Mochi: Composing Micro-Services to Build Application-Tailored Data Services

Matthieu Dorier
Argonne National Laboratory

Enabling Data Services for HPC - BoF@SC2018

Diversity in existing storage systems

- Data distribution and indexing methods
- Access semantics
- Transactions and locking
- Fault tolerance, replication

But they are not tailored to each application individually

Software Defined Storage

DOE project 2015-present

Composing HPC Microservices

- Formalize composition
- Unify single-process, multi-process, single-node, and multi-node designs
- Maximize efficient use of resources (network, storage)

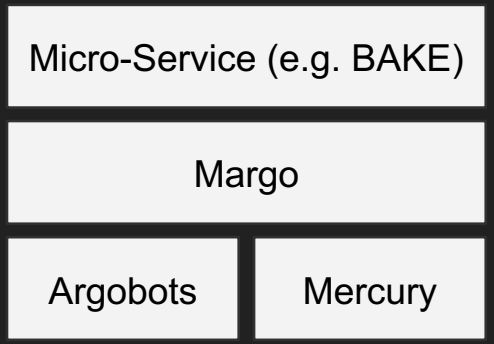
Common capabilities needed by data services

Runtime substrate
- RPC, RDMA
- Threading/Tasking

Core components
- Bulk storage management
- Key/Value storage
- Group membership
- Diagnostics and monitoring

Programmability/Expressiveness
- Embedded interpreters
- Wrappers (Python, C++, etc.)

Mochi micro-services

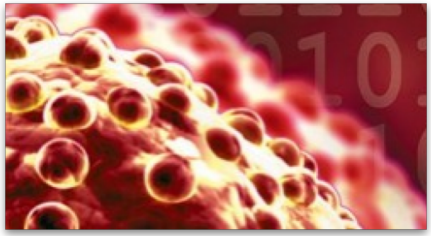


- Mercury: RPC/RDMA
- Argobots: Threading/Tasking
- Margo: Mercury+Argobots

Different users

=

Different needs



HEPnOS

Fast event-store for High Energy Physics experiments

Fermilab

HEPnOS: fast event-store for High-Energy Physics experiments

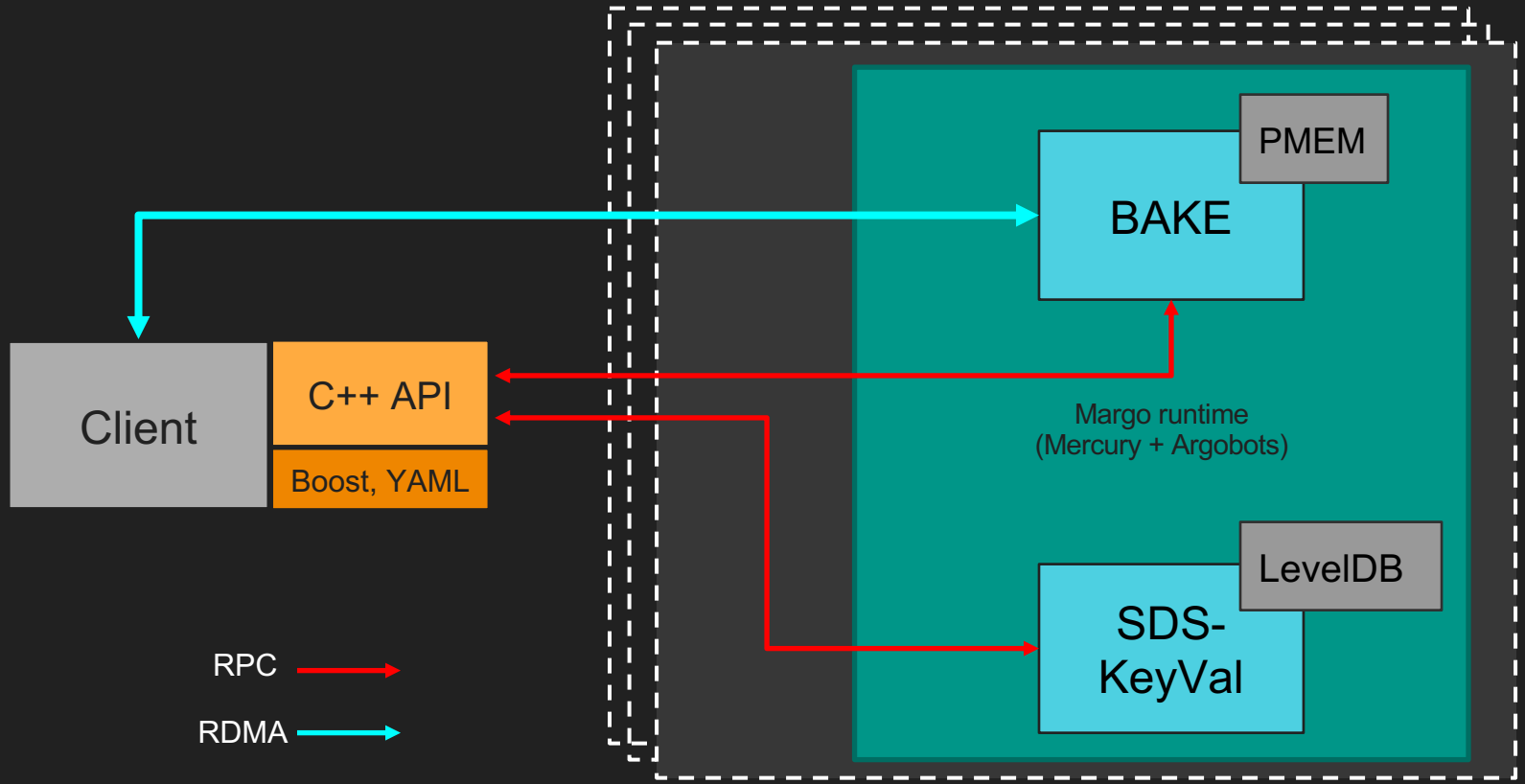
Needs

- Write-once-read-many
- Hierarchical namespace (datasets, runs, subruns)
- C++ API (serialization of C++ objects)

Components

- SDSKV: Stores key/values for metadata management
- BAKE: Manages RDMA data transfers to/from storage
- SSG: Group membership service

HEPnOS: fast event-store for High-Energy Physics experiments

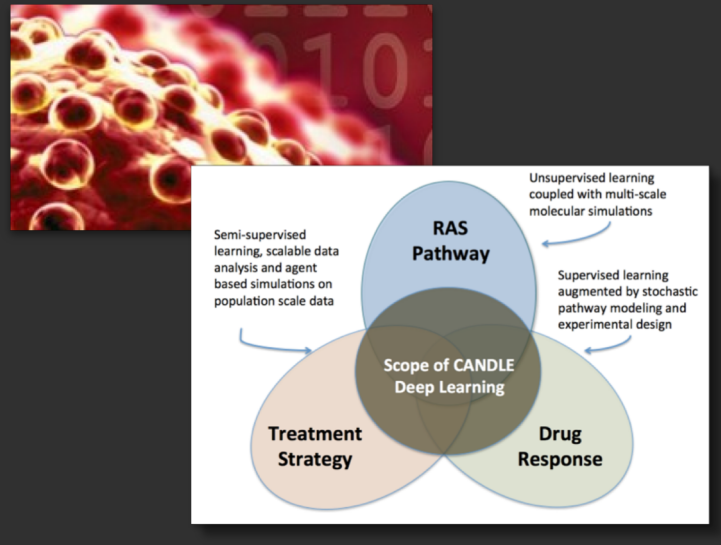


The hard questions…

- How do we deploy HEPnOS?
  - Mpirun (not ideal)
  - Investigating alternative deployment methods (e.g. PMIx)
- How do we handle resilience?
  - Data replication
  - Eventually: SSG for Group Membership (SWIM protocol)
- How is security provided?
  - Not provided currently. We will need to support jobs connecting to other jobs.
  - Could be supported by the platform, or by the storage system

FlameStore

A transient storage system for deep neural networks



Argonne NATIONAL LABORATORY

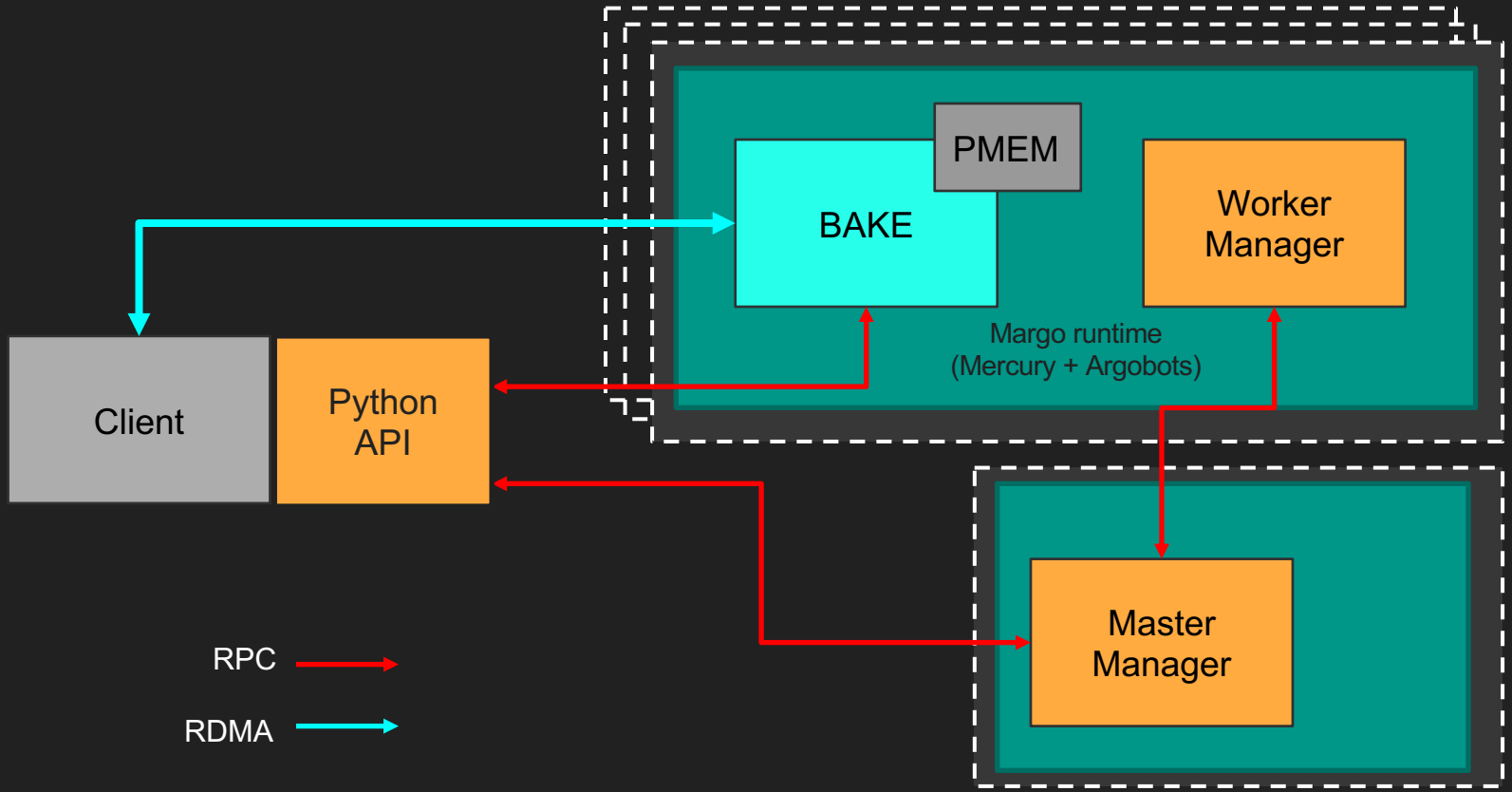
FlameStore: A transient storage system for deep neural networks

- Write-once-read-many
- Flat namespace
- High level of semantics
- Python API (stores Keras models)

- Components used: MERCURY, ARGOBOTS, MARGO, BAKE, POESIE, and their Python wrappers

- Extra code: Python API, master and worker managers

FlameStore: A transient storage system for deep neural networks



The hard questions…

- How do we deploy FlameStore?
  - Mpirun of a Python script using MPI4PY
  - OR FlameStore integrated within a Swift/T workflow (using embedded python facility)
- How do we handle resilience?
  - No plan yet
- How is security provided?
  - Not needed