Towards High Availability for High-Performance Computing System Services: Accomplishments and Limitations

Christian Engelmann[1,2]

[1] Oak Ridge National Laboratory, Oak Ridge, USA

[2] The University of Reading, Reading, UK

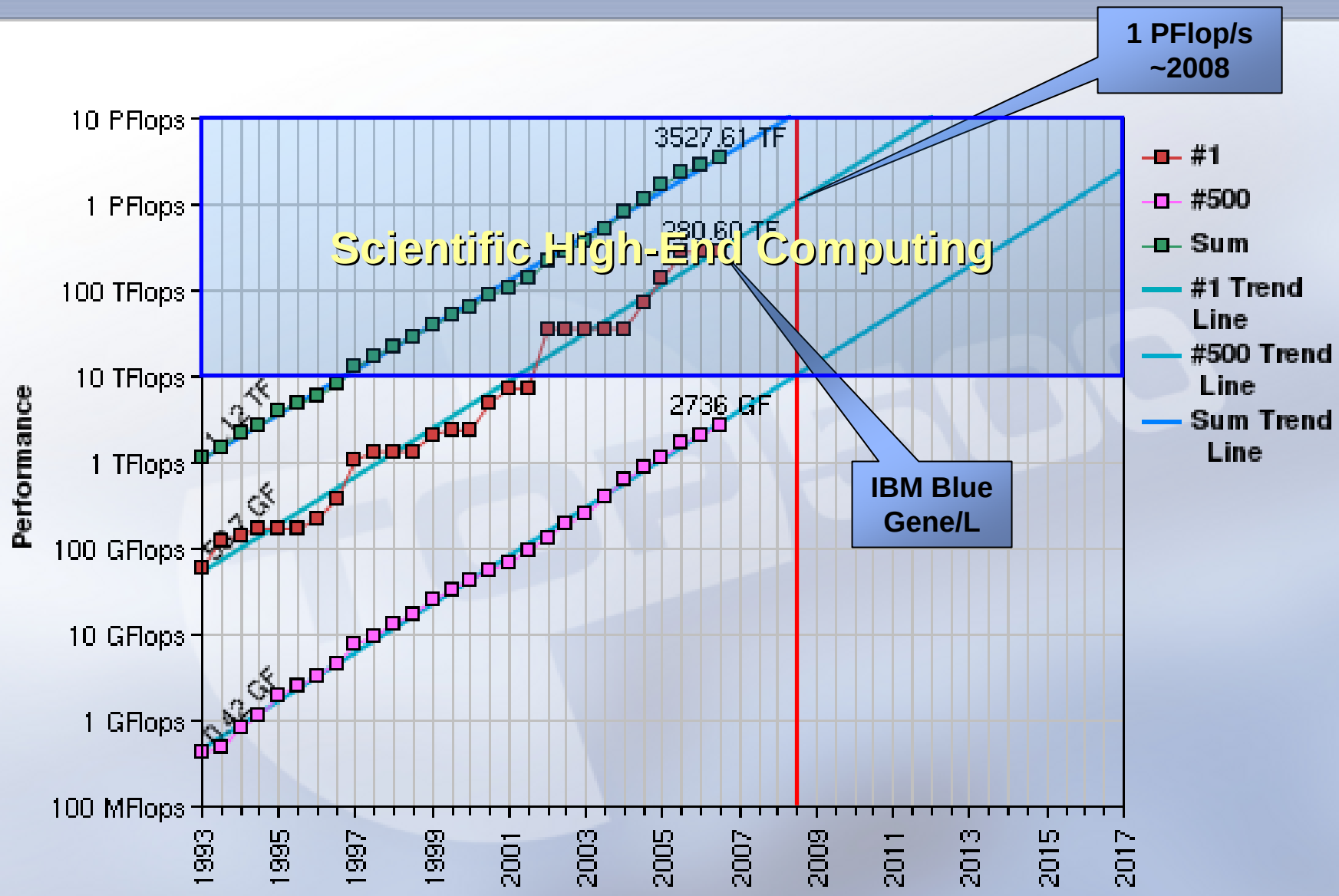
Talk Outline

- Scientific high-end computing (HEC)
- Availability deficiencies of today's HEC systems
- Projects and accomplishments overviews
- High availability (HA) models for services
- Developed prototypes overview
- Existing limitations and most pressing issues
- Generalization of HA programming models
- Enhancing the transparency of the HA infrastructure
- Generic HA framework infrastructure

Scientific High-End Computing (HEC)

- Large-scale HPC systems.
  - Tens-to-hundreds of thousands of processors.
  - Current systems: IBM Blue Gene/L and Cray XT4
  - Next-generation: petascale IBM Blue Gene and Cray XT
- Computationally and data intensive applications.
  - 10 TFLOP – 1PFLOP with 10 TB – 1 PB of data.
  - Climate change, nuclear astrophysics, fusion energy, materials sciences, biology, nanotechnology, …
- Capability vs. capacity computing
  - Single jobs occupy large-scale high-performance computing systems for weeks and months at a time.

Projected Performance Development

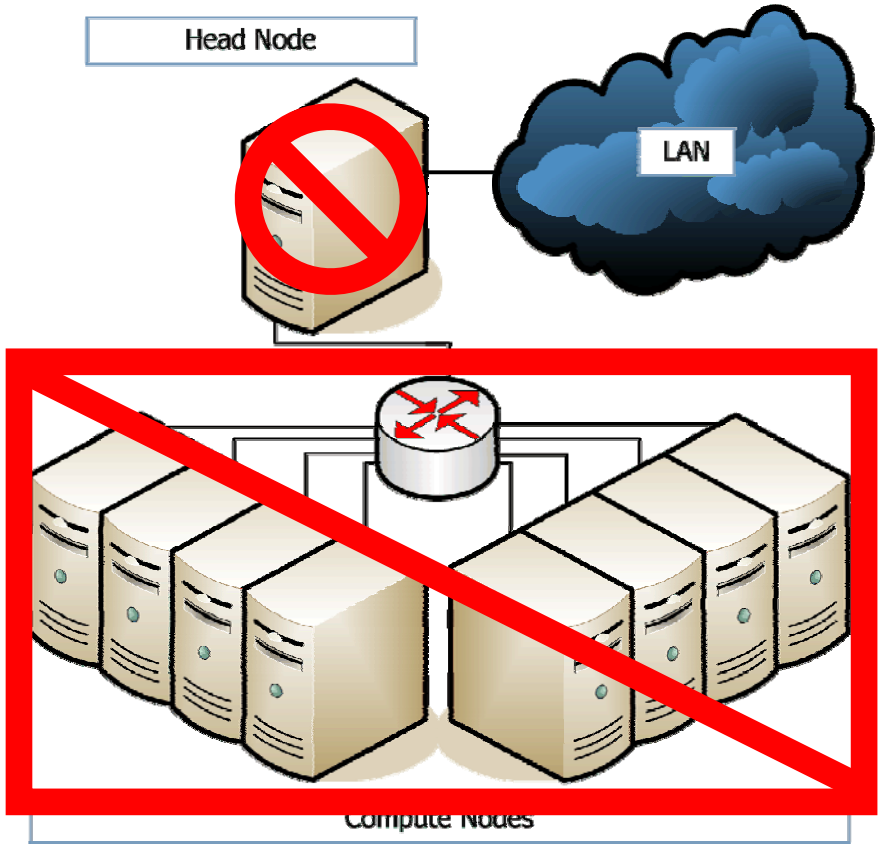


Availability Measured by the Nines

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

Single Head/Service Node Problem



- Single point of failure.
- Compute nodes sit idle while head node is down.
- A = MTTF / (MTTF + MTTR)
- MTTF depends on head node hardware/software quality.
- MTTR depends on the time it takes to repair/replace node.
- ➢ MTTR = 0 ➔ A = 1.00 (100%) **continuous availability**.

Projects Overview

- Initial **HA-OSCAR** research in active/standby technology for the batch job management system
- Ongoing **MOLAR** research in active/standby, asymmetric and symmetric active/active technology
- Recent **RAS LDRD** research in symmetric active/active technology

➔ 3-4 years of research and development in high availability for high-performance computing system services

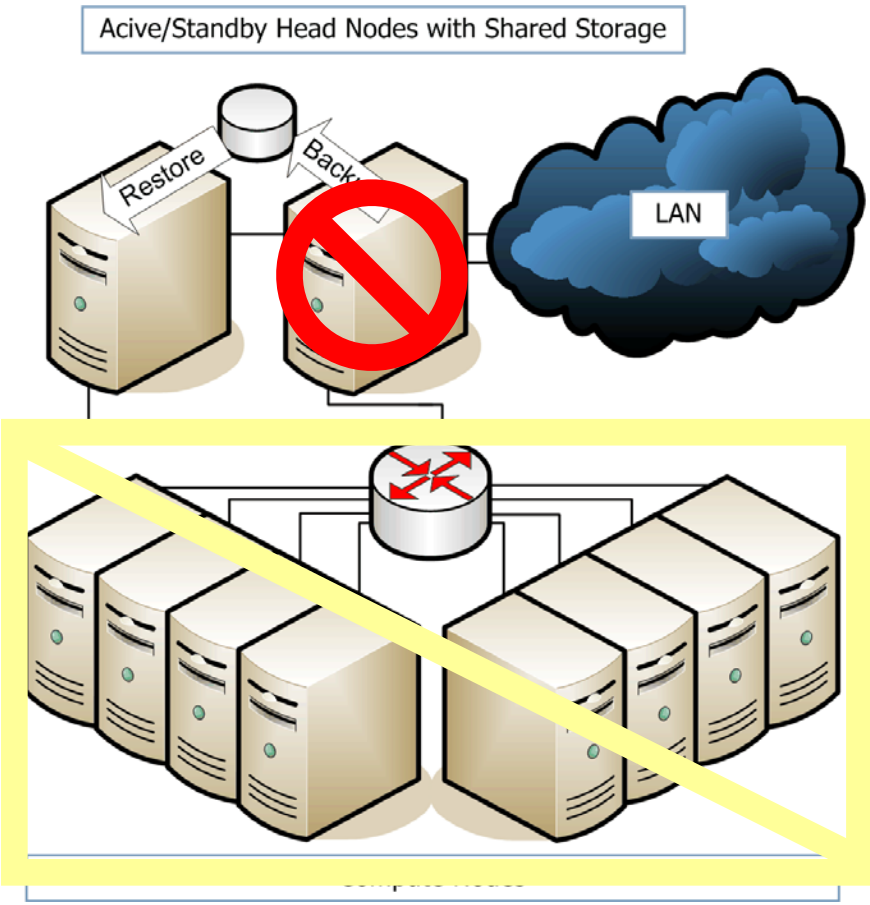
Accomplishments Overview

- Investigated the overall background of HA technologies in the context of HPC
  - Detailed problem description
  - Conceptual models
  - Review of existing solutions
- Developed different replication strategies for providing high availability for HPC system services
  - Active/standby
  - Asymmetric active/active
  - Symmetric active/active
- Implemented several proof-of-concept prototypes

High Availability Models

- Active/Standby (Warm or Hot)
  - For one active component at least one redundant inactive (standby) component
  - Fail-over model with idle standby component(s)
  - Level of high-availability depends on replication strategy
- Active/Active (Asymmetric or Symmetric)
  - Multiple redundant active components
  - No wasted system resources
  - State change requests can be accepted and may be executed by every member of the component group

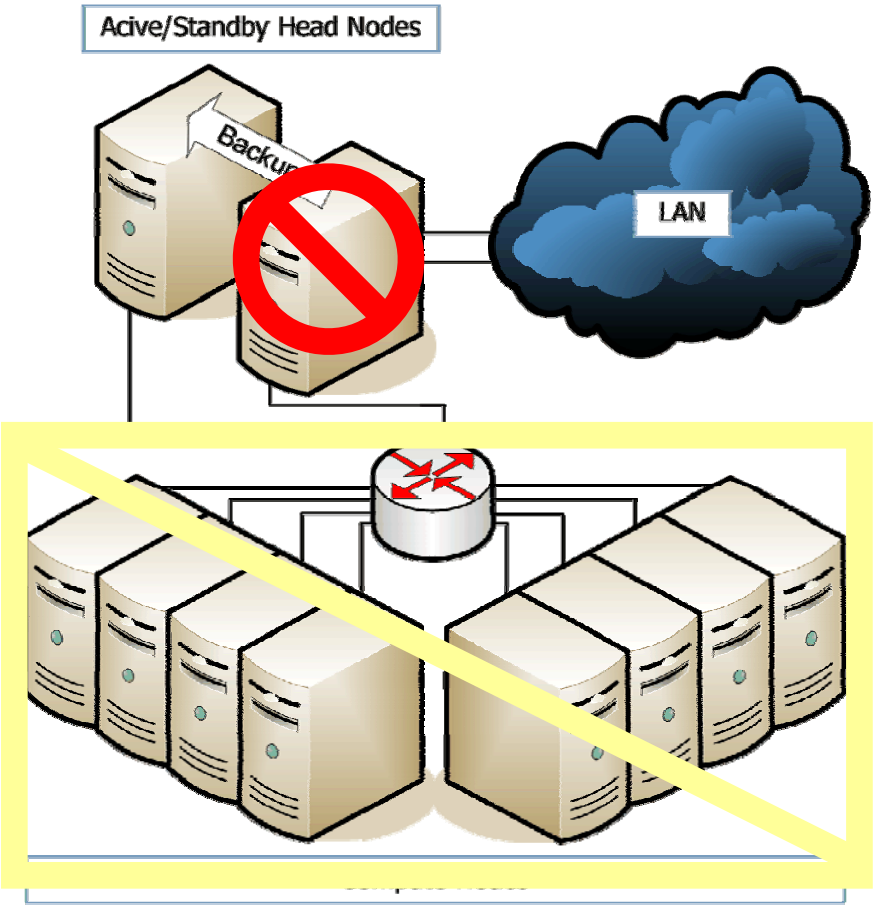
Active/Standby with Shared Storage



- Single active head node
- Backup to shared storage
- Simple checkpoint/restart
- Fail-over to standby node
- Possible corruption of backup state when failing during backup
- Introduction of a new single point of failure
- Correctness and availability are NOT ALWAYS guaranteed

➔ SLURM, meta data servers of PVFS and Lustre

Active/Standby Redundancy

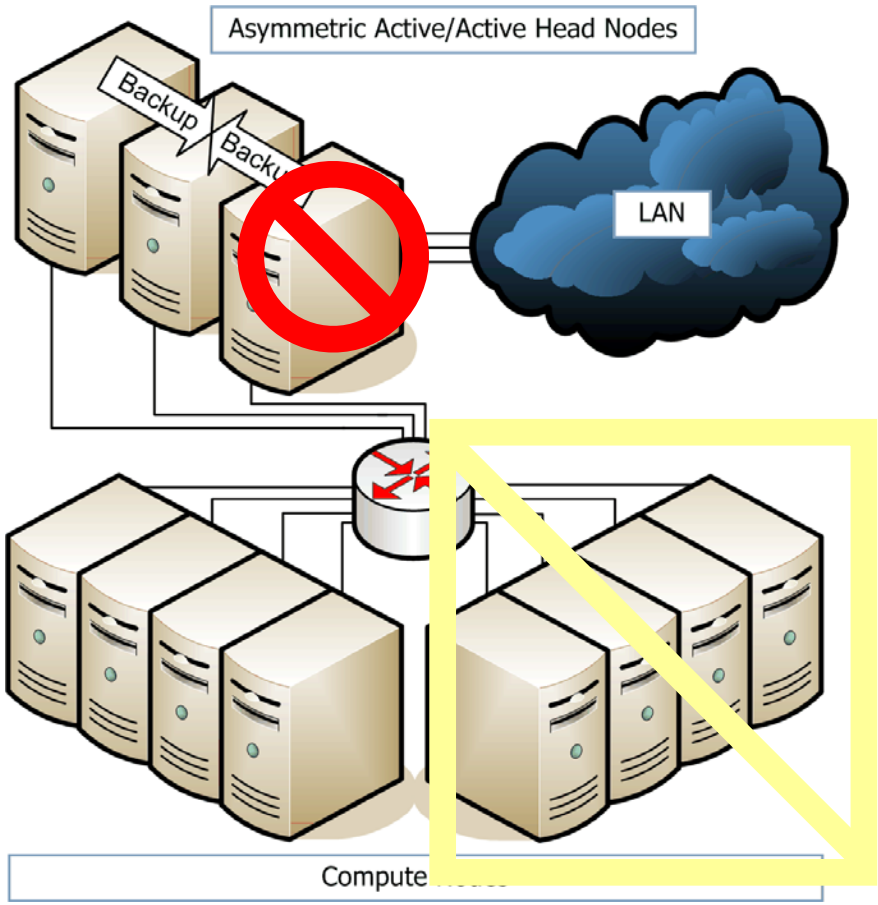


- Single active head node
- Backup to standby node
- Simple checkpoint/restart
- Fail-over to standby node
- Idle standby head node
- Rollback to backup
- Service interruption for fail-over and restore-over

➔ Torque on Cray XT

➔ <u>HA-OSCAR prototype</u>

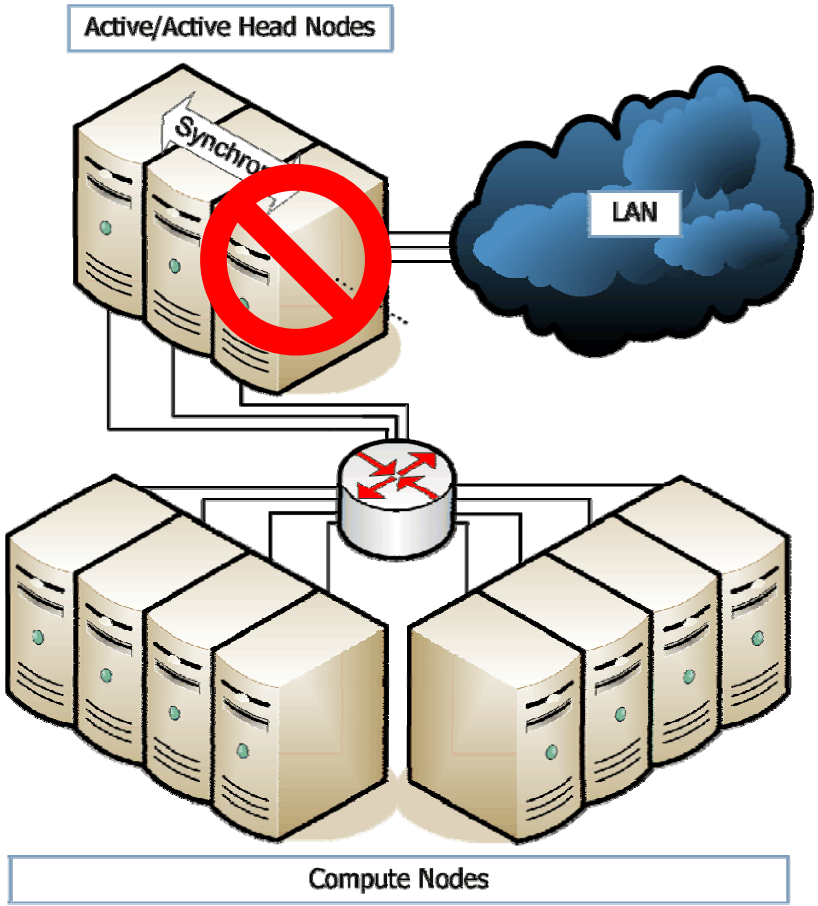
Asymmetric Active/Active Redundancy



- Many active head nodes
- Work load distribution
- Optional fail-over to standby head node(s) (*n+1* or *n+m*)
- No coordination between active head nodes
- Service interruption for fail-over and restore-over
- Loss of state w/o standby
- Limited use cases, such as high-throughput computing

➔ <u>Prototype based on HA-OSCAR</u>

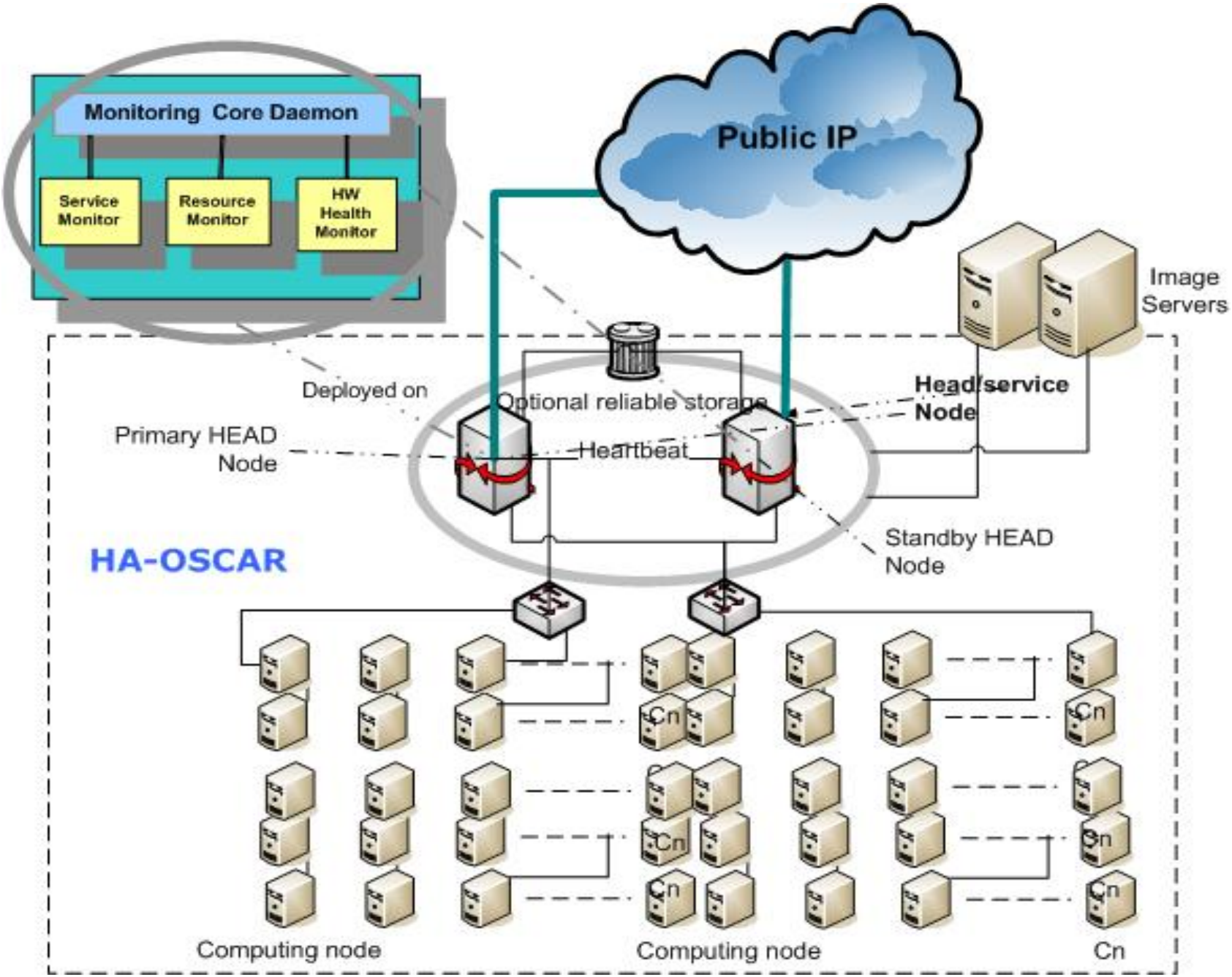
Symmetric Active/Active Redundancy

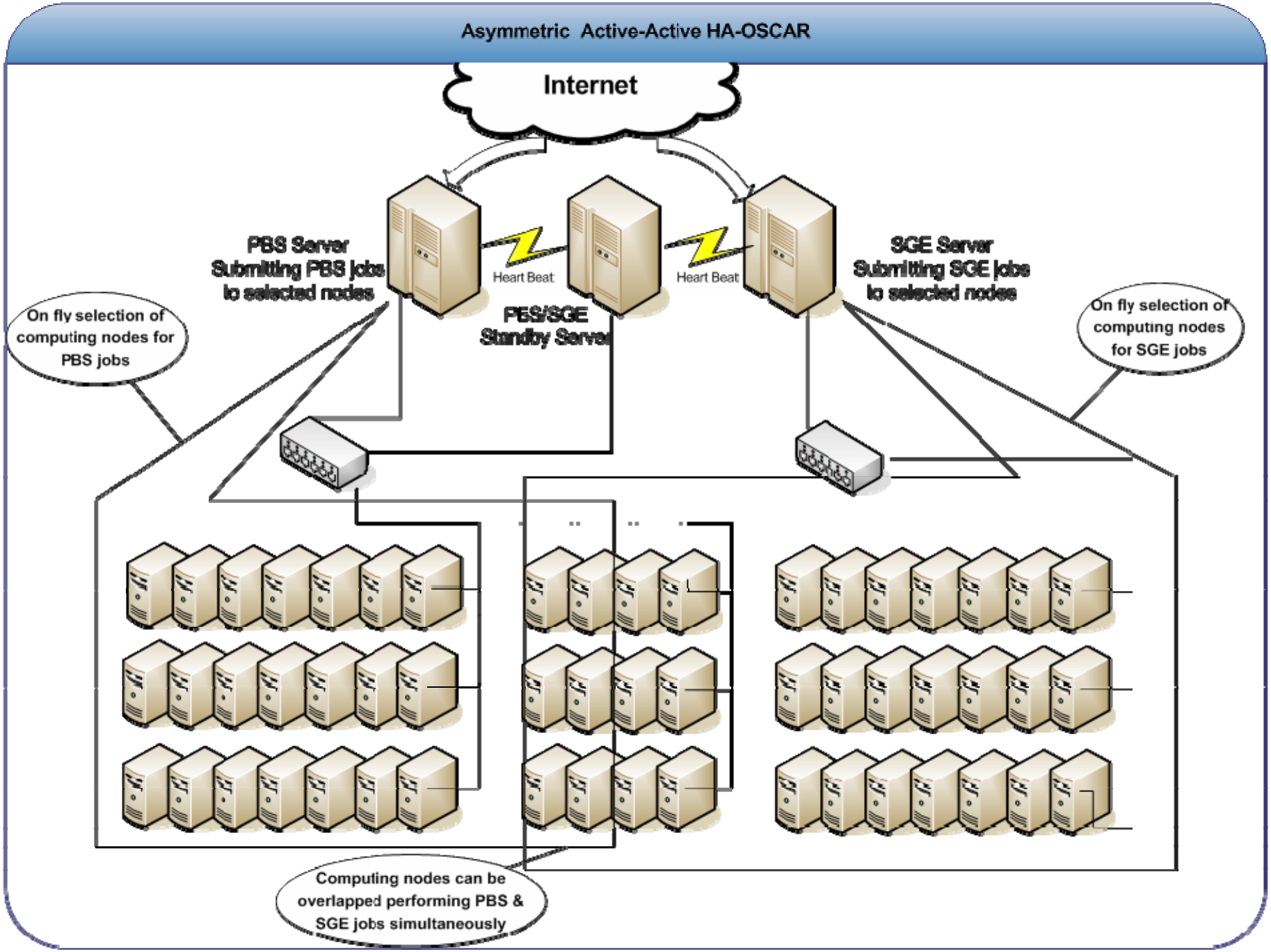


- Many active head nodes
- Work load distribution
- Symmetric replication between head nodes
- Continuous service
- Always up-to-date
- No fail-over necessary
- No restore-over necessary
- Virtual synchrony model
- Complex algorithms
- JOSHUA prototype for Torque

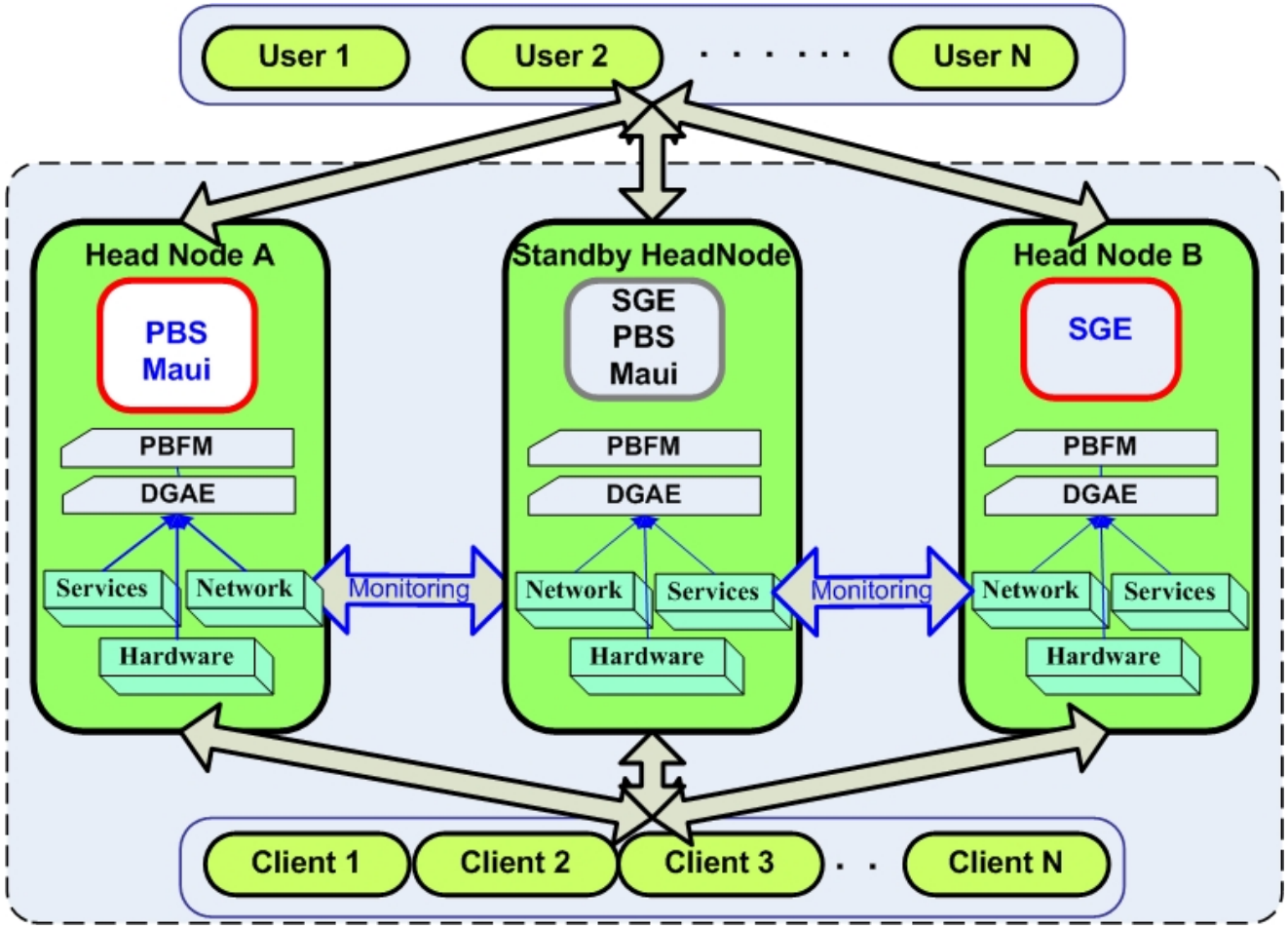
Developed Prototypes Overview (1/2)

- Active/Standby HA-OSCAR
  - High availability for Open PBS/TORQUE
  - Integration with compute node checkpoint/restart
- Asymmetric active/active HA-OSCAR
  - High availability for Open PBS & SGE
  - High throughput computing solution
- Symmetric active/active JOSHUA
  - High availability for PBS TORQUE
  - Fully transparent replication

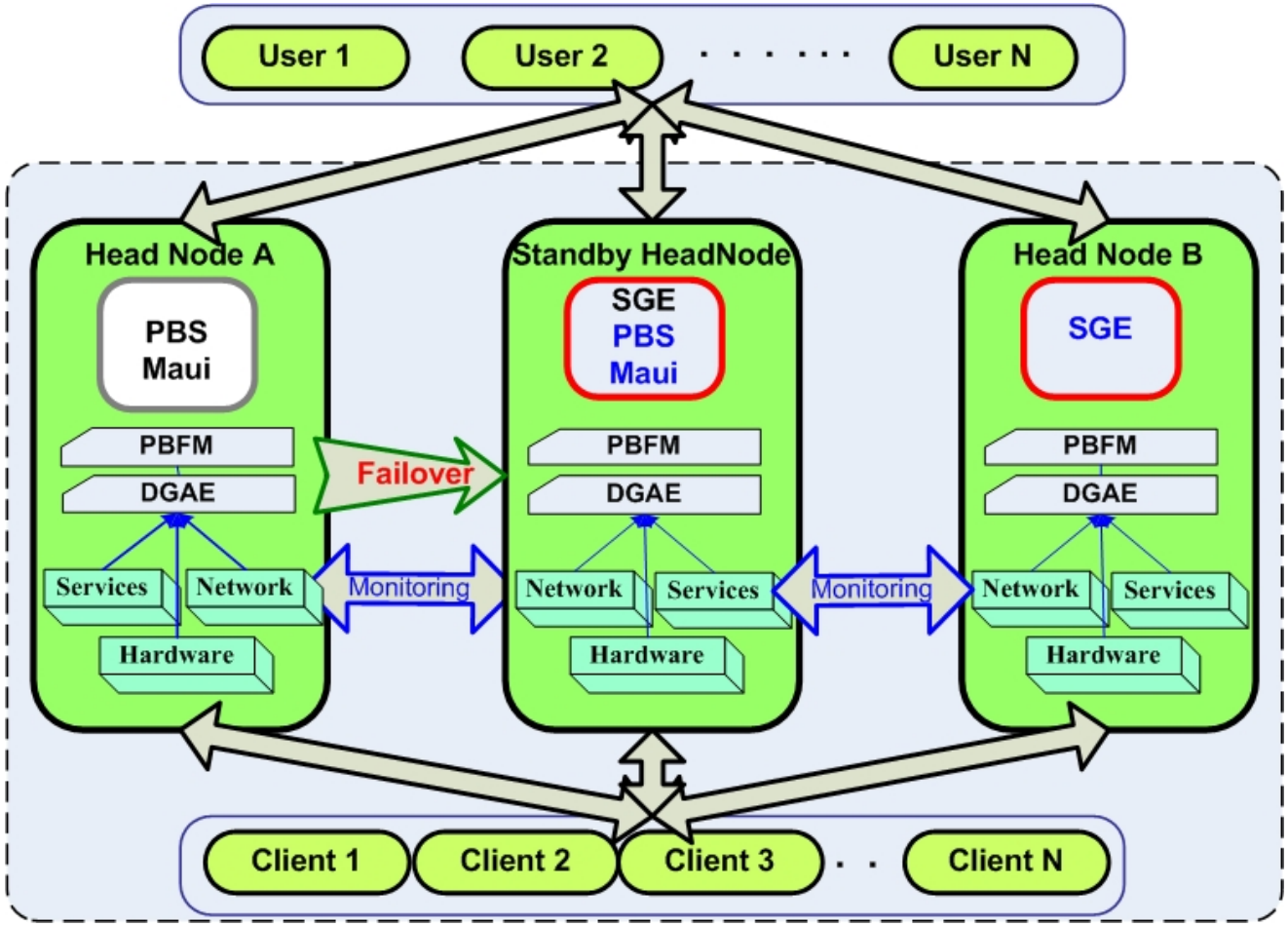

Monitoring Core Daemon
Service Monitor
Resource Monitor
HW Health Monitor
Public IP
Image Servers
Deployed on
Optional reliable storage
Head/service Node
Primary HEAD Node
Heartbeat
Standby HEAD Node
HA-OSCAR
Cn
Computing node
Computing node
Cn


Asymmetric Active-Active HA-OSCAR
Internet
PBS Server
Submitting PBS jobs
to selected nodes
Heart Beat
Heart Beat
PBS/SGE
Standby Server
SGE Server
Submitting SGE jobs
to selected nodes
On fly selection of
computing nodes for
PBS jobs
On fly selection of
computing nodes
for SGE jobs
Computing nodes can be
overlapped performing PBS &
SGE jobs simultaneously

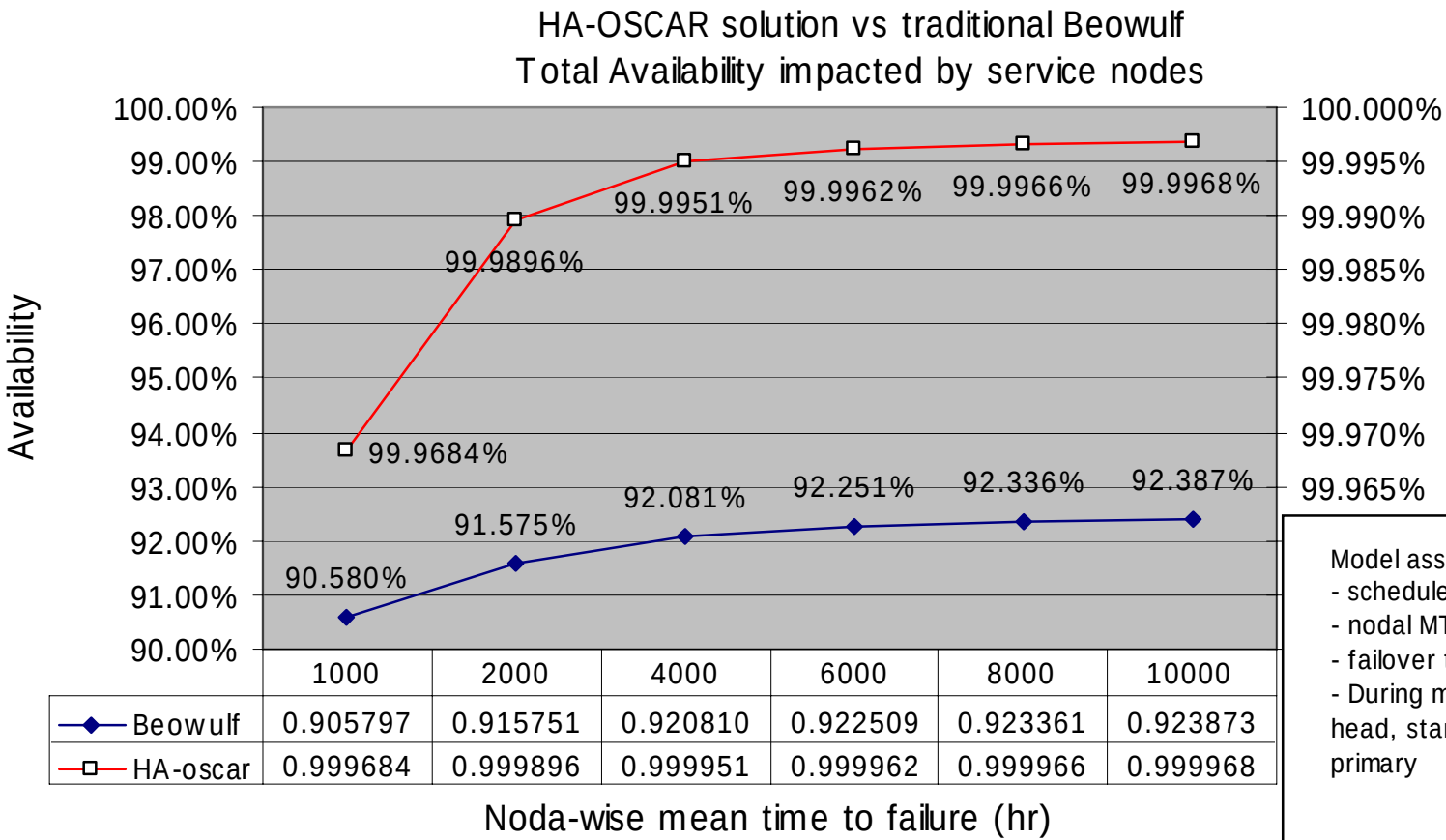
Normal Active-Active Operation



Failover Active-Active Operation



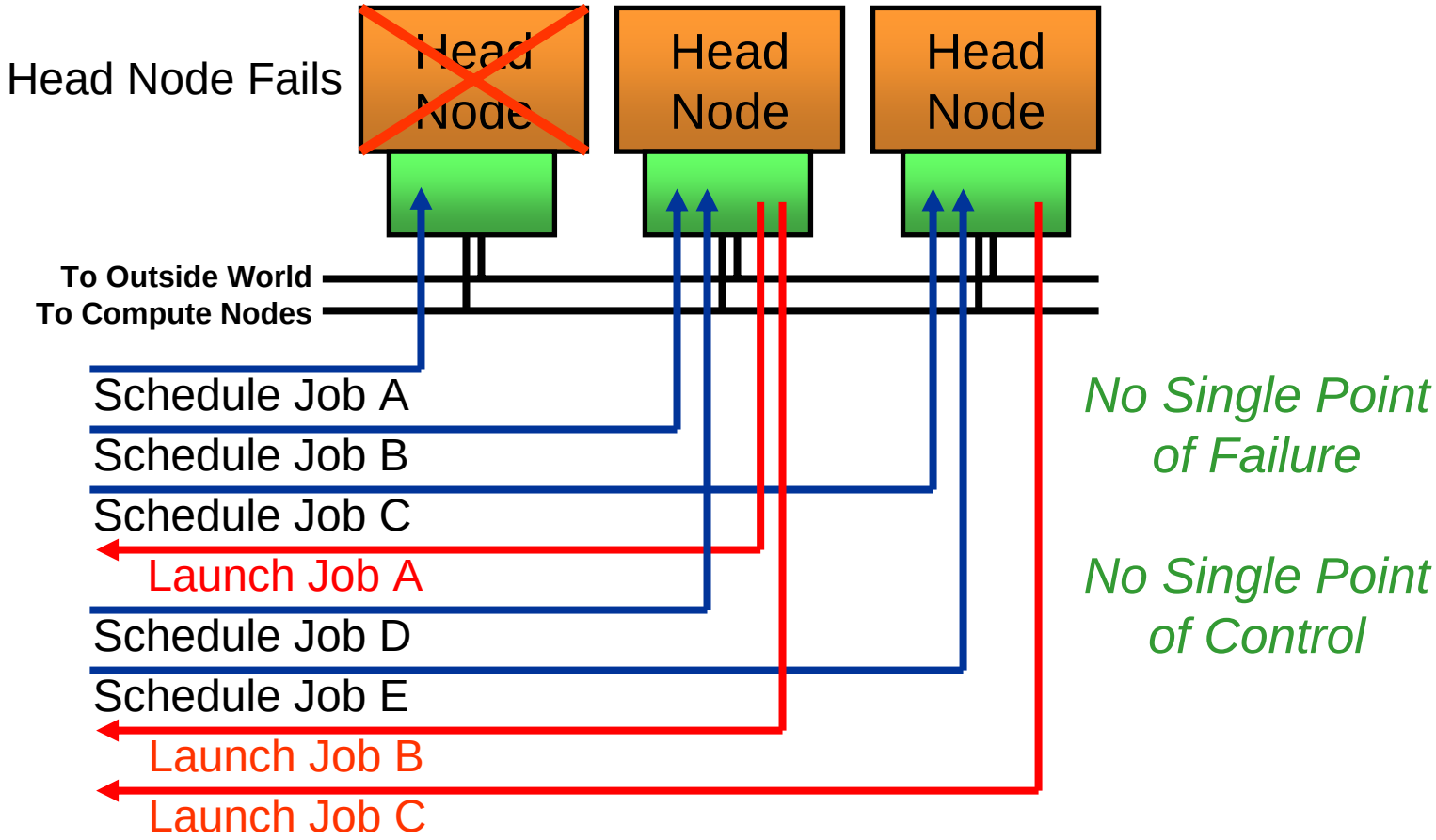
Asymmetric Active/Active Availability



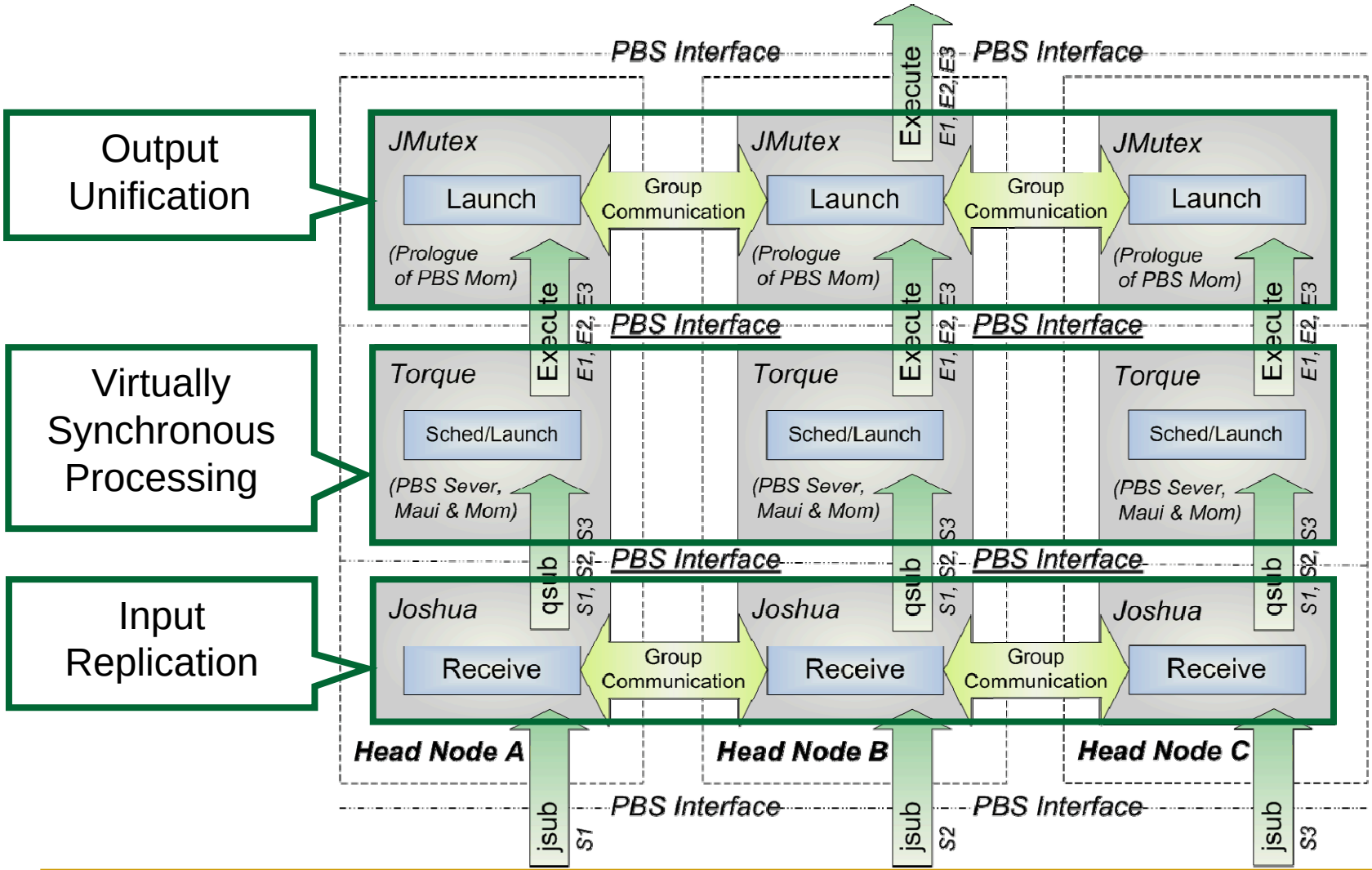
	1000	2000	4000	6000	8000	10000
Beowulf	0.905797	0.915751	0.920810	0.922509	0.923361	0.923873
HA-oscar	0.999684	0.999896	0.999951	0.999962	0.999966	0.999968

Model assumption:
- scheduled downtime=200 hrs
- nodal MTTR = 24 hrs
- failover time=10s
- During maintainance on the head, standby node acts as primary

JOSHUA: Symmetric Active/Active Replication for PBS Torque



Symmetric Active/Active Replication



Introduced Overhead

- Group communication system adds overhead for reliable and atomic multicast
- Latency increases with number of active nodes
- Throughput decreases with number of active nodes
- Overhead in acceptable range for this scenario

- Nodes: Pentium III 450MHz on 100MBit/s Ethernet

System	#	Latency	Overhead
TORQUE	1	98 ms	
JOSHUA/TORQUE	1	134 ms	36 ms / 37%
JOSHUA/TORQUE	2	265 ms	158 ms / 161%
JOSHUA/TORQUE	3	304 ms	206 ms / 210%
JOSHUA/TORQUE	4	349 ms	251 ms / 256%

Job Submission Latency Overhead

System	#	10 Jobs	50 Jobs	100 Jobs
TORQUE	1	0.93s	4.95s	10.18s
JOSHUA/TORQUE	1	1.32s	6.48s	14.08s
JOSHUA/TORQUE	2	2.68s	13.09s	26.37s
JOSHUA/TORQUE	3	2.93s	15.91s	30.03s
JOSHUA/TORQUE	4	3.62s	17.65s	33.32s

Job Submission Throughput Overhead

Symmetric Active/Active Availability

- $A_{component}$ = MTTF / (MTTF + MTTR)
- A_{system} = 1 - (1 - $A_{component}$) n
- T_{down} = 8760 hours * (1 – A)
- Single node MTTF: 5000 hours
- Single node MTTR: 72 hours

Nodes	Availability	Est. Annual Downtime
1	98.58%	5d 4h 21m
2	99.97%	1h 45m
3	99.9997%	1m 30s
4	99.999995%	1s

Single-site redundancy for 7 nines does not mask catastrophic events.

Existing Limitations

- The active/standby and asymmetric active/active technology interrupts the service during fail-over
- Generic *n+1* or *n+m* asymmetric active/active configurations have not been developed yet
- The 2+1 asymmetric active/active configuration uses two different service implementations
- The developed symmetric active/active technology has certain stability and performance issues
- All developed prototypes use a customized high availability environment
- Missing interaction with compute node fault tolerance mechanisms (except for HA-OSCAR for head node fail-over)

Most Pressing Issues

- For production-type deployment
  - Stability – guaranteed quality of service
  - Performance – low replication overhead
  - Interaction with compute node fault tolerance mechanisms – e.g. procedure for failing PBS mom
  - ➔ Testing, enhancements, and staged deployment
- For extending the developed technologies
  - Portability – ability to apply technology to different services
  - Ease-of-use – simplified service HA management (RAS)
  - ➔ Generic HA framework needed

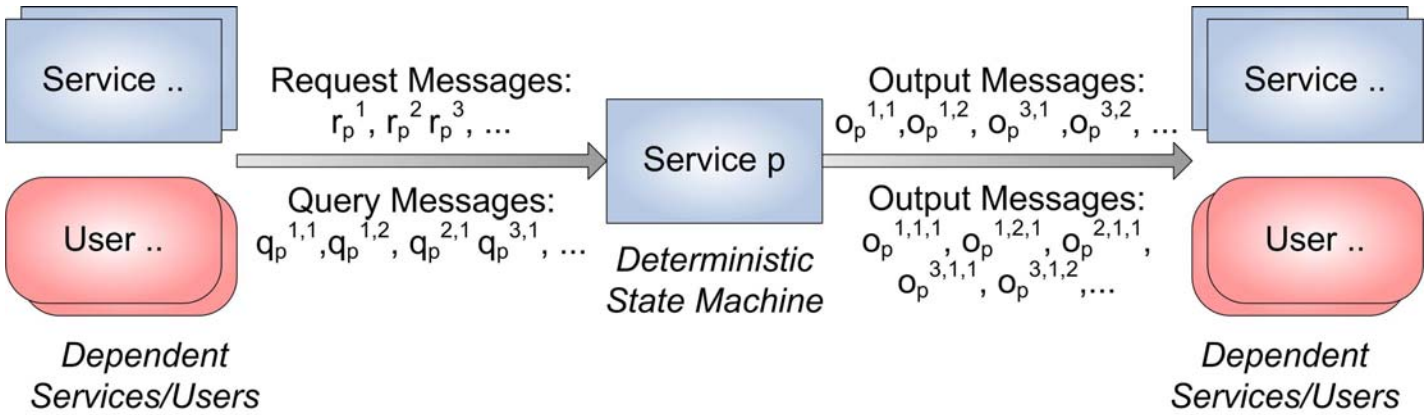
Next Step: Generic HA Framework

- Generalization of HA programming models
  - Active/Standby
  - Asymmetric active/active
  - Symmetric active/active
- Enhancing the transparency of the HA infrastructure
  - Minimum adaptation to the actual service protocol
  - Virtualized communication layer for abstraction

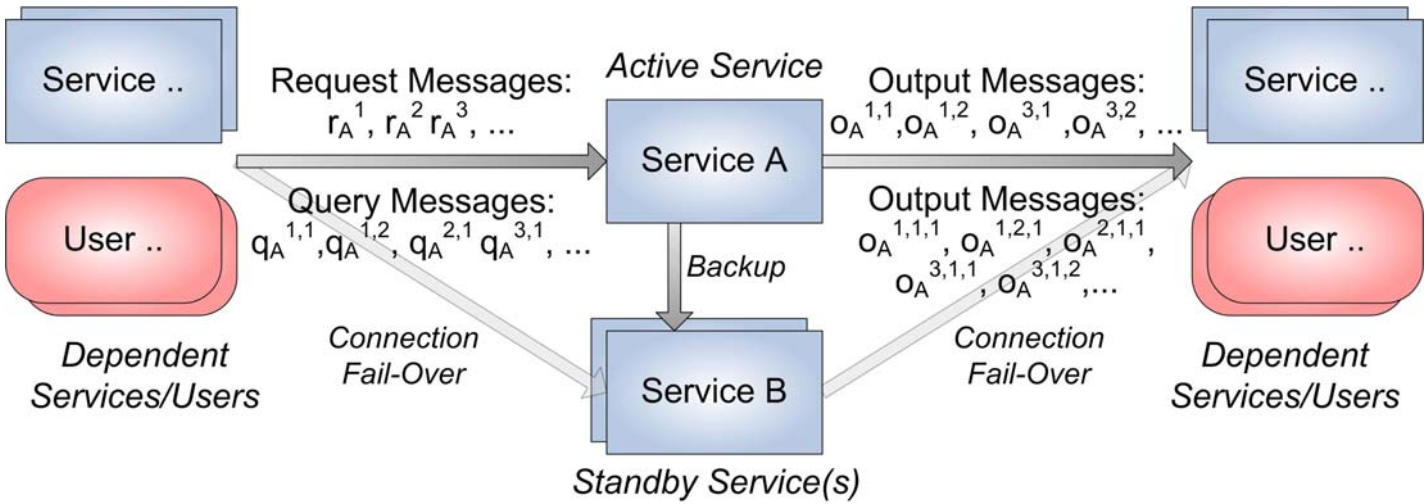
➔ Portability

➔ Ease-of-use

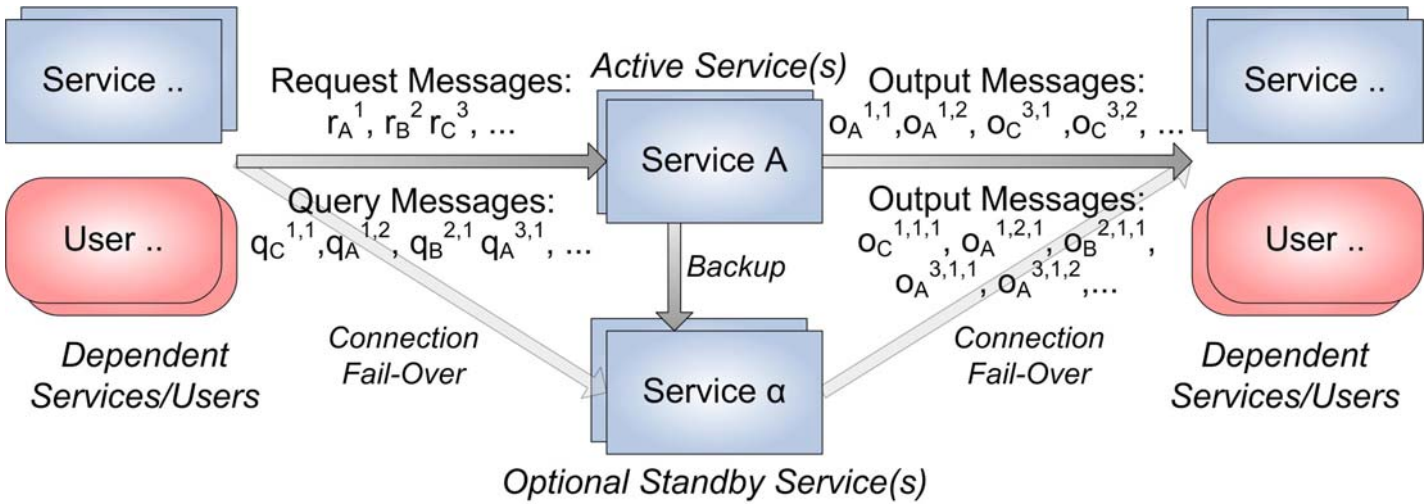
Communicating Process Generalization



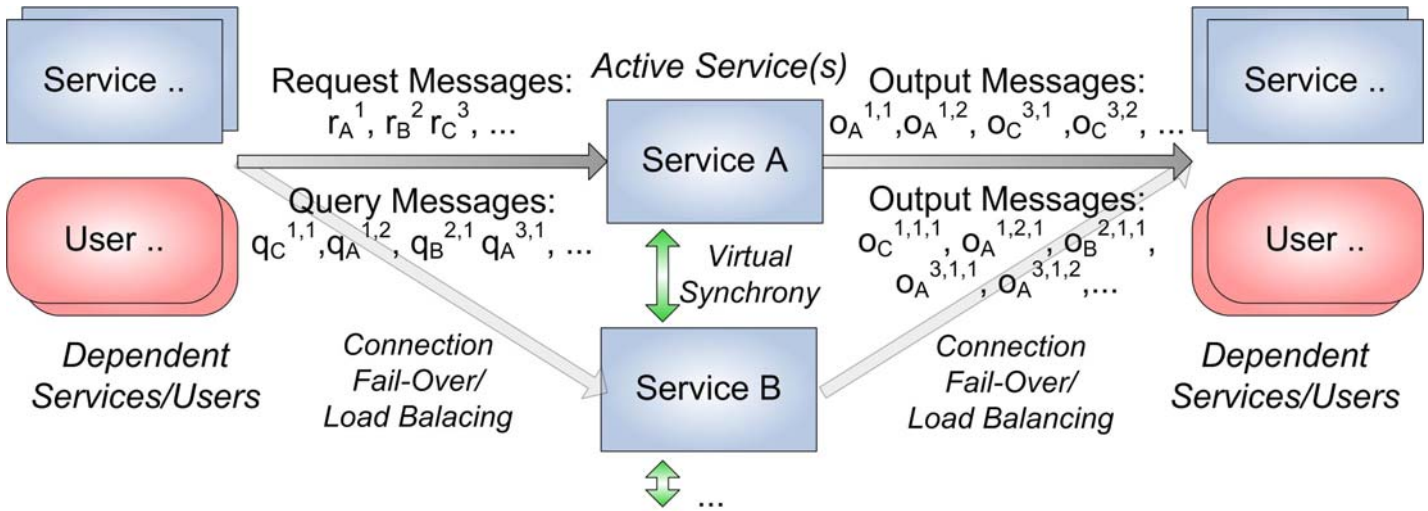
Active/Standby Generalization



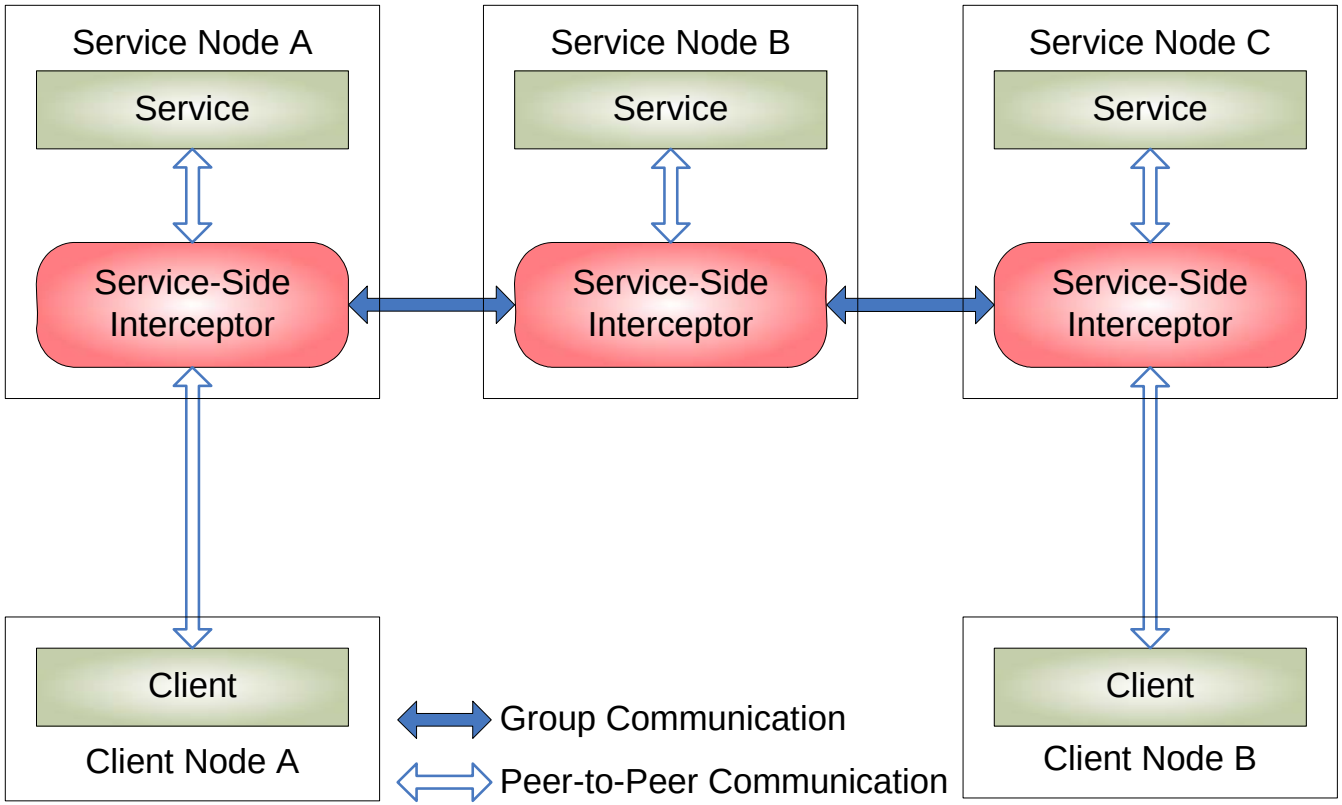
Asymmetric Active/Active Generalization



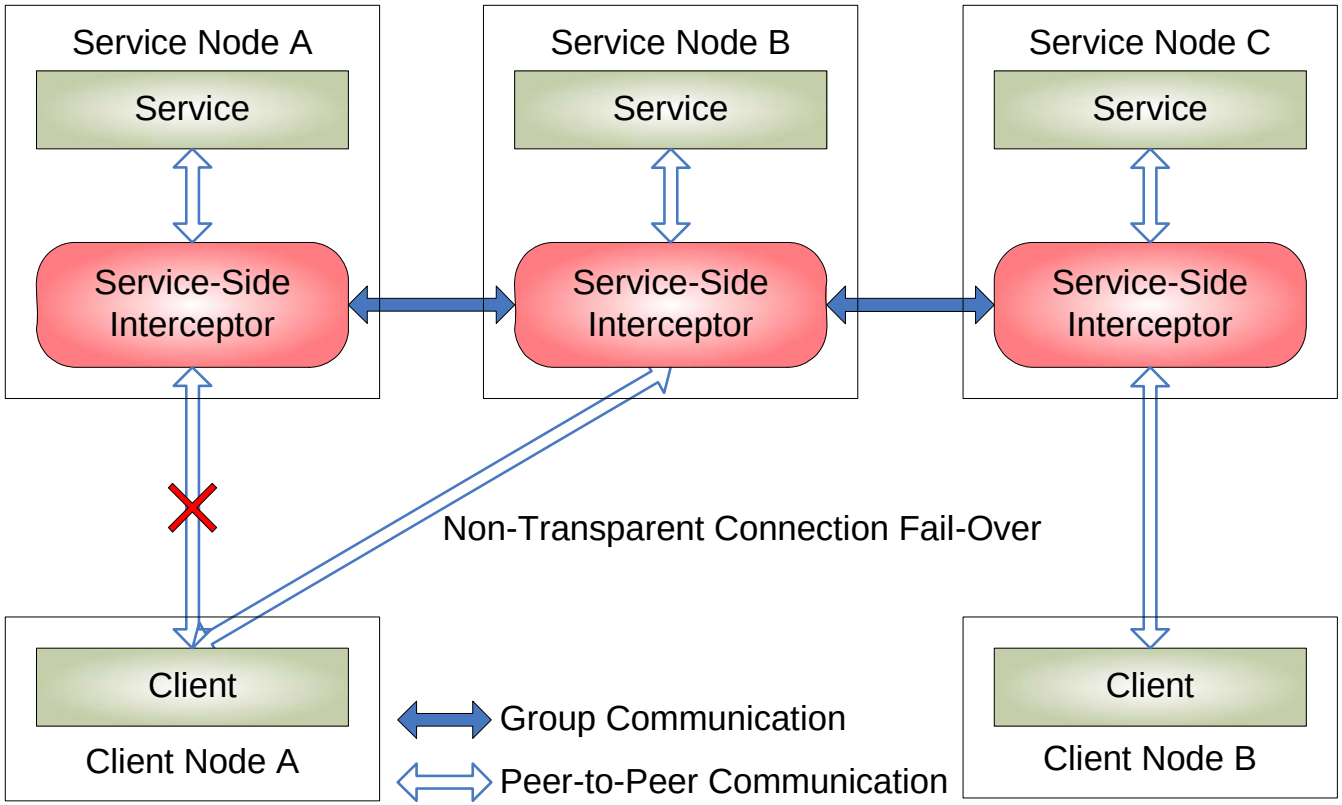
Symmetric Active/Active Generalization



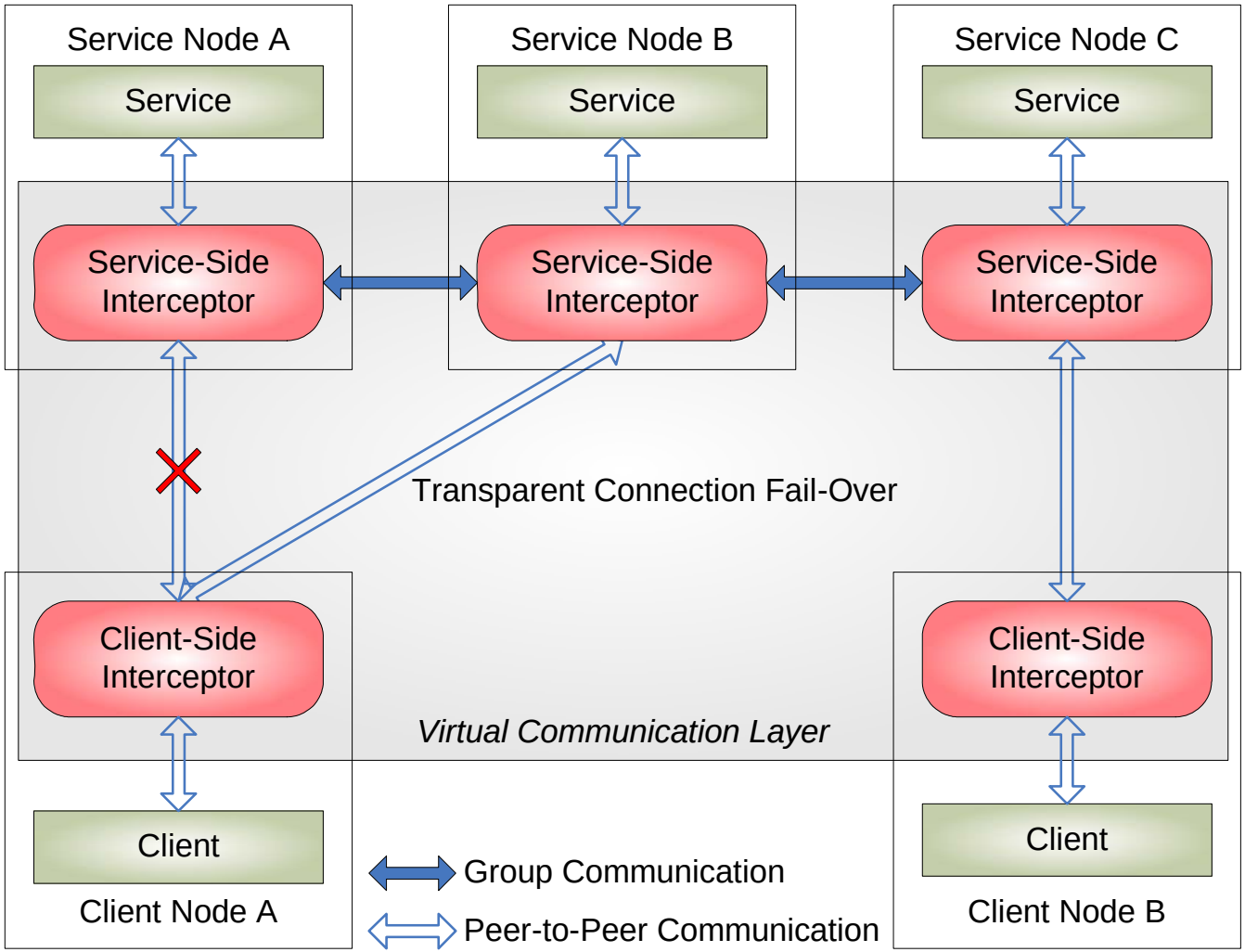
Symmetric Active/Active Replication



Non-Transparent Connection Fail-Over



Transparent Connection Fail-Over



Interceptors in the Communication Path: What about Performance?

Payload	Without Interceptors	With Service Interceptor	With Both Interceptors
$100B$	$149.9\mu s$	$150.6\mu s/\ +0.5\%$	$178.4\mu s/+19.0\%$
$1KB$	$284.3\mu s$	$314.6\mu s/+10.7\%$	$346.7\mu s/+21.9\%$
$10KB$	$1.9ms$	$1.9ms/\ \pm 0.0\%$	$2.0ms/\ +5.3\%$
$100KB$	$22.3ms$	$22.5ms/\ +0.8\%$	$22.7ms/\ +1.8\%$

Table 1. Ping-Pong Latency Comparison

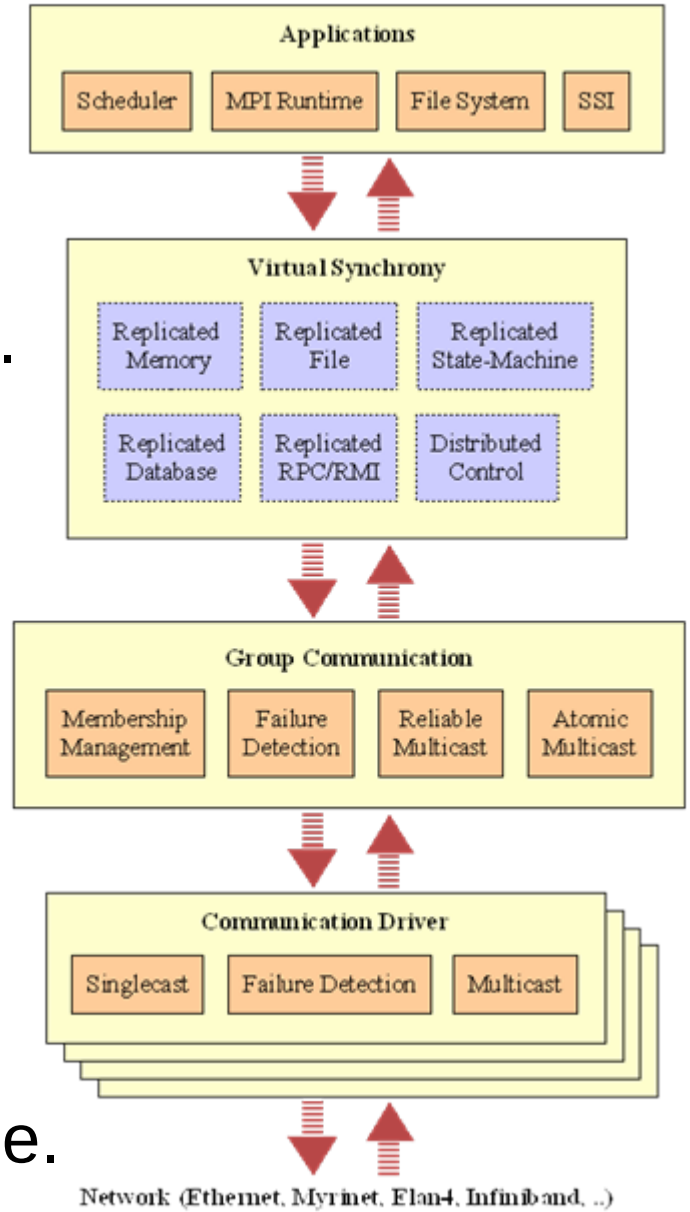
Payload	Without Interceptors	With Service Interceptor	With Both Interceptors
$100B$	$667KBps$	$664KBps/-0.4\%$	$561KBps/-15.9\%$
$1KB$	$3.5MBps$	$3.2MBps/-8.6\%$	$2.9MBps/-17.1\%$
$10KB$	$5.3MBps$	$5.2MBps/-1.9\%$	$5.0MBps/\ -5.7\%$
$100KB$	$4.5MBps$	$4.4MBps/-2.2\%$	$4.4MBps/\ -2.2\%$

Table 2. Ping-Pong Bandwidth Comparison

Test Results from a 100 Mbit/s LAN Environment

Modular HA Framework

- Pluggable component framework.
  - Communication drivers.
  - Group communication.
  - Virtual synchrony.
  - *Applications.*
- Interchangeable components.
- Adaptation to application needs, such as level of consistency.
- Adaptation to system properties, such as network and system scale.



Current Prototype

- Unique, flexible, dynamic, C-based component framework: Adaptive Runtime Environment (ARTE)
- Dynamic component loading/unloading on demand
- *XML as interface description language (IDL)*
- "Everything" is a component:
  - Communication driver modules
  - Group communication layer modules
  - Virtual synchrony layer modules

Future Work

- Continued implementation of framework components
  - Implementation of HA programming model components
- Integration with existing prototypes
  - For example, replacing Transis with the framework
- Availability and reliability modeling
- Testing and benchmarking
- Journal paper
- What about communication security/integrity?
  - For client-server connections across administrative domains
  - For distributed computing scenarios
- Start writing my PhD thesis!!!

MOLAR: Adaptive Runtime Support for High-end Computing Operating and Runtime Systems

- Addresses the challenges for operating and runtime systems to run large applications efficiently on future ultra-scale high-end computers.
- Part of the Forum to Address Scalable Technology for Runtime and Operating Systems (FAST-OS).
- MOLAR is a collaborative research effort (*www.fastos.org/molar*):

Directly Related Publications

1. C. Engelmann, S. L. Scott, C. Leangsuksun, and X. He. **Transparent symmetric active/active replication for service-level high availability**. In *Proceedings of 7th IEEE International Symposium on Cluster Computing and the Grid (**CCGrid**) **2007***, Rio de Janeiro, Brazil, May 14-17, 2007. To appear.
2. C. Engelmann, S. L. Scott, C. Leangsuksun, and X. He. **On programming models for service-level high availability**. In *Proceedings of 2nd International Conference on Availability, Reliability and Security (**ARES**) **2007***, Vienna, Austria, April 10-13, 2007. To appear.
3. C. Engelmann, S. L. Scott, C. Leangsuksun, and X. He. **Symmetric active/active high availability for high-performance computing system services**. *Journal of Computers (**JCP**)*, **1(8)**, **2006**.
4. C. Engelmann, S. L. Scott, C. Leangsuksun, and X. He. **Towards high availability for high-performance computing system services: Accomplishments and limitations**. In *Proceedings of High Availability and Performance Workshop (**HAPCW**) **2006***, Santa Fe, NM, USA, October 17, 2005.
5. K. Uhlemann, C. Engelmann, and S. L. Scott. **JOSHUA: symmetric active/active replication for highly available HPC job and resource management**. In *Proceedings of IEEE International Conference on Cluster Computing (**Cluster**) **2006***, Barcelona, Spain, September 25-28, 2006.
6. D. Okunbor, C. Engelmann, and S. L. Scott. **Exploring process groups for reliability, availability and serviceability of terascale computing systems**. In *Proceedings of 2nd **International Conference on Computer Science and Information Systems 2006***, Athens, Greece, June 19-21, 2006.

Directly Related Publications

7. C. Engelmann, S. L. Scott, D. E. Bernholdt, N. R. Gottumukkala, C. Leangsuksun, J. Varma, C. Wang, F. Mueller, A. G. Shet, and P. Sadayappan. **MOLAR: Adaptive runtime support for high-end computing operating and runtime systems**. ACM SIGOPS Operating Systems Review (**OSR**), **40(2)**, pages 63-72, **2006**
8. C. Engelmann, S. L. Scott, C. Leangsuksun, and X. He. **Active/active replication for highly available HPC system services**. In *Proceedings of 1st International Conference on Availability, Reliability and Security (**ARES**) **2006***, pages 639-645, Vienna, Austria, April 20-22, 2006.
9. C. Engelmann and S. L. Scott. **Concepts for high availability in scientific high-end computing**. In *Proceedings of High Availability and Performance Workshop (**HAPCW**) **2005***, Santa Fe, NM, USA, October 11, 2005.
10. C. Engelmann and S. L. Scott. **High availability for ultra-scale high-end scientific computing**. In *Proceedings of 2nd International Workshop on Operating Systems, Programming Environments and Management Tools for High-Performance Computing on Clusters (**COSET-2**) **2005***, Cambridge, MA, USA, June 19, 2005.
11. C. Leangsuksun, V. K. Munganuru, T. Liu, S. L. Scott, and C. Engelmann. **Asymmetric active-active high availability for high-end computing**. In *Proceedings of 2nd International Workshop on Operating Systems, Programming Environments and Management Tools for High-Performance Computing on Clusters (**COSET-2**) **2005***, Cambridge, MA, USA, June 19, 2005.
12. C. Engelmann, S. L. Scott, and G. A. Geist. **High availability through distributed control**. In *Proceedings of High Availability and Performance Workshop (**HAPCW**) **2004***, Santa Fe, NM, USA, October 12, 2004.

Indirectly Related Publications

1. C. Engelmann, H. Ong, and S. L. Scott. **Middleware in modern high performance computing system architectures**. In Lecture Notes in Computer Science: Proceedings of International Conference on Computational Science (**ICCS**) **2007**, Beijing, China, May 27-30, 2007. To appear.
2. C. Wang, F. Mueller, C. Engelmann, and S. L. Scott. **A job pause service under LAM/MPI+BLCR for transparent fault tolerance**. In Proceedings of 21st International Parallel and Distributed Processing Symposium (**IPDPS**) **2007**, Long Beach, CA, USA, March 26-30, 2007. To appear.
3. J. Varma, C. Wang, F. Mueller, C. Engelmann, and S. L. Scott. **Scalable, fault-tolerant membership for MPI tasks on HPC systems**. In Proceedings of 20th ACM International Conference on Supercomputing (**ICS**) **2006**, pages 219-228, Cairns, Australia, June 28-30, 2006.
4. K. Limaye, C. Leangsuksun, Z. Greenwood, S. L. Scott, C. Engelmann, R. Libby, and K. Chanchio. **Jobsite level fault tolerance for cluster and grid environments**. In Proceedings of IEEE International Conference on Cluster Computing (**Cluster**) **2005**, Boston, MA, USA, September 26-30, 2005.
5. H. Song, C. Leangsuksun, R. Nassar, Y. Liu, C. Engelmann, and S. L. Scott. **UML-based Beowulf cluster availability modeling**. In Proceedings of International Conference on Software Engineering Research and Practice (**SERP**) **2005**, pages 161167, Las Vegas, NV, USA, June 27-30, 2005.

Towards High Availability for High-Performance Computing System Services: Accomplishments and Limitations

Christian Engelmann[1,2]

[1] Oak Ridge National Laboratory, Oak Ridge, USA

[2] The University of Reading, Reading, UK