ARGO: An Exascale Operating System and Runtime

Swann Perarnau (ANL), Rinku Gupta (ANL), Pete Beckman(ANL), Pavan Balaji (ANL), Cyril Bordage (UIUC), George Bosilca (UTK), Franck Cappello (ANL), Jack Dongarra (UTK), Daniel Ellsworth (UO, LLNL), Brian Van Essen (LLNL), Damien Genet (UTK), Roberto Gioiosa (PNNL), Maya Gokhale (LLNL), Thomas Herault (UTK), Henry Hoffman (UChicago), Kamil Iskra (ANL), Laxmikant Kale (UIUC), Gokcen Kestor (PNNL), Sriram Krishnamoorthy (PNNL), Edgar Leon (LLNL), Jonathan Lifflander (UIUC), Huiwei Lu (ANL), Allen Malony (UO), Nikita Mishra (UChicago), Kenneth Raffenetti (ANL), Barry Rountree (LLNL),Martin Schulz (LLNL), Sangmin Seo (ANL), Sameer Shende (UO), Marc Snir (ANL), Wyatt Spear (UO), Yanhua Sun(UIUC), Rajeev Thakur (ANL), Kazutomo Yoshii (ANL), Xuechen Zheng (UO), Huazhe Zhang (UChicago), Judicael Zounmevo (ANL)

Pete Beckman's talk

Objective

Exascale Challenges

- Heterogeneous, massively parallel compute nodes
- High Performance complex network topologies
- Constrained Resources (Power, I/O)
- Resilience, Fault and system management
- **Need the community to redesign and rethink Operating System and Runtime (OS/R) architectures for extreme-scale systems.**

What is ARGO?

- A system-wide global operating system → designed to manage node operating system, provides light-weight concurrency, system-wide power control, runtime resource management, resiliency and fault management
- One of the 3 projects funded under the Department of Energy ExaOSR initiative

Core Ideas

ENCLAVES

- Groups of nodes share the same configuration
- Enclave-specific *Master* node handles management of that enclave
- Enclaves in their lifetime can change in size and be recursively divided into sub enclaves
- A *Root* enclave exists for the global system

DISTRIBUTED MANAGEMENT

- Enclaves managed hierarchically and control distributed across *masters*
- Masters of parent enclaves have priority over masters of children
- Reaction to events (failures, environment/configuration changes) are distributed across *masters*;
- Privileged operations (admin, shutdown) are located on root

ARGO

RESOURCE PARTIONING

- Each node resource is partitioned at fine-grained level and given to user apps exclusively
- System services limited to a few dedicated cores of a node
- Custom, HPC-focused, memory, scheduler policies are available
- Similar interface to Docker/Rocket containers with less overhead

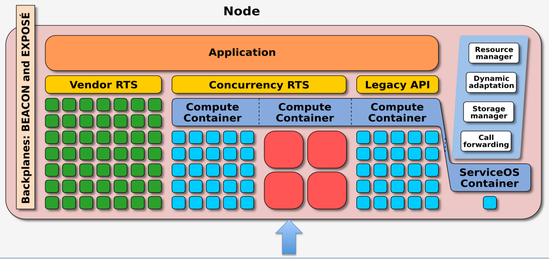
COMMUNICATION BACKPLANES

- Part of the Global Information Bus (GIB)
- BEACON Backplane provides scalable communication using publish-subscribe model and is used by all layers of ARGO software to exchange information
- EXPOSÉ Backplane provides performance introspection, in situ analysis and feedback mechanisms

ARGOBOTS

- Efficient runtime systems to exploit the massive on-node parallelism
- A new low-level threading/tasking model that exposes hardware characteristics of exascale systems effectively
- Explore new libraries and high-level tasking frameworks that can take advantage of such low-level model

The ARGO Approach

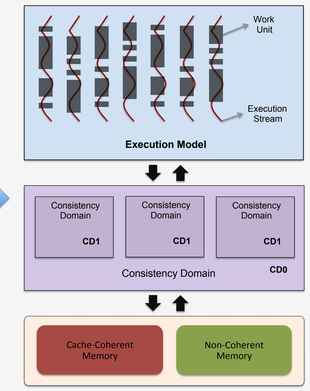


The Node OS

- Partition node resources for exclusive use
- Transparently incorporate NVRAM into memory hierarchy for applications
- Explore multi-level memory management and their impact on performance and energy
- Focus on optimizing interfaces for functionality that HPC applications actually use

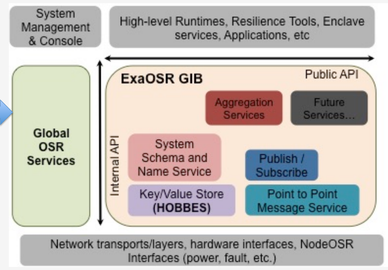
Concurrency

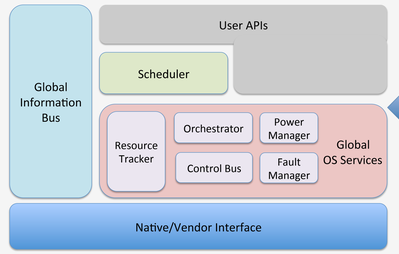
- Argobots: new low-level threading/ tasking model for exascale
- Explore high-level tasking frameworks (Cilk, PTGE) that exploit low-level threading/communication models
- Investigate popular high-level programming models, e.g., Charm++, that are specialized in dynamic execution environments and can exploit the low-level threading and communication frameworks
- Techniques for Argobots interoperability with PUT/GET model



The Global Information Bus (GIB)

- An infrastructure with components such as lightweight framework for sharing information across ARGO layers supporting both event and control notification (BEACON) as well as performance and energy introspection (EXPOSÉ)
- Provide high-level APIs: Publish/Subscribe, Key-Value Store (*from HOBBES ExaOSR project*), Aggregation services and system schema and naming service





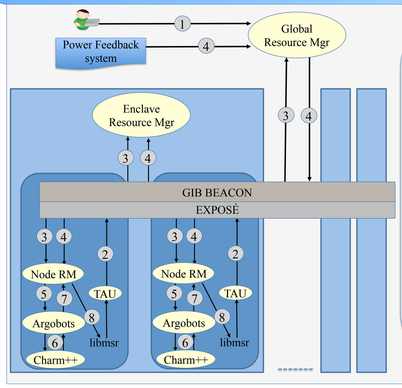
The Global Operating System

- Forms the core of the exascale machine
- Handles machine configuration, deployment, monitoring, management, and application launching across the system
- Responsible for measuring and controlling power across the machine (racks/nodes/cores; as well as enclaves/processes) → publish using BEACON

Status

- APIs, select software releases and publications can be found at : www.argo-osr.org
- Prototype implementation of system-wide Global OS → Built on top of Openstack services. Implementation uses bare metal provisioning and provides enclave creation and tracking, configuration of system services and job launching
- Distributed enclave and system-wide power management algorithms are a part of Global OS
- Prototype implementation of BEACON (on EVPATH and RIAK KVS)
- NodeOS provides partitioning of CPU and memory resources, a prototype implementation of the compute containers and custom scheduling policy for modern HPC runtimes
- Techniques to exploit NVRAM using DI-MMAP will help transparently incorporate NVRAM in memory hierarchies for applications
- Successfully demonstrated Argobots integration with several programming models: MPI, OpenMP, Charm++, Cilk, PTGE
- Collaboration with RIKEN in Japan led to highly scalable OpenMP for nested and irregular loops/tasks on top of Argobots
- Initial Argobots and Argobots+MPI prototype implementation completed. Development of Cilk + Argobots in progress

Managing Power in ARGO



1. User submits several jobs which are launched in their enclaves by Global OS
2. TAU software monitors sensors and publishes this info through BEACON
3. Various components receive this information
4. Global OS decides to reduce power in an enclave and publishes a request via BEACON
5. Node OSR components (such as NodeRM: Node Resource Manager) in that enclave receive this command and decide to shutdown a core and ask Argobots for approval

6,7. Argobots works with higher-level libraries and applications to shutdown an execution stream and inform NodeRM

8. NodeRM shuts down a core

Acknowledgments

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

U.S. DEPARTMENT OF ENERGY