + DSaaS evaluation of XGC1-XGCa coupled simulation workflow

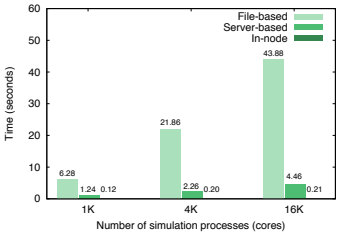

Figure 4, Read time of particle data. In-node sharing decreases particle read time by 98% on average as compared to file-based approach and 93% as compared with server-based approach

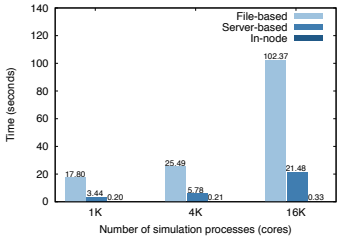

Figure 5, Read time of turbulence data. In-node sharing decreases turbulence read time by 99% on average as compared to file-based approach and 96% as compared to server-based approach.