Office of Science
U.S. DEPARTMENT OF ENERGY

OAK RIDGE NATIONAL LABORATORY

MANAGED BY UT-BATTELLE FOR THE DEPARTMENT OF ENERGY

High Availability for Ultra-Scale Scientific High-End Computing

Christian Engelmann [1,2] and Stephen L. Scott [1]

[1] Computer Science and Mathematics Division
Oak Ridge National Laboratory, Oak Ridge, USA

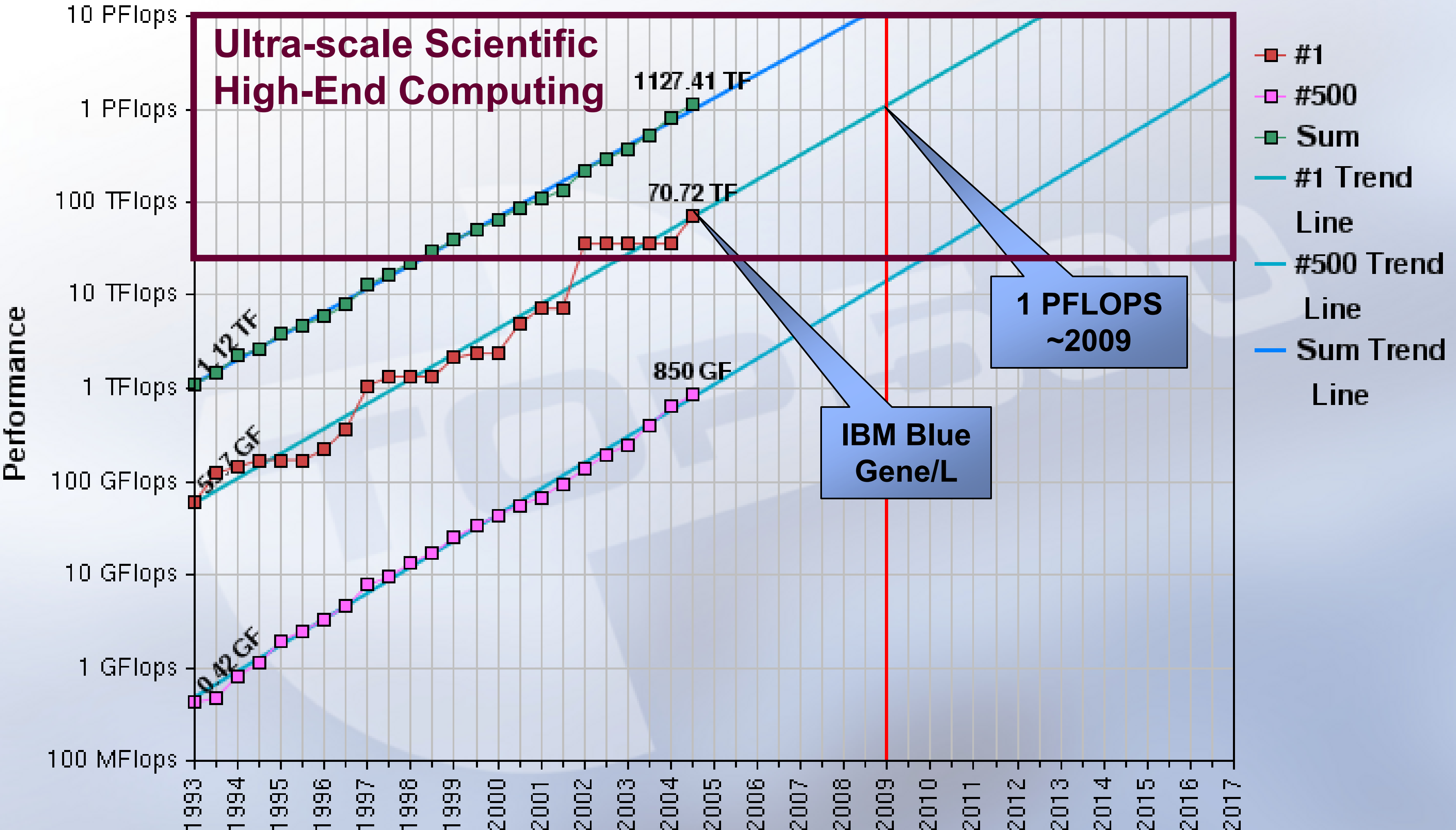
[2] Department of Computer Science
The University of Reading, Reading, UK

2nd International Workshop on Operating Systems, Programming Environments and Management Tools for High-Performance Computing on Clusters (COSET-2)
Cambridge, MA

Ultra-scale Scientific High-End Computing

- Next generation supercomputing.
  - Large-scale cluster, parallel, distributed and vector systems.
  - 131,072 processors for computation in IBM Blue Gene/L.
- Computationally and data intensive applications.
  - Many research areas: (multi-)physics, chemistry, biology…
  - Climate, supernovae (stellar explosions), nuclear fusion, material science and nanotechnology simulations.
- Ultra-scale = upper end of processor count (+5,000).
  - 25+ TeraFLOPS (20,000,000,000,000 FLOPS and more).
  - Cray X1 and XT3, IBM Blue Gene/L, etc.

Projected Performance Development



Availability of Current Systems

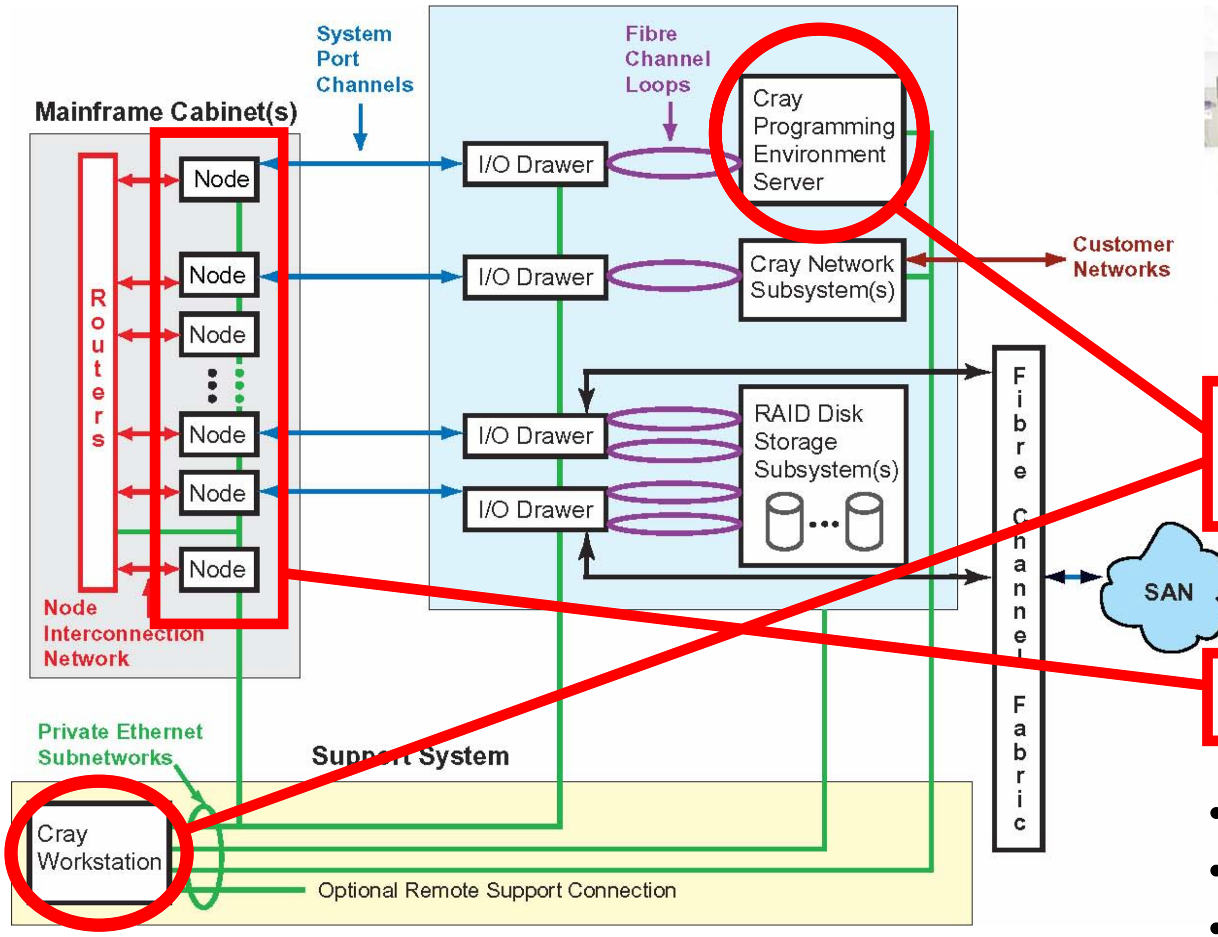
- Today's supercomputers typically need to reboot to recover from a single failure.
- Entire systems go down (regularly and unscheduled) for any maintenance or repair.
- Compute nodes sit idle while their head node or one of their service nodes is down.
- Availability will get worse in the future as the MTBI decreases with growing system size.
- *Why do we accept such significant system outages due to failures, maintenance or repair?*

Availability Measured by the Nines

9's	Availability	Downtime/Year	Examples
1	90.0%	36 days, 12 hours	Personal Computers
2	99.0%	87 hours, 36 min	Entry Level Business
3	99.9%	8 hours, 45.6 min	ISPs, Mainstream Business
4	99.99%	52 min, 33.6 sec	Data Centers
5	99.999%	5 min, 15.4 sec	Banking, Medical
6	99.9999%	31.5 seconds	Military Defense

- Enterprise-class hardware + Stable Linux kernel = 5+
- Substandard hardware + Good high availability package = 2-3
- Today's supercomputers = 1-2
- My desktop = 1-2

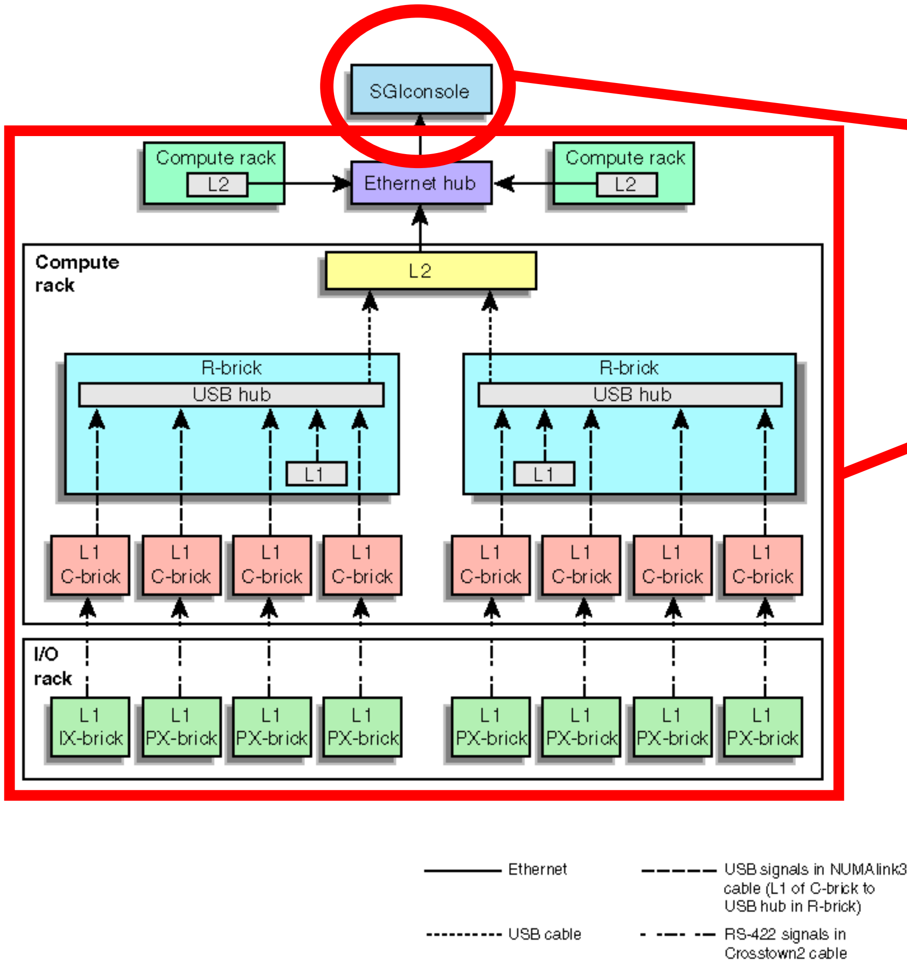
Vector Machines: Cray X1 (Phoenix)



- 512 Vector Processors
- 2 Tera Byte RAM
- 6,4 TFLOPS (29/Top500)

SSI Clusters: SGI Altix (Ram)

- 256 Itanium 2 Processors
- 2 Tera Byte RAM
- 1,5 TFLOPS (245/Top500)

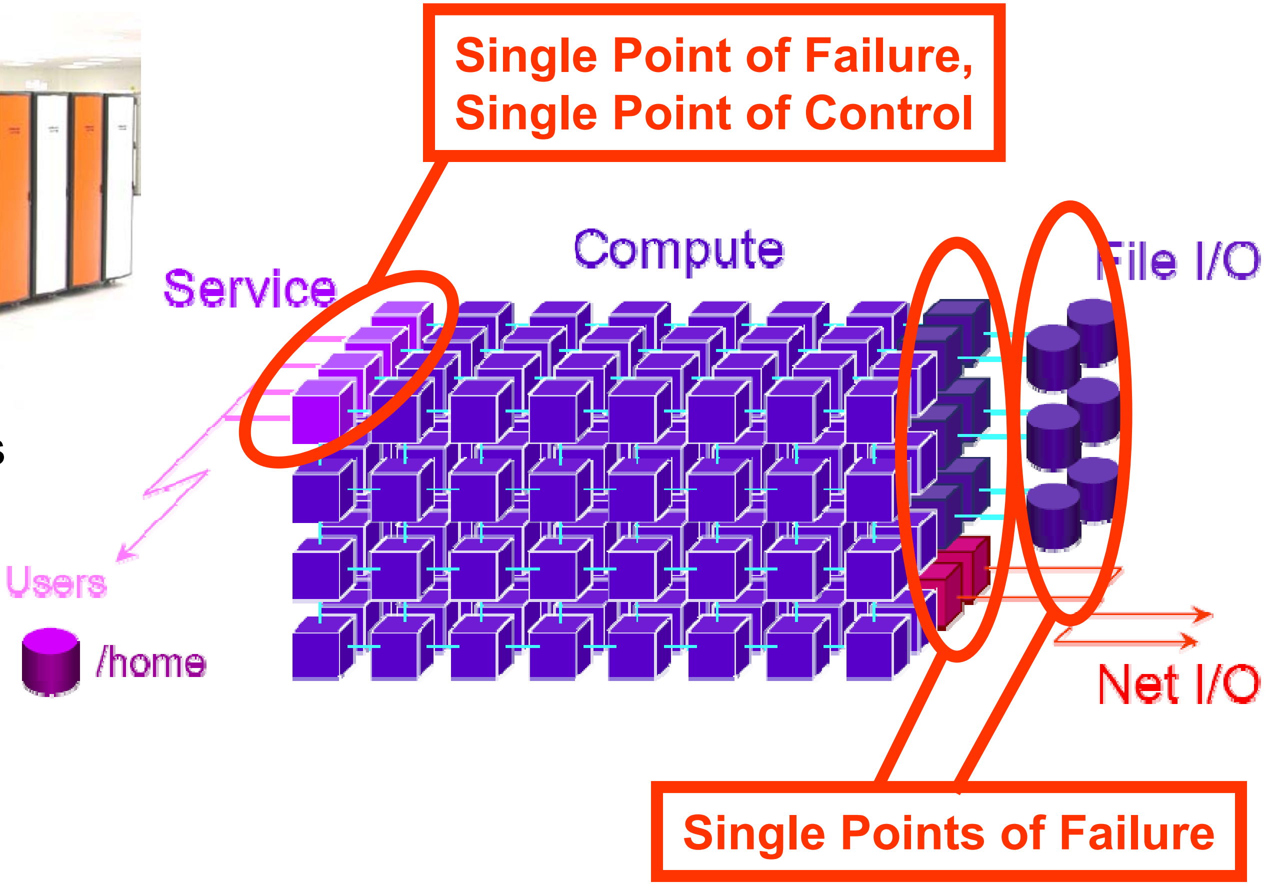


Single Point of Failure, Single Point of Control

Single Points of Failure

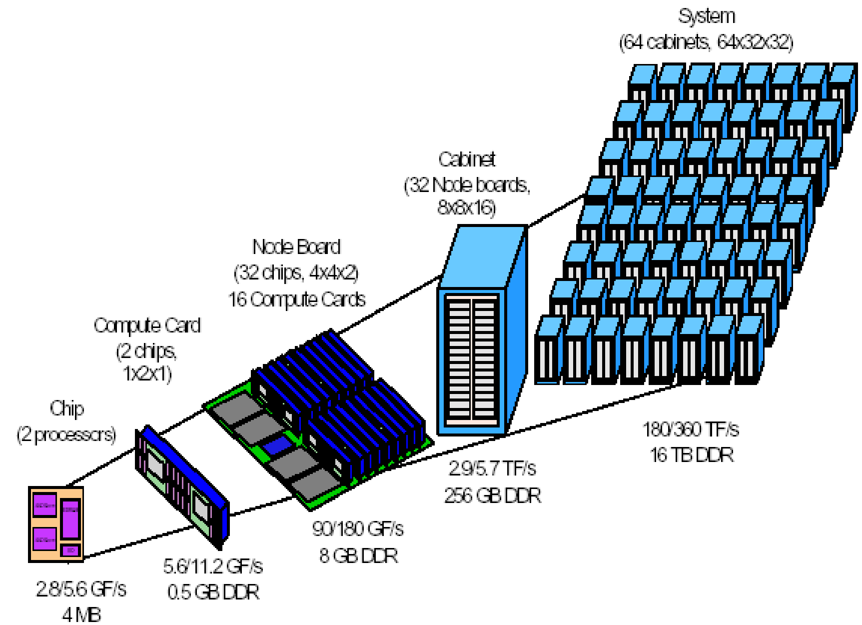
Clusters/MPPs: Cray XT3 (Jaguar)

- AMD Opteron Processors
- Installation in progress
-   25 TFLOPS in 2005
- 100 TFLOPS in 2006
- 250 TFLOPS in 2007



IBM Blue Gene/L

- 64K diskless nodes with 2 processors per node.
- 512MB RAM per node.
- Additional service nodes.
- 360 Tera FLOPS.
- Over 150k processors.
- Various networks.
- Operational in 2005.
- Partition (512 nodes) outages on single failure.
- MTBF = hours, minutes?



High Availability Methods

Active/Hot-Standby:

- Single active head node.
- Backup to shared storage.
- Simple checkpoint/restart.
- Rollback to backup.
- Idle standby head node(s).
- Service interruption for the time of the fail-over.
- Service interruption for the time of restore-over.
- Possible loss of state.

Active/Active (symmetric):

- Many active head nodes.
- Work load distribution.
- Symmetric replication between participating nodes.
- Continuous service.
- Always up-to-date.
- No restore-over necessary.
- Virtual synchrony model.
- Complex algorithms.

High Availability Technology

Active/Hot-Standby:

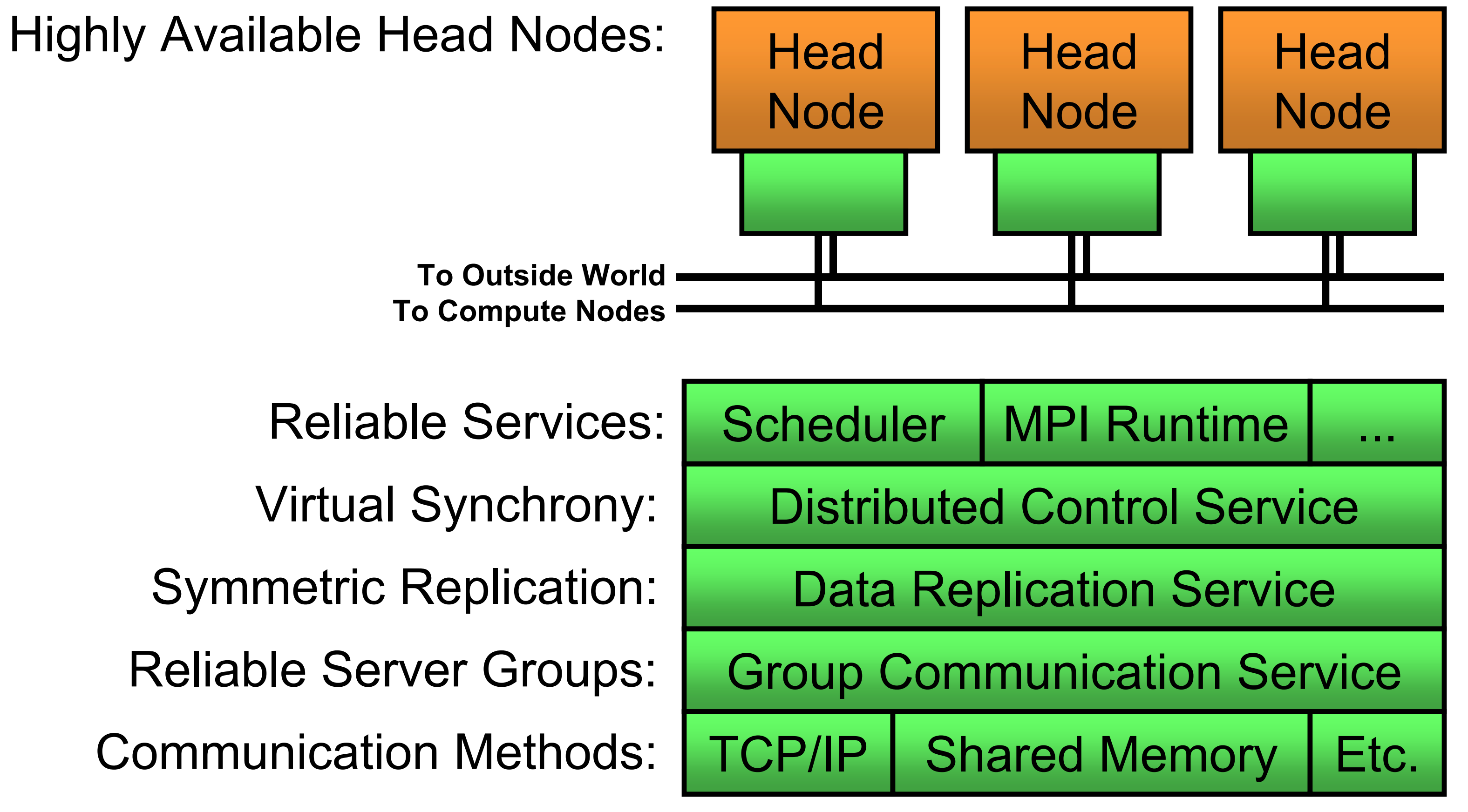
- HA-OSCAR with active/hot-standby head node.
- Similar projects: HA Linux
- Cluster system software.
- No support for multiple active/active head nodes.
- No application support.

Active/Active (symmetric):

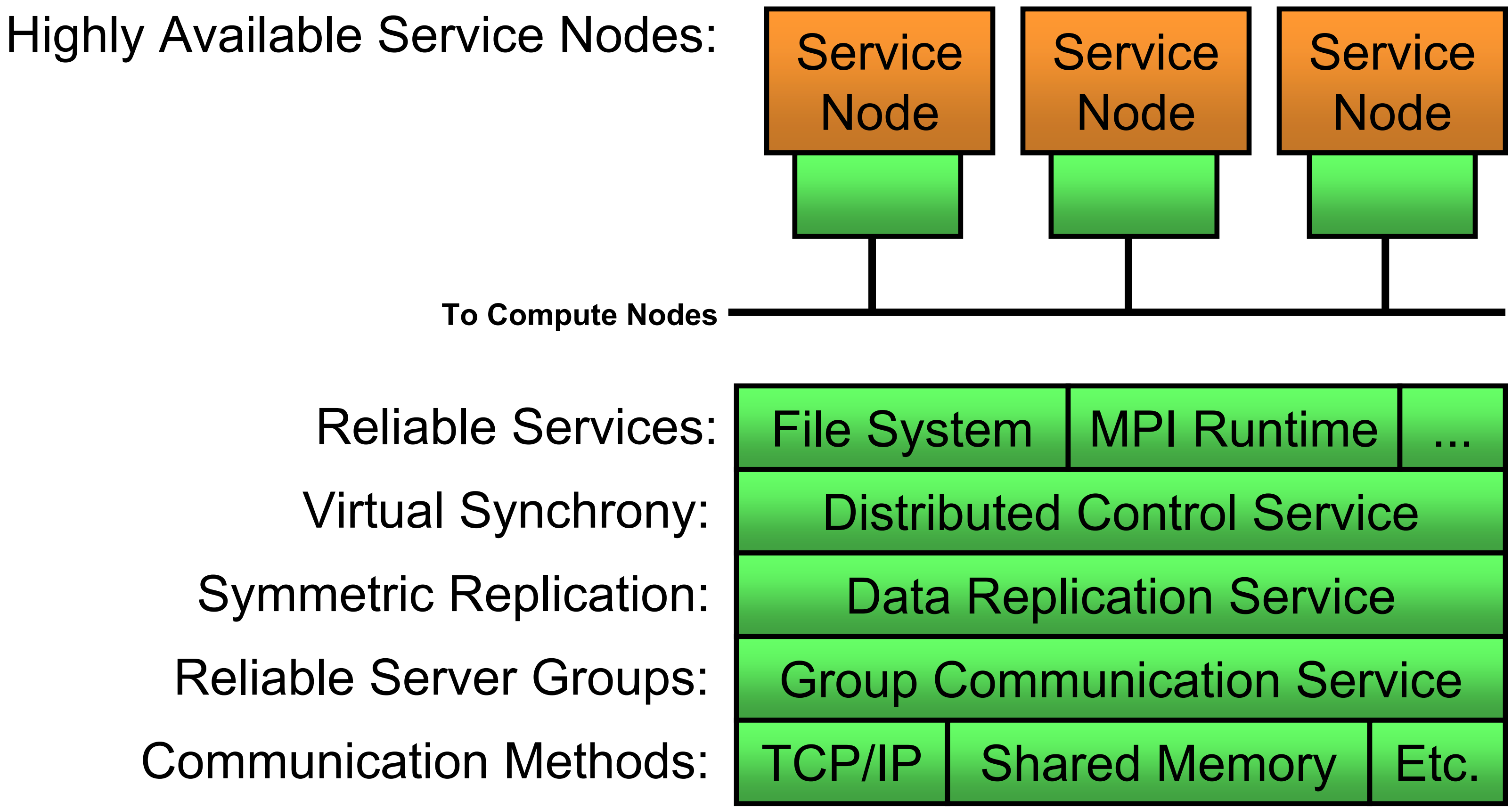
- HARNESS with symmetric distributed virtual machine.
- Similar projects: Cactus …
- Heterogeneous adaptable distributed middleware.
- No system level support.
- Solutions not flexible enough.

- *System-level data replication and distributed control service needed for active/active head node solution.*
- *Reconfigurable framework similar to HARNESS needed to adapt to system properties and application needs.*

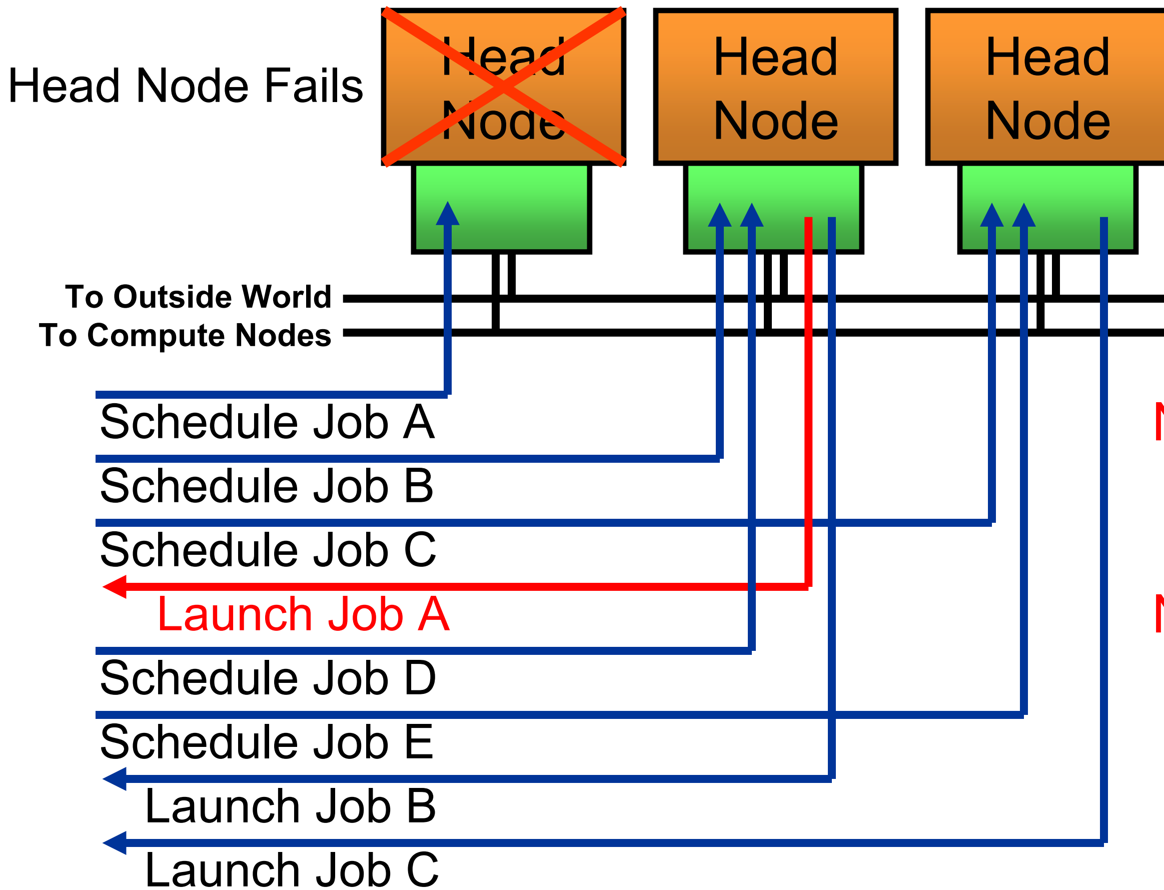
Modular HA Framework on Active/ Active Head Nodes



Modular HA Framework on Active/Active Service Nodes



Modular HA Framework on Active/ Active Head Nodes: Scheduler Example

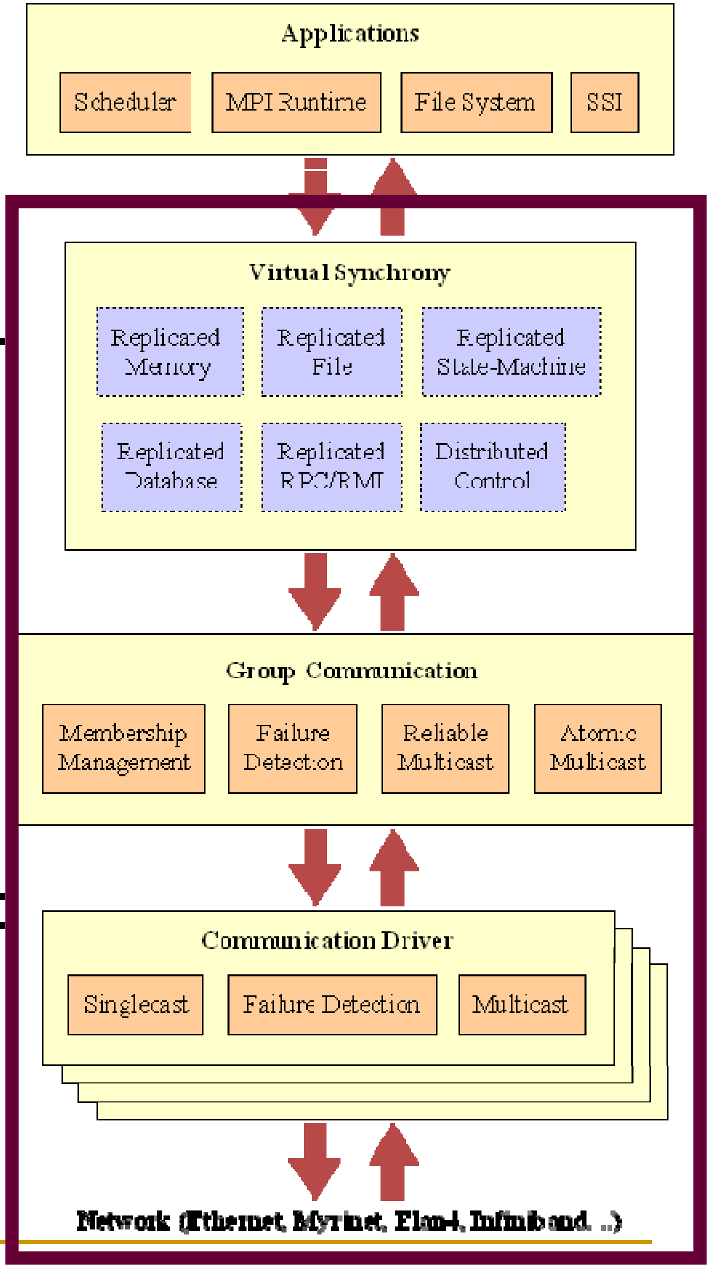


No Single Point of Failure

No Single Point of Control

HA Framework Design

- Pluggable component framework.
  - Communication drivers.
  - Group communication.
  - Virtual synchrony.
- Interchangeable components
- Adaptation to application needs.
- Adaptation to system properties.
- Partial reuse of existing solutions:
  - Harness lightweight kernel.
  - Open MPI comm. drivers.
  - Group comm. algorithms.



Many HA Framework Use Cases

- Active/Active and Active/Hot-standby process state replication for multiple head or service nodes.
  - Reliable system services, such as scheduler, MPI-runtime and system configuration/management/monitoring.
- Memory page replication for SSI clusters.
- Meta data replication for parallel/distributed FS.
- Super-scalable peer-to-peer diskless checkpointing.
- Super-scalable localized FT-MPI recovery.
- !!! No protection from Byzantine failures !!!

What is the relation to Horus?

- *Horus* is a group communication framework in C.
- It is based on stackable micro-protocol layers.
- It uses a fixed unified interface for all protocols.
- It is only free for research purpose.
- NDA for code and binary access, which can result in licensing issues later (e.g. reuse of code).
- The micro-protocol layer programming model will be supported in the HA framework among others.
- *Ensemble* is a free follow-on developed in OCaml.

What is the relation to Coyote?

- *Coyote* is a group communication framework in C.
- It is based on a reconfigurable event-driven micro-protocol state machine.
- It allows protocol adaptation (transition) at runtime.
- The micro-protocol state machine programming model will also be supported in the HA framework.
- *Cactus* is an ongoing follow-on project targeting real-time properties and configurable QoS.

What is the relation to …?

- Other group communication solutions exist.
- *Transis, Totem, Spread*, … (still counting, 80+ alg.).
- Most implement one specific algorithm.
- They were developed in the late 90s, while group communication models were still a research issue.
- None of these solutions provide full configurability.
- Their protocols and algorithms can be easily moved into the HA framework.
- The HA framework is based on a generalization of the developed solutions.

Conclusions

- High availability is a pressing issue in ultra-scale scientific high-end computing.
- Active/Hot-standby solutions exist, but rely on an idle backup server for failover.
- We proposed a flexible, component-based active/ active high availability framework.
- Partial reuse of existing solutions, such as lightweight runtime environments (RTEs).
- Group communication framework that avoids the usual “reinvention of the wheel”.

Office of Science
U.S. DEPARTMENT OF ENERGY

OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT-BATTELLE FOR THE DEPARTMENT OF ENERGY

High Availability for Ultra-Scale Scientific High-End Computing

Christian Engelmann and Stephen L. Scott

Network and Cluster Computing Group
Computer Science and Mathematics Division
Oak Ridge National Laboratory, Oak Ridge, USA

2nd International Workshop on Operating Systems, Programming Environments and Management Tools for High-Performance Computing on Clusters (COSET-2)
Cambridge, MA