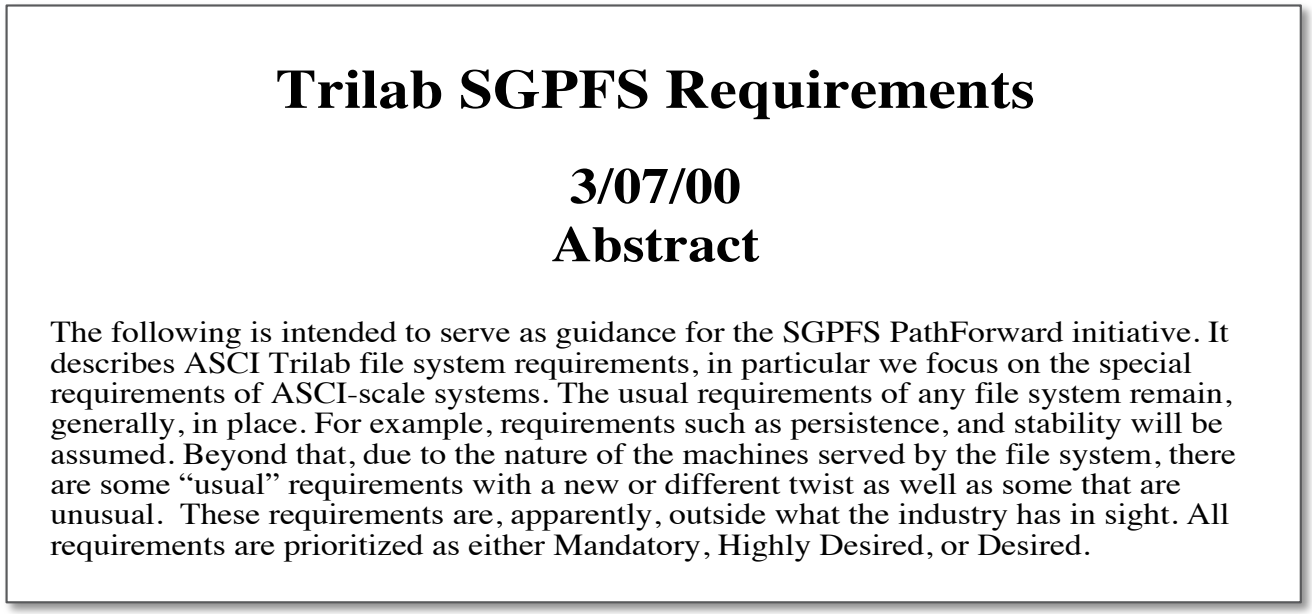
Argonne NATIONAL LABORATORY

FROM FILE SYSTEMS TO SERVICES:
CHANGING THE DATA MANAGEMENT MODEL IN HPC

ROB ROSS Argonne National Laboratory, rross@mcs.anl.gov
GARTH GIBSON Carnegie Mellon University, garth@cs.cmu.edu
JEROME SOUMAGNE The HDF Group, jsoumagne@hdfgroup.org
GALEN SHIPMAN Los Alamos National Laboratory, gshipman@lanl.gov

Salishan Conference on High-Speed Computing, April 28, 2016

HPC DATA MANAGEMENT IN THE 2000s



Trilab SGPFS Requirements

3/07/00

Abstract

The following is intended to serve as guidance for the SGPFS PathForward initiative. It describes ASCI Trilab file system requirements, in particular we focus on the special requirements of ASCI-scale systems. The usual requirements of any file system remain, generally, in place. For example, requirements such as persistence, and stability will be assumed. Beyond that, due to the nature of the machines served by the file system, there are some "usual" requirements with a new or different twist as well as some that are unusual. These requirements are, apparently, outside what the industry has in sight. All requirements are prioritized as either Mandatory, Highly Desired, or Desired.

- Focus was on the POSIX file system model:
  "The usual requirements of any file system, remain, generally, in place."
- We've gotten a lot of mileage out of this model.

Thanks to G. Grider for digging this up!

WHAT'S WRONG WITH POSIX?

- Storage model – no notion of locality
- Consistency – strong consistency requires heavy-weight enforcement
- Data model – difficult to map complex, distributed data sets into single "stream of bytes"

WHAT'S WRONG WITH TODAY'S PARALLEL FILE SYSTEMS?

- Expensive – typically rely on expensive underlying hardware
- Fragile – poor fault handling
- Inefficient – poorly utilize new fast devices, heavy-weight consistency management (arguably this last isn't their fault)
- Inflexible – not built to support a variety of application abstractions

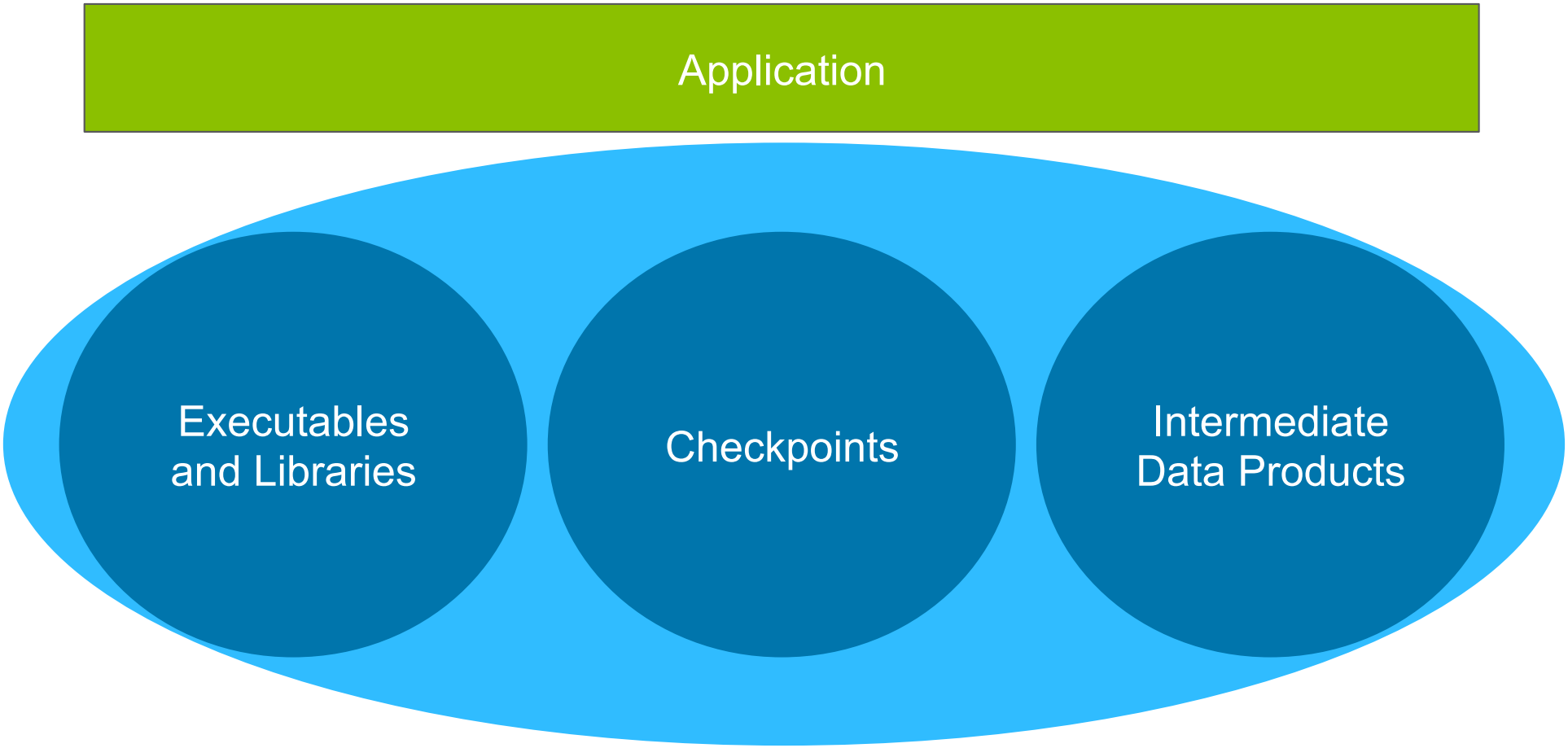
WHAT COMES NEXT?

- Assumptions
  - New layers in storage hierarchy, lower latencies
  - Storage resources will be highly contended for
  - No “holy grail” emerges that solves everyone’s problems

- Alternative model to the “PFS for data management”
  - Multiple services employed for different classes of data
  - Specialization for scalability/efficiency/productivity
  - In some cases, co-design with applications

SPECIALIZATION IN DATA MANAGEMENT

SPECIALIZATION BY CLASSES OF DATA



MANAGING EXECUTABLES AND LIBRARIES

Dynamic libraries are a clean class of data to treat separately.

- Characteristics:
  - Can assume data doesn't change during runtime
  - High degree of sharing across application processes
  - No need for redundancy in service (original stored elsewhere)
- Opportunities:
  - Dramatic reduction in parallel file system traffic
  - Stripping of libraries on load
  - Pre-staging of data (with scheduler integration)
- SPINDLE is a great example of how to manage this data.

Frings, Wolfgang, et al. "Massively parallel loading." ICS 2013, June 2013.

MANAGING CHECKPOINTS

- Characteristics
  - Typically (still) bulk synchronous
  - Write once, often not read
- Opportunities
  - Latency hiding
  - **Leveraging multiple layers of storage**
  - **Adjusting rate/placement to match fault rates**
- Fault Tolerant Interface (FTI)
  - Simple “snapshot” abstraction
  - Manages all the layers for the user

```
int main(int argc, char **argv) {

  MPI_Init(&argc, &argv);
  FTI_Init("conf.fti",
    MPI_COMM_WORLD);

  double *grid;
  int i, steps=500, size=10000;
  initialize(grid);
  FTI_Protect(0, &i,   1,   FTI_INTG);
  FTI_Protect(1, grid, size,FTI_DFLT);

  for (i=0; i<steps; i++) {
    FTI_Snapshot();
    kernel1(grid);
    kernel2(grid);
    comms(FTI_COMM_WORLD);
  }

  FTI_Finalize();
  MPI_Finalize();
  return 0;
}
```

Local Storage: SSD, NVM

Partner Copy: Ckpt. Replication

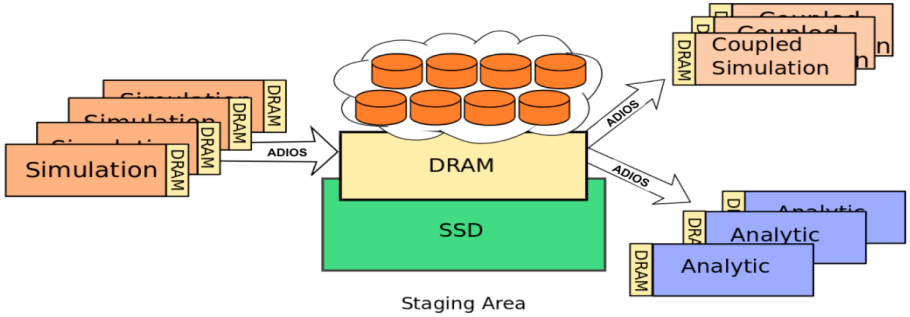
RS Encoding: Ckpt. Encoding

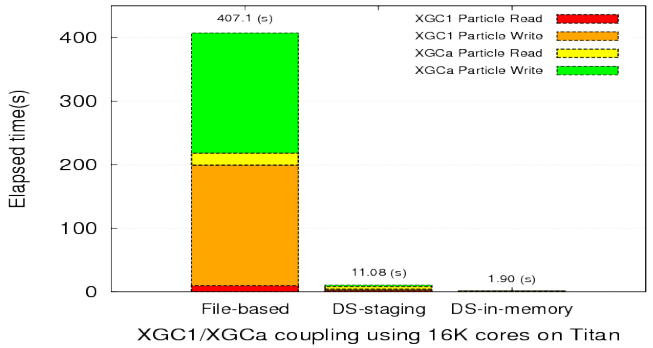
File System: Classic Ckpt.

L. Bautista-Gomez et al. "FTI: high performance fault tolerance interface for hybrid systems." SC 2011. November 2011.
S. Di et al. "Optimization of multi-level checkpoint model for large scale HPC applications." IPDPS 2014. 2014.

MANAGING INTERMEDIATE DATA PRODUCTS

- Characteristics:
  - Data leaves application but not the system
  - Variety of different data abstractions
  - Producer-consumer model is common
- Opportunities:
  - Exploiting locality
  - Avoiding data movement off system
  - More efficient synchronization

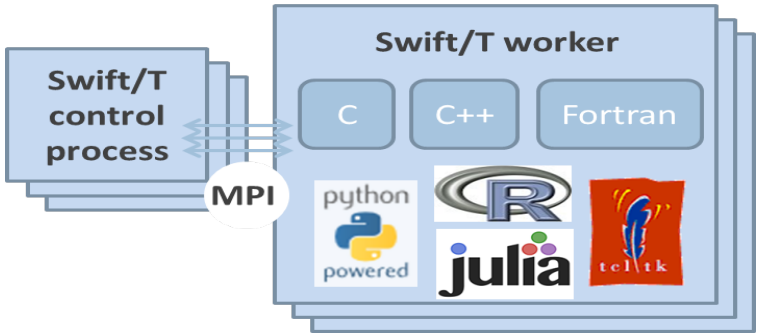




Impact of coupling via ADIOS/DataSpaces on XGC1/XGCa fusion application.
Material from S. Klasky (ORNL).

C. Docan et al. "DataSpaces: an interaction and coordination framework for coupled simulation workflows." *Cluster Computing* 15.2 (2012).
C. Ulmer. "Leveraging In-Memory Key/Value Stores in HPC: Kelpie." Salishan 2013, April 2013.

SPECIALIZATION FOR MANY-TASK WORKFLOW

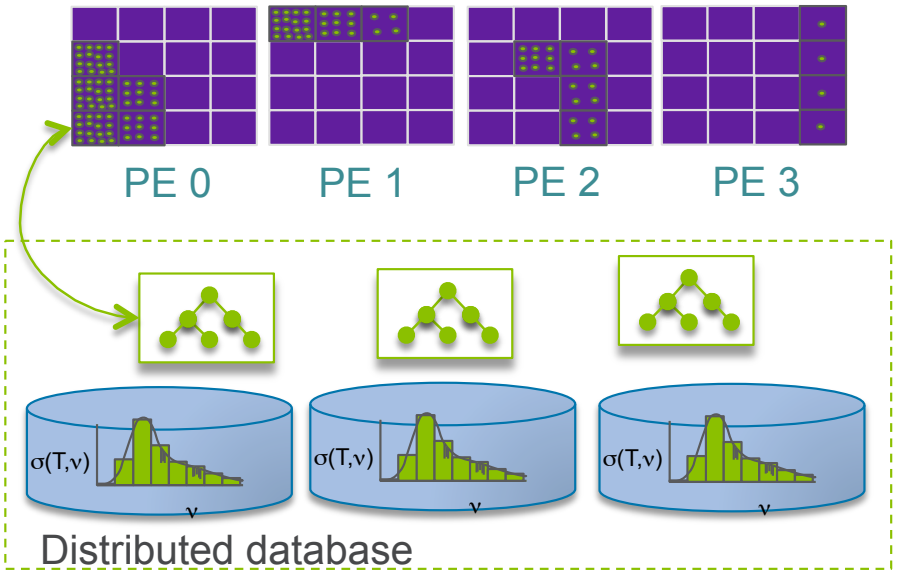


- Swift script controls execution – generates an ADLB program (see below)
  - Tasks can be basically anything (e.g., MPI code)
  - Data dependencies are emitted as run proceeds
- Asynchronous Dynamic Load Balancer (ADLB) manages data and work
  - Distributed, data-dependent work queue
  - Work units have (optional) priorities, types, and **locality constraints**
  - Enables heuristic, coarse-grained data-aware scheduling, mixing user control and automatic decisions
- Applied in materials science, power grid, etc.
  - E.g., transforming TBs of X-ray data from the Advanced Photon Sources, streaming to compute nodes at 100 GB/s

J. Wozniak et al. "Swift/T: large-scale application composition via distributed-memory dataflow processing. CCGrid 2013.
E. Lusk et al. “More Scalability, less pain: a simple programming model and its implementation...” SciDAC Review, 2010.
F. Duro et al. “Flexible data-aware scheduling for workflows over an in-memory object store”. CCGrid 2016.

EQUATIONS OF STATE, OPACITIES, AND DISTRIBUTED DATABASES

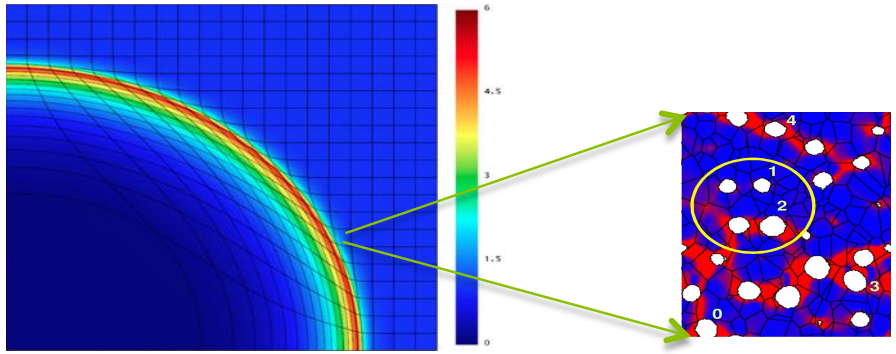
- Current generation applications often replicate material properties across all memories in the system
  - Data is often tabular, indexed by model parameters
  - Tradeoff between representative physics of the material and performance / memory utilization
- Distributed database can hold much larger table (billions of values) imported from external storage



Example: Implicit Monte Carlo simulation looking up discrete opacity values in distributed database.

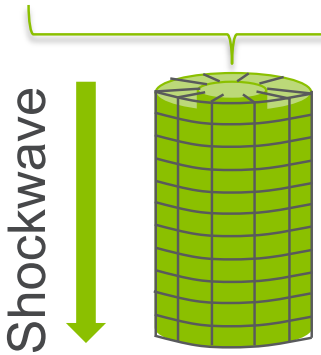
RB Lowrie and TS Haut, "Reconstructing opacities for multigroup thermal radiative transport," LA-UR-14-24608, 2014.

CONTINUUM MODEL COUPLED WITH VISCOPLASTICITY MODEL



Lulesh continuum model:
- Lagrangian hydro dynamics
- Unstructured mesh

Viscoplasticity model [1]:
- FFT based PDE solver
  - Structured sub-mesh

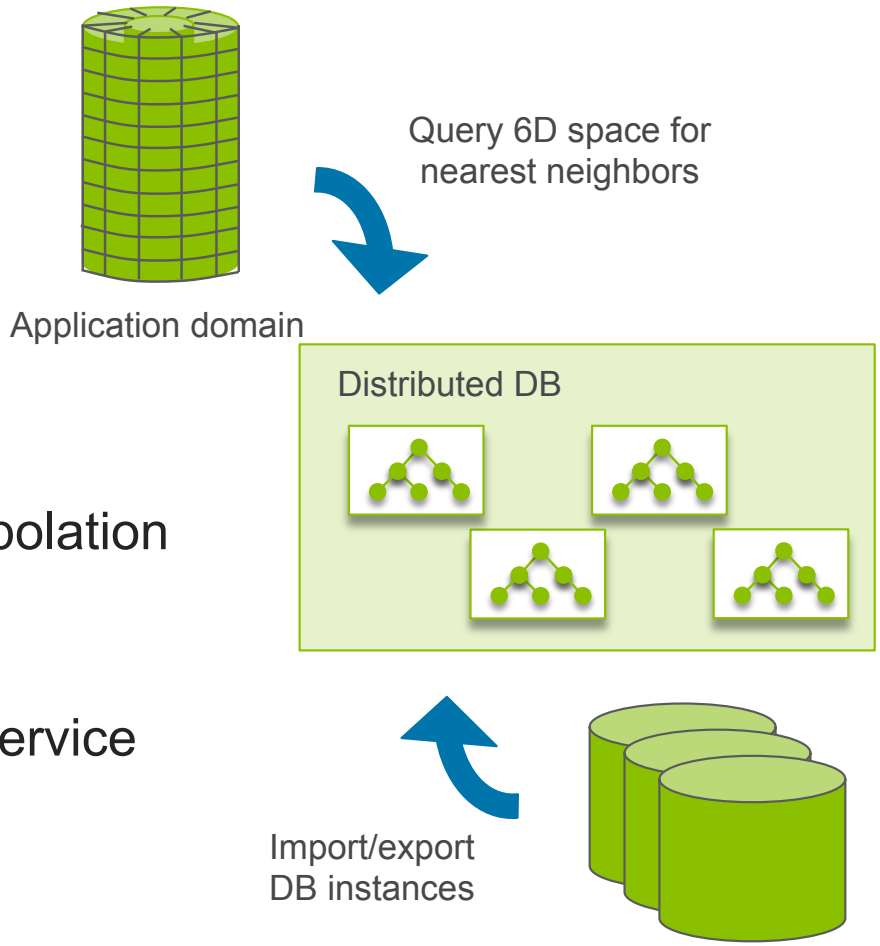


- Future applications are exploring the use of multi-scale modeling
- As an example: Loosely coupling continuum scale models with more realistic constitutive/response properties
  - e.g., Lulesh from ExMatEx
- Fine scale model results can be cached and new values interpolated from similar prior model calculations

R. Lebensohn et al, Modeling void growth in polycrystalline materials, Acta Materialia, http://dx.doi.org/10.1016/j.actamat.2013.08.004.

CO-DESIGNING A FINE SCALE MODEL DATABASE

- Goals
  - Minimize fine scale model executions
  - Minimize query/response time
  - Load balance DB distribution
- Approach
  - Start with a key/value store
  - Distributed approx. nearest-neighbor query
  - Data distributed to co-locate values for interpolation
  - Import/export to persistent store
- Status
  - Mercury-based, centralized in-memory DB service
  - Investigating distributed, incremental nearest-neighbor indexing



ENABLING DATA SERVICES

ROB ROSS, PHILIP CARNS, KEVIN HARMS, JOHN JENKINS, AND SHANE SNYDER	Argonne National Laboratory
GARTH GIBSON, CHUCK CRANOR, AND QING ZHENG	Carnegie Mellon University
JEROME SOUMAGNE AND JOE LEE	The HDF Group
GALEN SHIPMAN AND BRAD SETTLEMYER	Los Alamos National Laboratory

Argonne NATIONAL LABORATORY

	Provisioning	Comm.	Local Storage	Fault Mgmt. and Group Membership	Security
ADLB *Data store and pub/sub.*	MPI ranks	MPI	RAM	N/A	N/A
DataSpaces *Data store and pub/sub.*	Indep. job	Dart	RAM (SSD)	Under devel.	N/A
DataWarp *Burst Buffer mgmt.*	Admin./ sched.	DVS/ lnet	XFS, SSD	Ext. monitor	Kernel, lnet
FTI *Checkpoint/restart mgmt.*	MPI ranks	MPI	RAM, SSD	N/A	N/A
Kelpie *Dist. in-mem. key/val store*	MPI ranks	Nessie	RAM (Object)	N/A	Obfusc. IDs
SPINDLE *Exec. and library mgmt.*	Launch MON	TCP	RAMdisk	N/A	Shared secret

OUR GOAL

Enable composition of data services for DOE science and systems

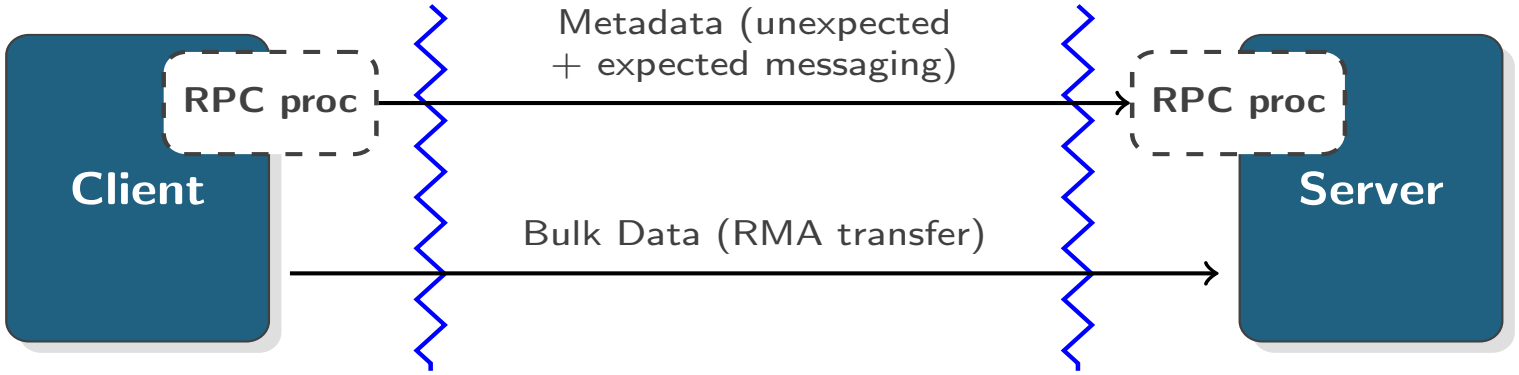
- Application-driven
  - Identify and match to science needs
  - Traditional data roles (e.g., checkpoint, data migration)
  - New roles (e.g., equation of state/opacity databases)
- Composition
  - Develop/adapt building blocks
    - Communication
    - Concurrency
    - Local Storage
    - Resilience
    - Authentication/Authorization
  - Enable rapid development of specialized services

COMMUNICATION: MERCURY

https://mercury-hpc.github.io/

Mercury is an RPC system for use in the development of high performance system services. Development is driven by the HDF5 Group with Argonne participation.

- Portable across systems and network technologies
- Builds on lessons learned from IOFSL, Nessie, Inet, and others
- Efficient bulk data movement to complement control messages

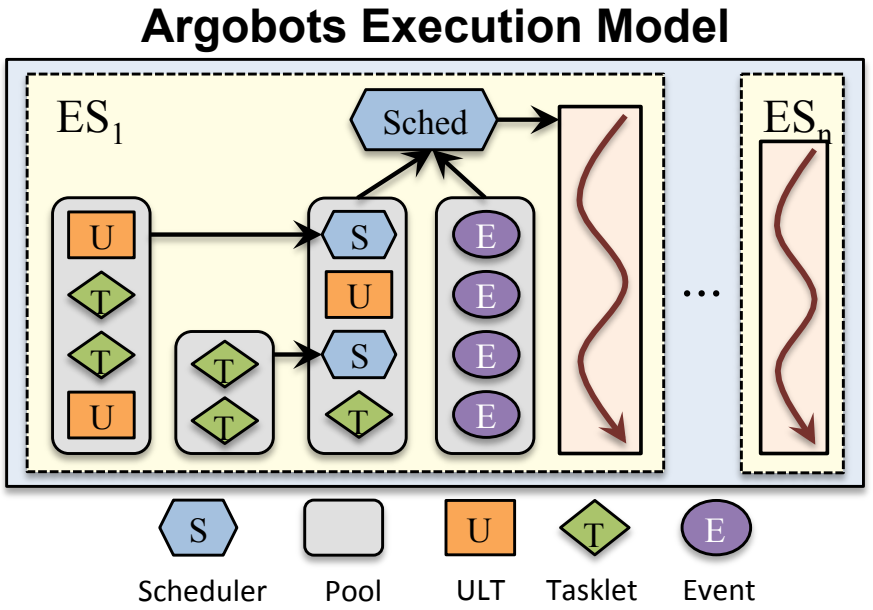


CONCURRENCY: ARGOBOTS

https://collab.cels.anl.gov/display/argobots/

Argobots is a lightweight threading/tasking framework.

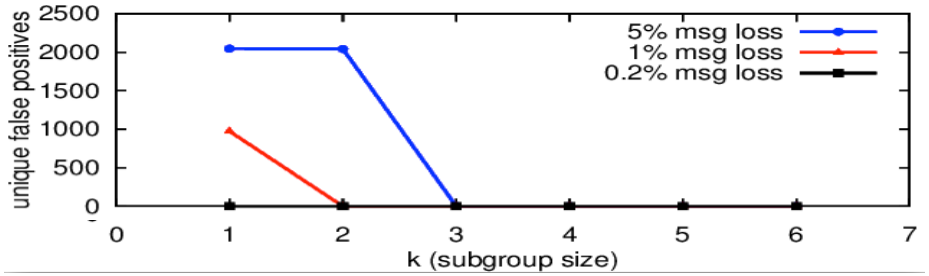
- Features relevant to I/O services:
  - Flexible mapping of work to hardware resources
  - Ability to delegate service work with fine granularity across those resources
  - Modular scheduling
- We developed asynchronous bindings to:
  - Mercury
  - LevelDB
  - POSIX I/O
- Working with Argobots team to identify needed functionality (e.g., idling)



GROUP MEMBERSHIP

Lots of work in this space already.

- Gossip-based detection
  - Scalable, distributes the comm. load
  - SWIM protocol is one example, rolls membership in with detection
  - Could introduce jitter…
- Vendors could provide an “oracle” for specific classes of faults
  - Won’t necessarily know your service is misbehaving
- Replicated state machine for consistent view of membership
  - PAXOS, RAFT, Corfu



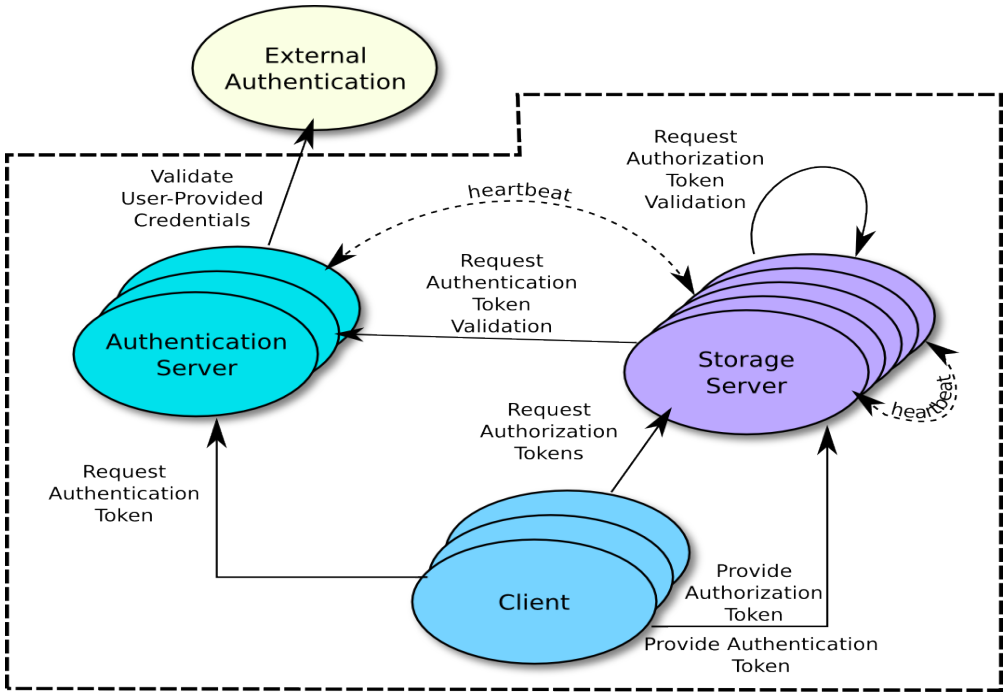
SWIM protocol with 2K nodes, 30 minutes of simulated time. Subgroup size determines number of peers that follow up on a failed ping.

A. Das et al. “SWIM: Scalable weakly-consistent infection-style process group membership protocol.” DSN ’02. 2002.
D. Ongaro et al. "In search of an understandable consensus algorithm.” USENIX ATC 14. 2014.

AUTHENTICATION AND AUTHORIZATION

Services intending to replace parallel file systems must provide (scalable) access control.

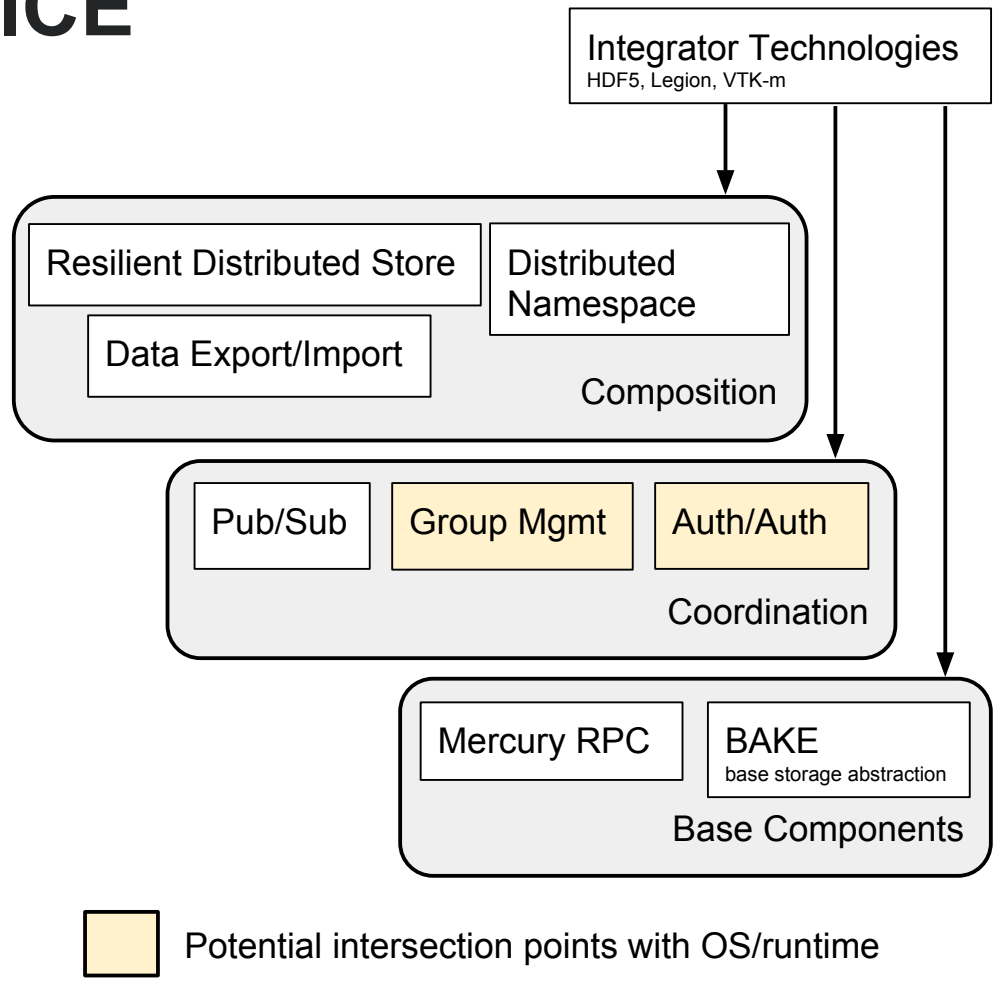
- Integrate with external authentication (Kerberos, LDAP)
- Capability-based approach
  - Caching, delegation to improve scalability
- Building off LWFS work and follow-on activities with L. Ward (SNL) and R. Brooks (Clemson)
  - Mercury prototype



R. Oldfield et al. "Lightweight I/O for scientific applications." Cluster 2006, 2006.

ENABLING A NEW SERVICE ECOSYSTEM

- Provide the building blocks for the next generation of HPC services
- Toolkit of interoperating components
  - Solutions to hard problems
  - Integration with related tech.
- Lower the barrier of entry
  - Teams casually build new services
- Work with vendors



THE END OF PARALLEL FILE SYSTEMS?

- Data services are a natural step in the trend of composition and specialization seen in large scale application codes
  - Codes supported by a handful of services
- Need the community to foster a “data service ecosystem”
  - Need buy-in from all the interested parties!
- PFSes won’t be ended by this, they’ll be ended by object stores or some other persistent storage back-end
- Future directions
  - Traffic throttling?
  - More building blocks for resilient services?
  - QoS?

THIS WORK IS SUPPORTED BY THE DIRECTOR, OFFICE OF ADVANCED SCIENTIFIC COMPUTING RESEARCH, OFFICE OF SCIENCE, OF THE U.S. DEPARTMENT OF ENERGY UNDER CONTRACT NO. DE-AC02-06CH11357.