Distributed Monitoring and Management of Exascale Systems in the Argo Project

Swann Perarnau[1] Rajeev Thakur[1] Kamil Iskra[1] Franck Cappello[1]
Pete Beckman[1] Marc Snir[1] Martin Schulz[2] Henry Hoffmann[3]

Argonne National Laboratory

Lawrence Livermore National Laboratory

University of Chicago

2015/06/03

Current State of HPC Systems

Mira

Argonne's Current System. #5 on TOP500.

- Compute Nodes: 50 000.
- CPU Cores/node: 16
- Interconnect: 5D torus
- Power: 4.8 MW

Others/Next 5 years

- Compute Nodes: O(10 000)
- Cores/node: O(100)
- New Tech: NVRAM, Stacked Memory, ...

What to Expect of an Exascale System

Hardware

Projected for 2022–2025.

- Compute Nodes: O(100 000)
- CPU Cores/node: O(1000)
- Interconnect: $>$ 3 dimensions
- Power: $\sim$ 20 - 30 MW

Consequences

- High intra-node parallelism
- OS must integrate/abstract new technologies
- High failure rate
- Complex resource management

A Software Stack for Exascale: Argo

Argobots

Modern Runtime for high intra-node parallelism.

- User-level task and threading model
- Interactions with OS and Communication libraries

NodeOS

Linux with HPC specializations.

- Compute Containers: partitioning instead of isolation/sharing.
- New scheduler and memory subsystems.

Argo (2)

Global Information Bus

High-level components for complex communications requirements.

- Pub-sub system, key-value store, ...
- Designed with failures and dedicated networks in mind.

GlobalOS

Distributed management of the entire system.

- Encapsulate configuration and policies inside group of nodes.
- Dedicated nodes for system management across the machine.

Enclaves?

A new abstraction inside the system

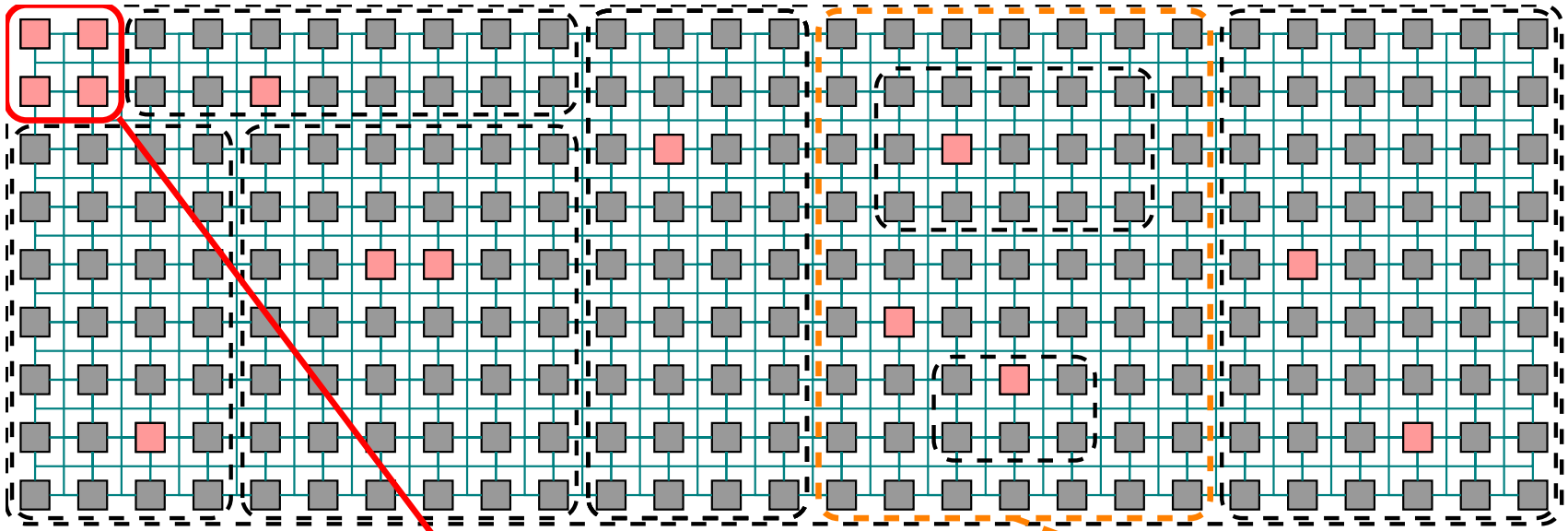
Group together, and manage as a whole, nodes configured the same way.

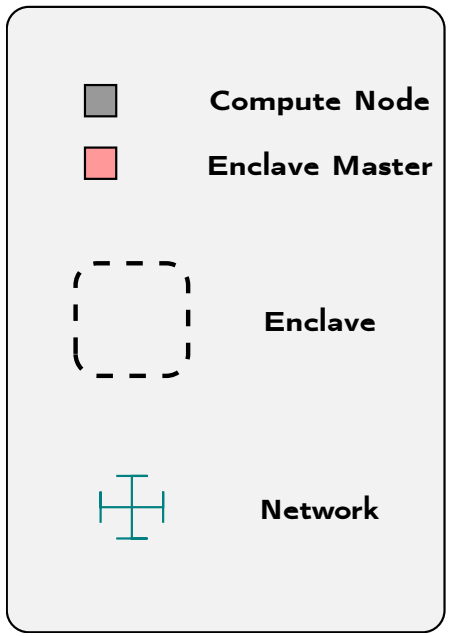
- Allow subdivisions, dedicate nodes to management of the enclave
- Allow users to reconfigure their enclaves.

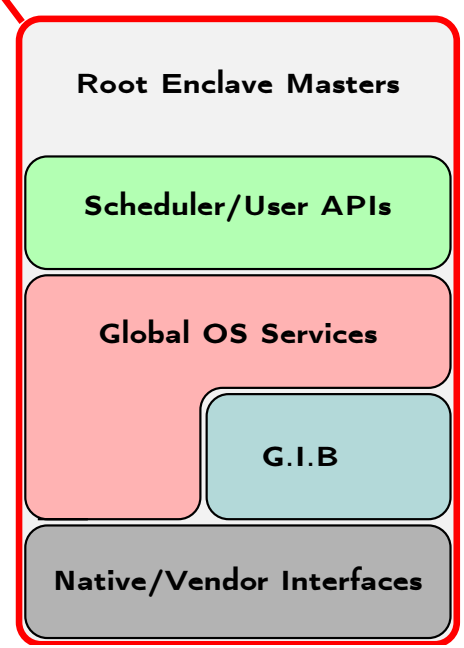
A hierarchy inside the system

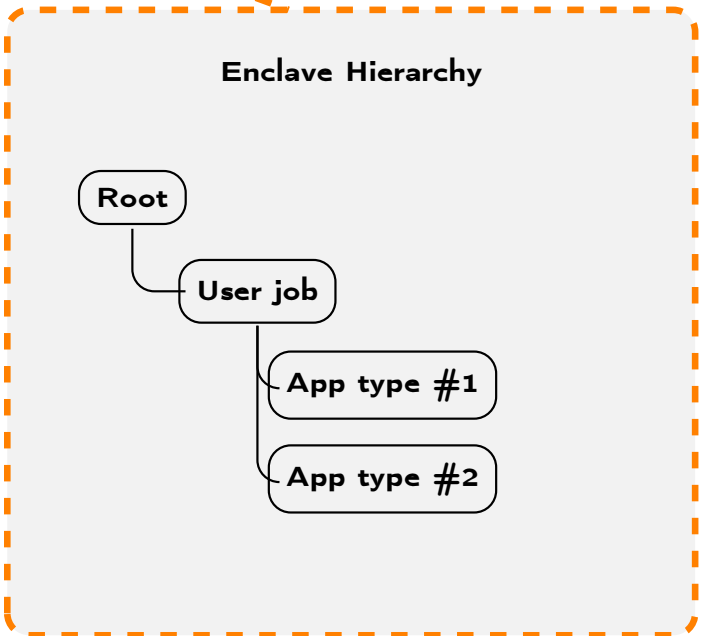
Manage enclaves as a tree.

- Level 1 is a user job, user can create additional levels.
- Enclave master only manage nodes at their level.




Compute Node
Enclave Master
Enclave
Network


Root Enclave Masters
Scheduler/User APIs
Global OS Services
G.I.B
Native/Vendor Interfaces


Enclave Hierarchy
Root
User job
App type #1
App type #2

A Control Bus over an Enclave Tree

Principles

Send commands across the hierarchy of enclaves.

- Fine-grained, configurable control located *close* to nodes.
- Keep some high-priority commands on the root.

Building Blocks

- Naming scheme: attach commands to a path in the enclave hierarchy.
- Dedicated Message Broker: runs on each node and route/filter/modify message on the fly.
- Priorities? Security? Isolation?

Distributed Power Management

Challenge

The power budget of the entire system must be respected at all time.

- Shutdown unused nodes.
- Slowdown under-used nodes.

How ?

Distribute monitoring and power control across all nodes.

- Measure data on all nodes.
- Enclave masters aggregate data and reacts to it.
- OS/Architecture provide fine-grained control over power (RAPL).
- What about runtime?

Managing Failures as Exceptions

Challenge

Failures will happen more frequently.

- How to react to them ?
- How to segregate failures to the smallest set of nodes ?

Principle

Consider failures as exceptions navigating the enclave tree.

- Let the closest master try recovery first.
- Escalate issues in the hierarchy, taking radical measures at the top.
- Let users configure the recovery system depending on its software.
- Restarting inter-dependant services?

Any questions?

This work was supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computer Research, under Contract DE-AC02-06CH11357.