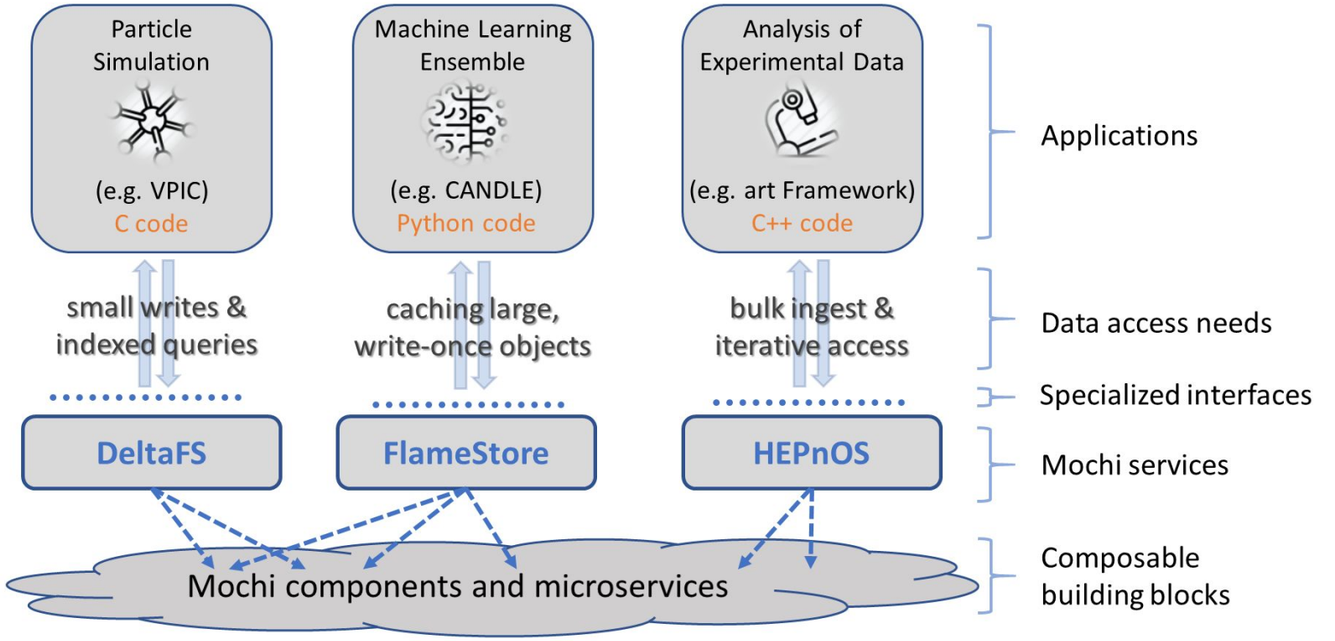

# PMIx Usage in Mochi Data Services

Shane Snyder
ssnyder@mcs.anl.gov
Argonne National Lab


PMIx BoF
ECP Annual Meeting ‘21

# Mochi background

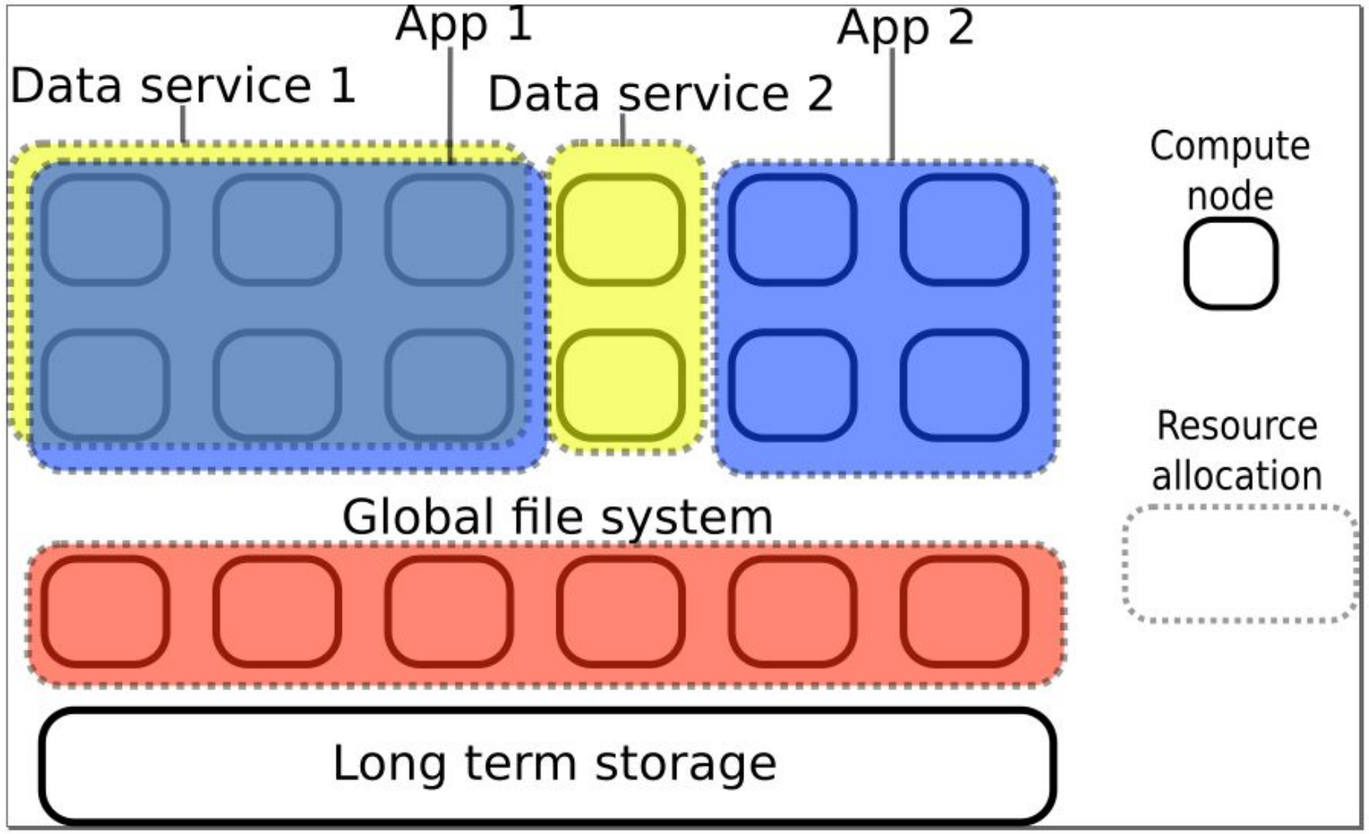
- Diverse DOE scientific computing applications have distinct data management needs
  - Simulation, data analytics, AI
- **Mochi project mission:** design methodologies and tools enabling rapid development of distributed data services in support of DOE science
- Focus is on *composability*: define common data management building blocks that simplify development of new services:
  - Communication and concurrency control; BLOB and key-val storage; group membership




https://www.mcs.anl.gov/research/projects/mochi/
https://github.com/mochi-hpc

# Mochi background

- Mochi data services are dynamically deployed across sets of nodes, potentially separate from applications
  - Bootstrapping mechanisms required for establishing service connectivity
- Services can grow/shrink, either as part of changing resource allocations or failures
  - Fault detection and group membership foundational to distributed data services



# Mochi Group Membership

- **Motivation**: Distributed systems frequently require a group membership service to reach agreement on the set of processes comprising the system, even in the face of process failures and changing resource allocations

- SSG (Scalable Service Groups): dynamic group membership building block for distributed Mochi services
  - Service group bootstrapping
    - Who are the initial participants of the group? What are their network addresses?
  - Fault detection and elasticity support
    - Have existing group members failed or explicitly left the group? Have new members joined the group?

# PMIx usage in SSG

- Service group bootstrapping
  - MPI and PMIx bootstrapping methods currently supported
  - For PMIx, this is implemented following the model of the “business exchange card” use case in full-modex mode
    - Business cards contain the Mochi endpoint address for each group member, which other group members use to build a group communication network
- Fault detection and elasticity support
  - SSG fault detection primarily provided via SWIM, a gossip-based group membership and fault detection protocol
  - We have augmented SSG fault detection to additionally ingest PMIx events indicating failure of SSG group members
    - Short-circuit SWIM protocol fault detection algorithm in the case of known failures

# It would be great if PMIx...

- ...was supported natively by the runtime environment on more production HPC systems (i.e., no need to deploy PRRTE)
  - MPI is a heavy-weight dependency for Mochi services, but very convenient since it is always available and simple to use
- ...was more heavily tested and reliable on production HPC systems
  - Our team has encountered PMIx/PRRTE bugs/regressions on numerous systems that make it difficult to commit to PMIx
  - CI testing on popular HPC systems would be really helpful for ensuring reliability
- ...offered interfaces for discovering and managing storage resources
  - An ability to discover storage resources and their performance characteristics could be very useful for selecting suitable storage resources for a Mochi service