High Throughput Computing on Blue Gene

IBM Rochester: Amanda Peters, Tom Budnik

With contributions from:
IBM Rochester: Mike Mundy, Greg Stewart, Pat McCarthy
IBM Watson Research: Alan King, Jim Sexton
UW-Madison Condor: Greg Thain, Miron Livny, Todd Tannenbaum

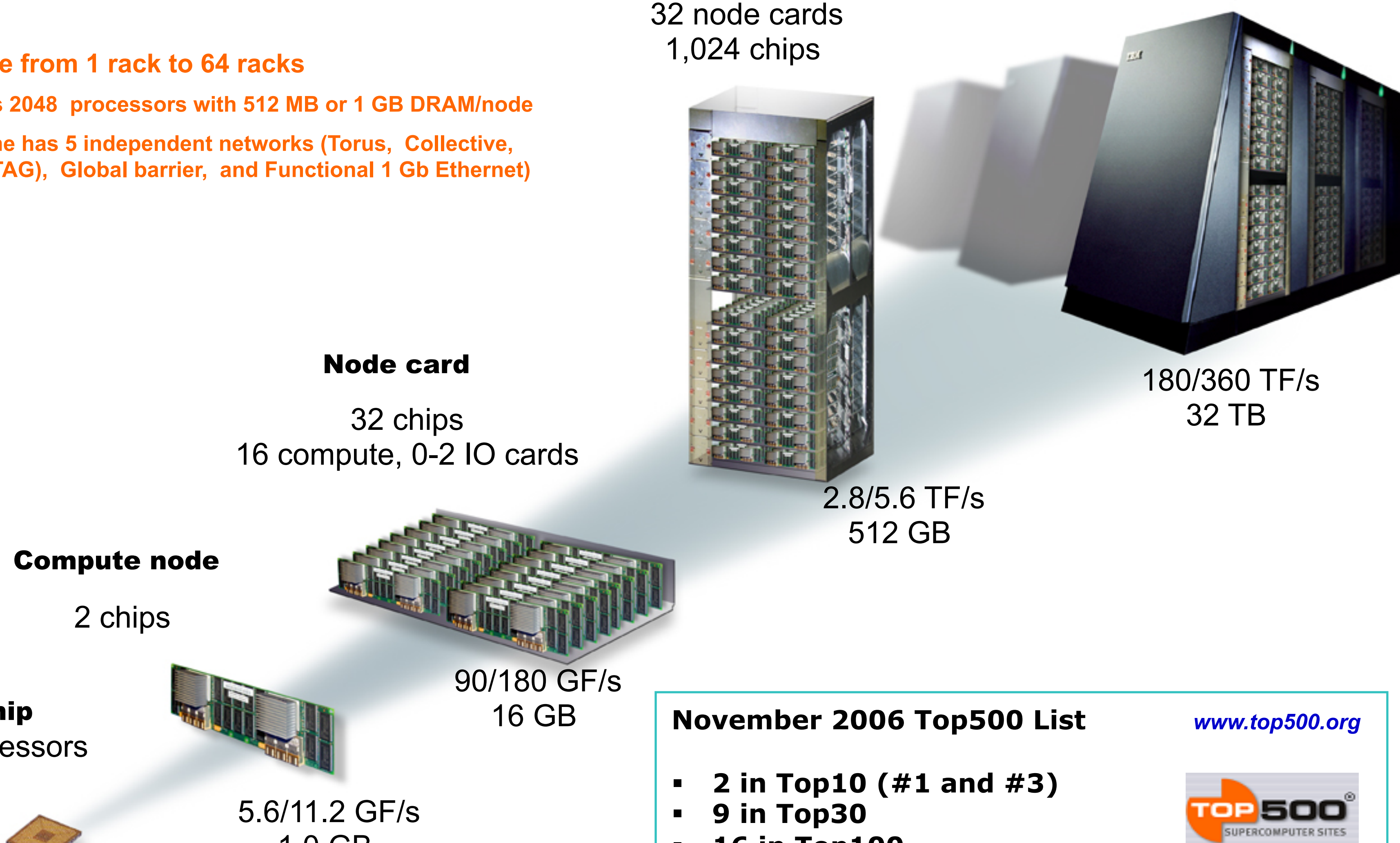
Agenda

- **Blue Gene Architecture Overview**
- **High Throughput Computing (HTC) on Blue Gene**
- **Condor and IBM Blue Gene Collaboration**
- **Exploratory Application Case Studies for Blue Gene HTC**
- **Questions and Web resource links**

Blue Gene/L Overview

Scalable from 1 rack to 64 racks

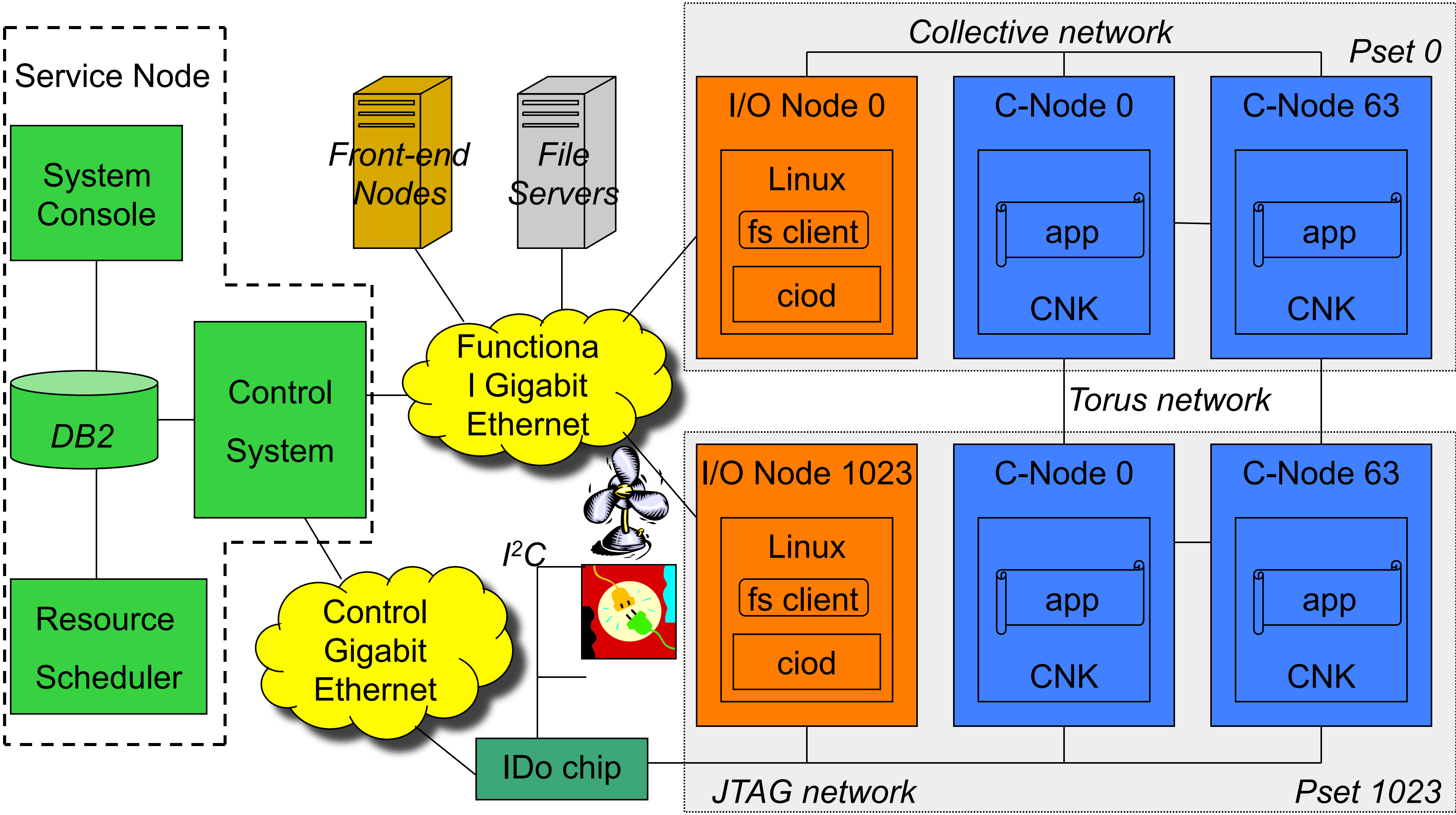
- **Rack has 2048 processors with 512 MB or 1 GB DRAM/node**
- **Blue Gene has 5 independent networks (Torus, Collective, Control (JTAG), Global barrier, and Functional 1 Gb Ethernet)**



November 2006 Top500 List *www.top500.org*

- **2 in Top10 (#1 and #3)**
- **9 in Top30**
- **16 in Top100**
- **27 overall in Top150**

Blue Gene System Architecture



HPC vs. HTC Comparison

- **High Performance Computing (HPC) Model**
  - Parallel, tightly coupled applications
    - Single Instruction, Multiple Data (SIMD) architecture
  - Programming model: typically MPI
  - Apps need tremendous amount of computational power over short time period
- **High Throughput Computing (HTC) Model**
  - Large number of independent tasks
    - Multiple Instruction, Multiple Data (MIMD) architecture
  - Programming model: non-MPI
  - Apps need large amount of computational power over long time period
  - Traditionally run on large clusters
- **HTC and HPC modes co-exist on Blue Gene**
  - Determined when resource pool (partition) is allocated

Why Blue Gene for HTC?

- **High processing capacity with minimal floor space**
  - High compute node density – 2,048 processors in one Blue Gene rack
  - Scalability from 1 to 64 racks (2,048 to 131,072 processors)
- **Resource consolidation**
  - Multiple HTC and HPC workloads on a single system
  - Optimal use of compute resources
- **Low power consumption**
  - **#1 on Green500 list @ 112 MFlops/Watt (www.green500.org/CurrentLists.html)**
  - Twice the performance per watt of a high frequency microprocessor
- **Low cooling requirements enable extreme scale-up**
- **Centralized system management**
  - Blue Gene Navigator

Welcome, tbudnik

End session

Blue Gene Navigator

Administration
- HealthCenter
- Jobs
- Blocks
- MMCS Console
- RAS Event Log
- Job History
- System Logs
- Utilization
- Midplane Activity
- Environmental Queries
- Hardware Browser
- Link Summary
- Block Builder
- Diagnostics
- Service Actions

Resources
- IBM Support

All Midplanes available
All Service cards available
All Link cards available
All Node cards available

Attention

65 Kernel Fatal
78 Application Fatal
103 Kernel Informational

1 days Apply

Job Summary

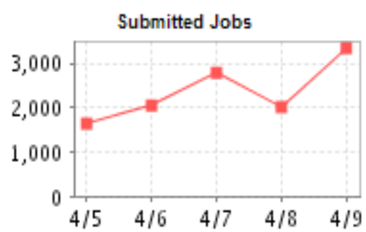
Filter Options

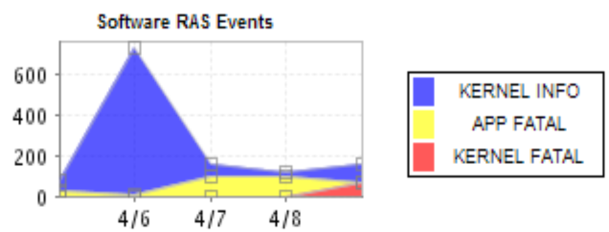
Filter Options: All jobs

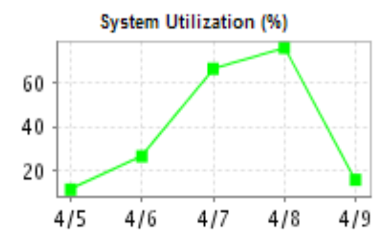
Job ID	Block ID	User Name	Executable	Mode	Status	Status Changed
285933		mmcsfvt0	/bglscratch/mmcsfvt0/testsdir/taw.JFmmr16125/bin/Hello	Coprocessor	Queued	3/27/07 2:23:17 AM
286094		mmcsfvt0	/bglscratch/mmcsfvt0/testsdir/taw.JFmmr16125/bin/HelloArgEnv	Coprocessor	Queued	3/27/07 3:36:32 AM
295551		cheema	/bglscratch/cheema/t-node.rts	Coprocessor	Queued	4/5/07 5:46:29 PM
295552		cheema	/bglscratch/cheema/t-node.rts	Coprocessor	Queued	4/5/07 5:46:29 PM
306255	GPFS_R010	randyab	/gpfs/bgl0/randyab/Run-IOR/IOR/IOR-2.8.3/IOR.rts	Coprocessor	Running	4/9/07 3:47:23 PM

Refresh

Dashboard



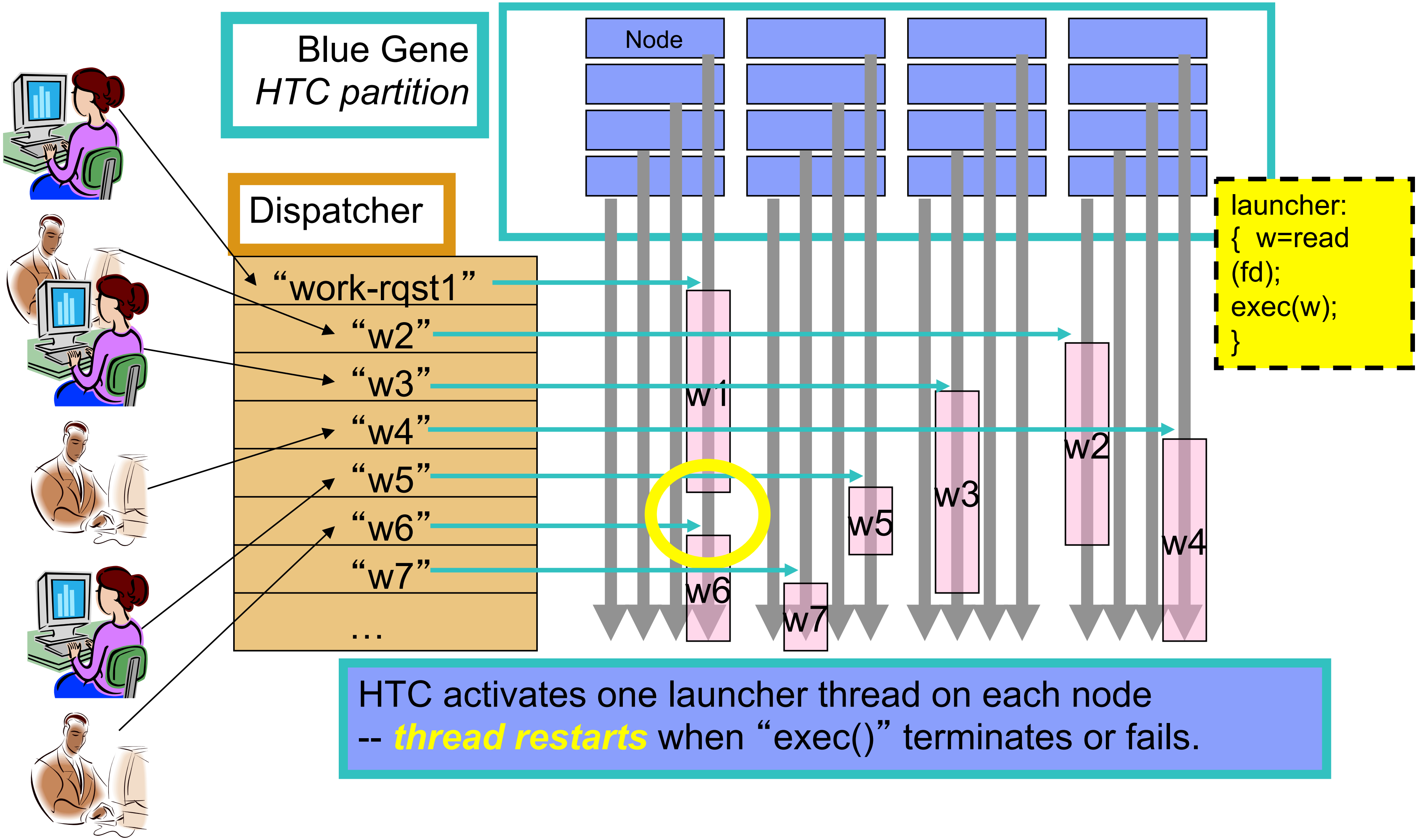




Generic HTC Flow on Blue Gene

- **One or more dispatcher programs are started on front end/service node**
  - Dispatcher will manage HTC work request queue
- **A pool (partition) of compute nodes is booted on Blue Gene**
  - Every compute node has a launcher program started on it that connects back to the designated HTC dispatcher
  - New pools of resources can be added dynamically as workload increases
- **External work requests are routed to HTC dispatcher queue**
  - Single or multiple work requests from each source
- **HTC dispatcher finds available HTC client and forwards the work request**
- **HTC client runs executable on compute node**
  - A launcher program on each compute node handles work request sent to it by the dispatcher. When work request completes, the launcher program is reloaded and client is ready to handle another work request.
- **Executable exit status is reported back to dispatcher**

Generic HTC Flow on Blue Gene



HTC activates one launcher thread on each node
-- ***thread restarts*** when "exec()" terminates or fails.

Node Resiliency for HTC

- **In HPC mode a single failing node in a partition (pool of compute nodes) causes termination of all nodes in the partition**
  - Expected behavior for parallel MPI type apps, but unacceptable for HTC apps
  - HTC mode partition handles this situation
- **In HTC mode Blue Gene can recover from soft node failures**
  - For example parity errors
  - If failure is not related to network hardware, a software reboot will recover the node
    - Other nodes in the partition are unaffected and continue to run jobs
    - Job on failed node is terminated and must be resubmitted by dispatcher
  - If the partition is started in HTC mode, the Control System will poll at regular intervals looking for nodes in the reset state
    - Nodes in the reset state will be rebooted and launcher restarted on them

Condor and IBM Blue Gene Collaboration

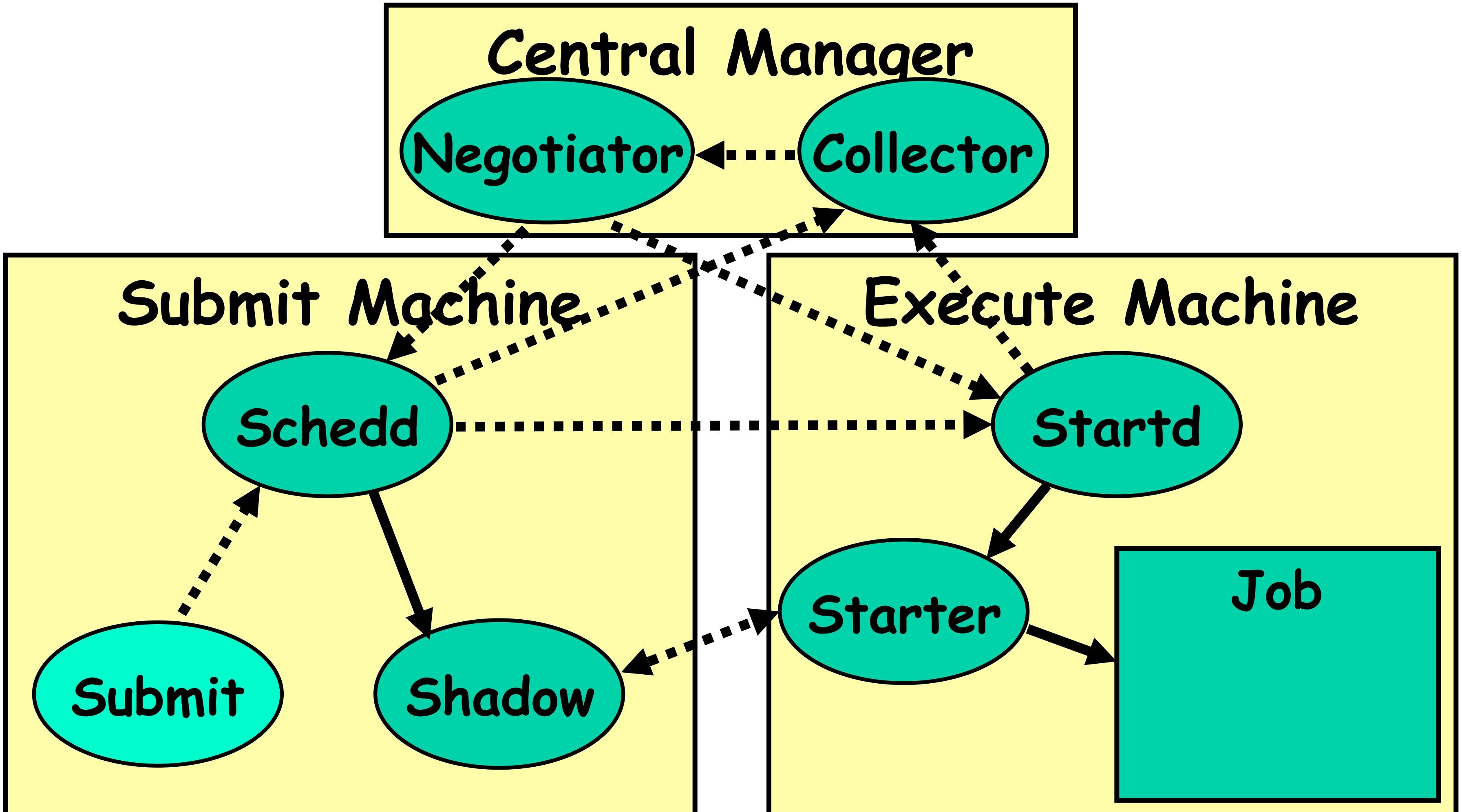
- **Both IBM and Condor teams engaged in adapting code to bring Condor and Blue Gene technologies together**

- **Initial Collaboration (Blue Gene/L)**
  - Prototype/research Condor running HTC workloads on Blue Gene/L
    - Condor developed dispatcher/launcher running HTC jobs
    - Prototype work for Condor being performed on Rochester On-Demand Center Blue Gene system

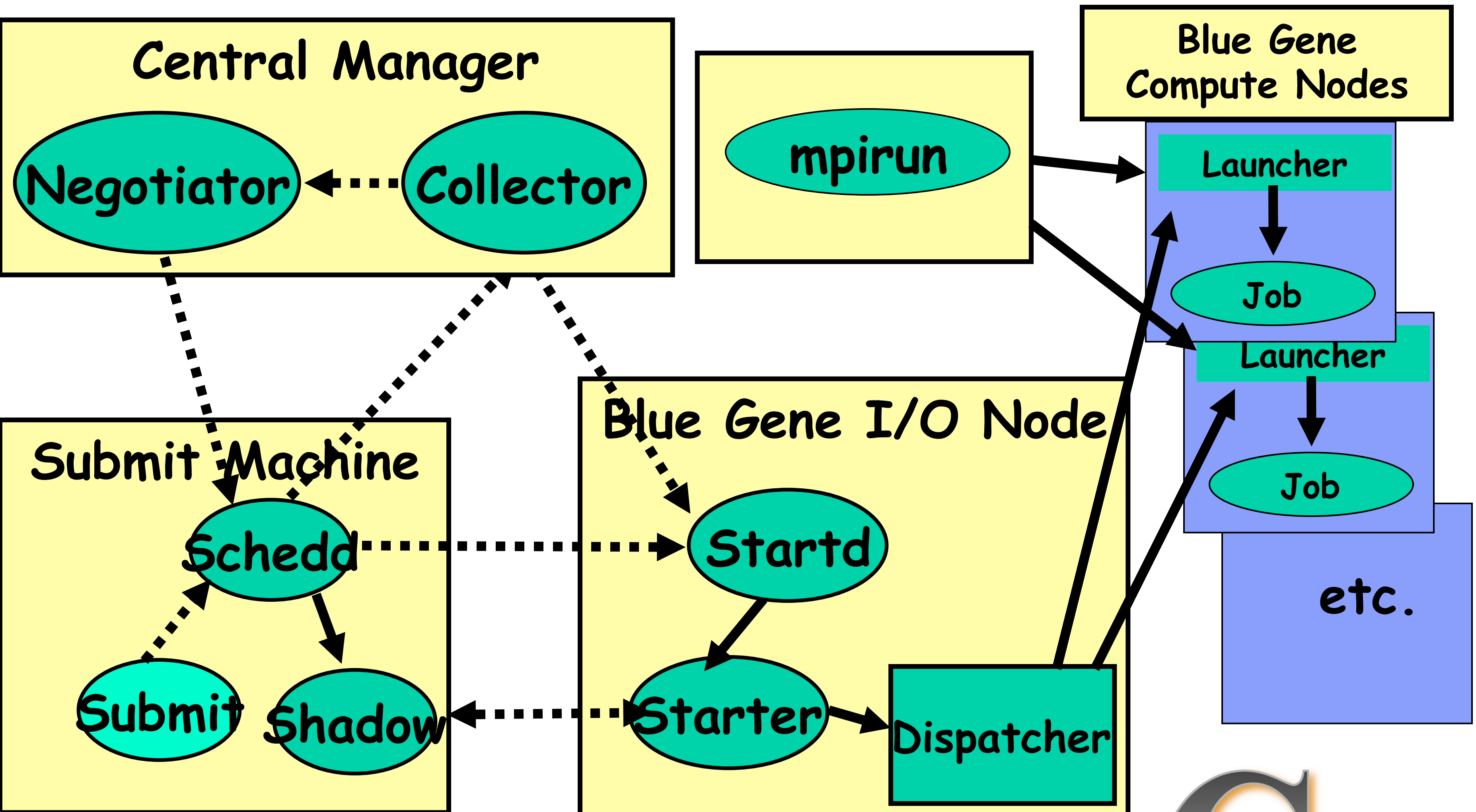
- **Mid-term Collaboration (Blue Gene/L)**
  - Condor supports HPC workloads along with HTC workloads on Blue Gene/L

- **Long-term Collaboration (Next Generation Blue Gene)**
  - I/O Node exploitation with Condor
  - Partner in design of HTC services for Next Generation Blue Gene
    - Standardized launcher, boot/allocation services, job submission/tracking via database, etc.
  - Study ways to automatically switch between HTC/HPC workloads on a partition
  - Data persistence (persisting data in memory across executables)
    - Data affinity scheduling
  - Petascale environment issues

Condor Architecture



Condor with Blue Gene/L



Exploratory Application Case Studies for Blue Gene HTC

- **Case Study #1: Financial overnight risk calculation for trading portfolio**
  - Large number of calculations to be completed by market opening
  - Algorithm is Monte Carlo simulation
    - Easy to distribute and robust to resource failure (fewer simulations just gives less accurate result)
  - Grid middleware bundles tasks into relatively long-running jobs (45 minutes)
  - Limiting resource is number of CPUs
  - In some cases power density (KW/sq foot) is critical
- **Case Study #2: Molecular docking code for virtual drug screening**
  - Docking simulation algorithm for screening large databases of potential drugs against targets
  - Large number of independent calculations to determine the minimization energy between the target and each potential candidate, and subsequently find the strongest leads

Exploratory Application Case Studies for Blue Gene HTC

- **Experience results:**
  - Demonstrated scalable task dispatch to 1000's of processors
  - Successfully verified multiple dispatcher architecture
  - Discovered optimal ratio of dispatcher to partition (pool) size is 1:64 or less
    - Latencies increase as ratio increases above this level, possibly due to launcher contention for socket resource as scaling increases – still investigating in this area
    - May depend on task duration and arrival rates
  - Running in HTC mode changes the I/O patterns
    - Typical MPI programs read and write to the file system with small buffer sizes
    - HTC requires loading the full executable into memory and sending it to compute node
      - Launcher is cached on IO Node but not the executable
    - Experiments with delaying dispatch proportional to executable size for effective task distribution across partitions were successful
      - Due to IO Node to Compute Node bandwidth
    - To achieve the fastest throughput a low compute node to I/O node ratio is desirable

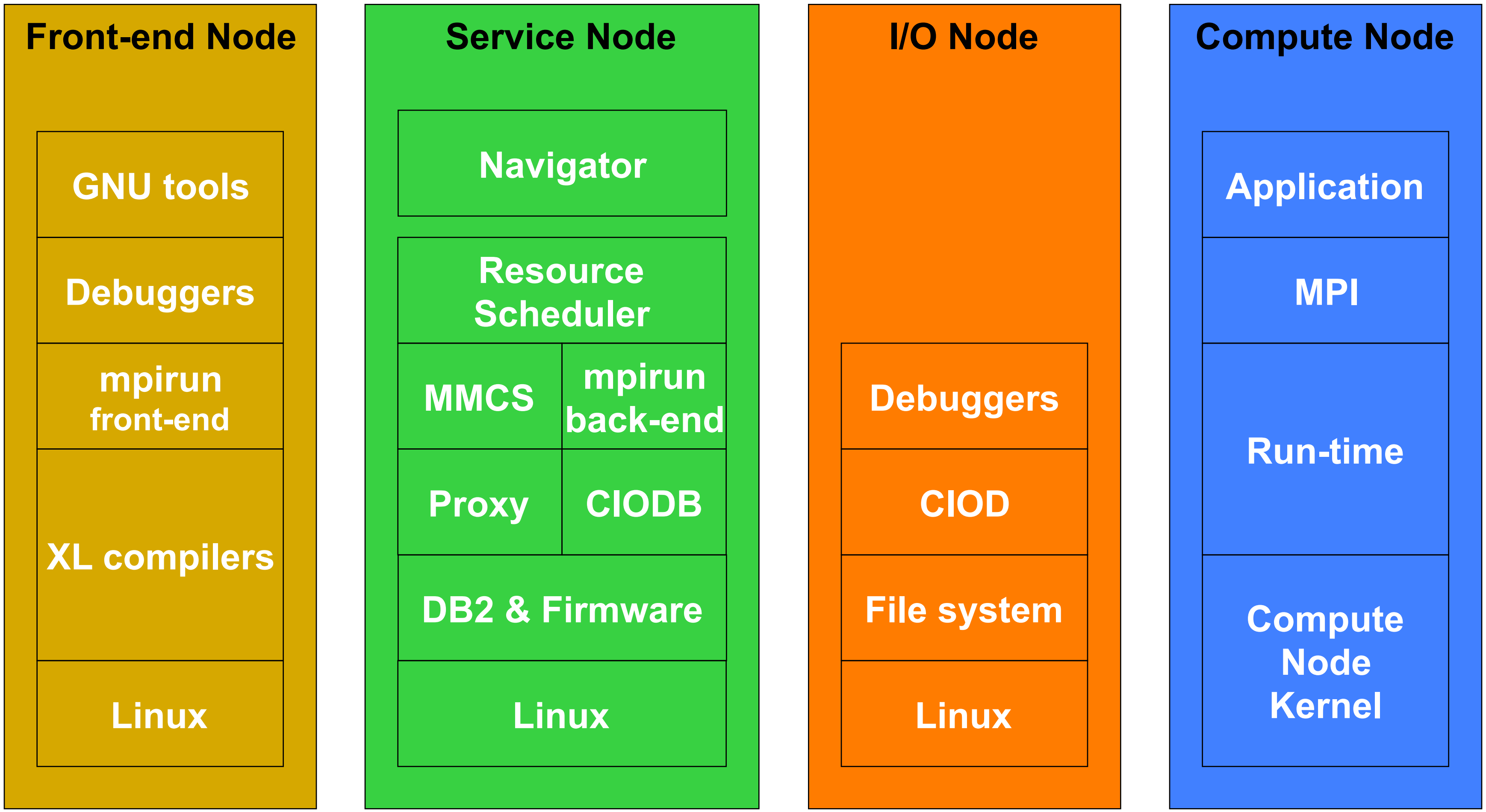
Questions?

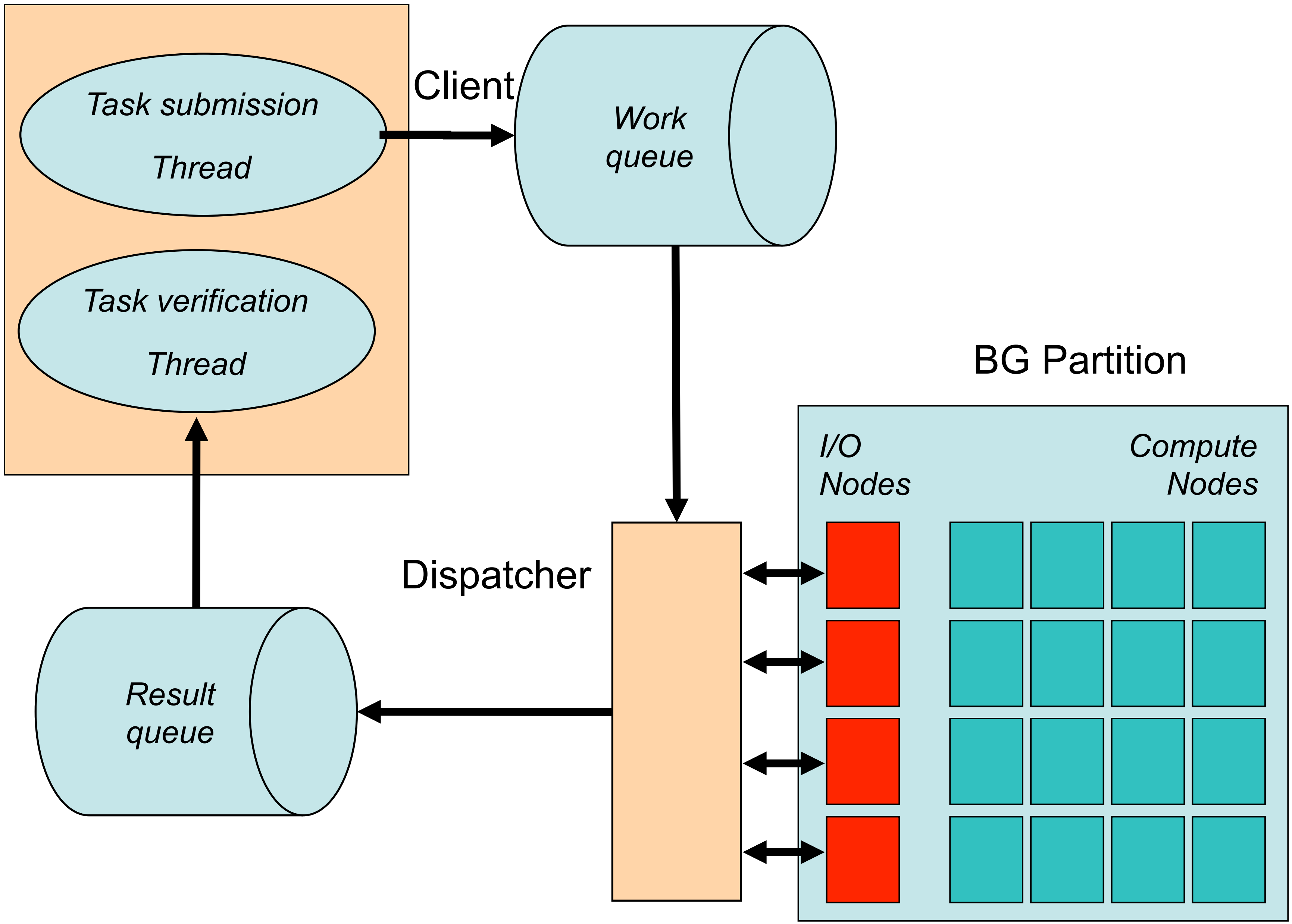
Web resources:

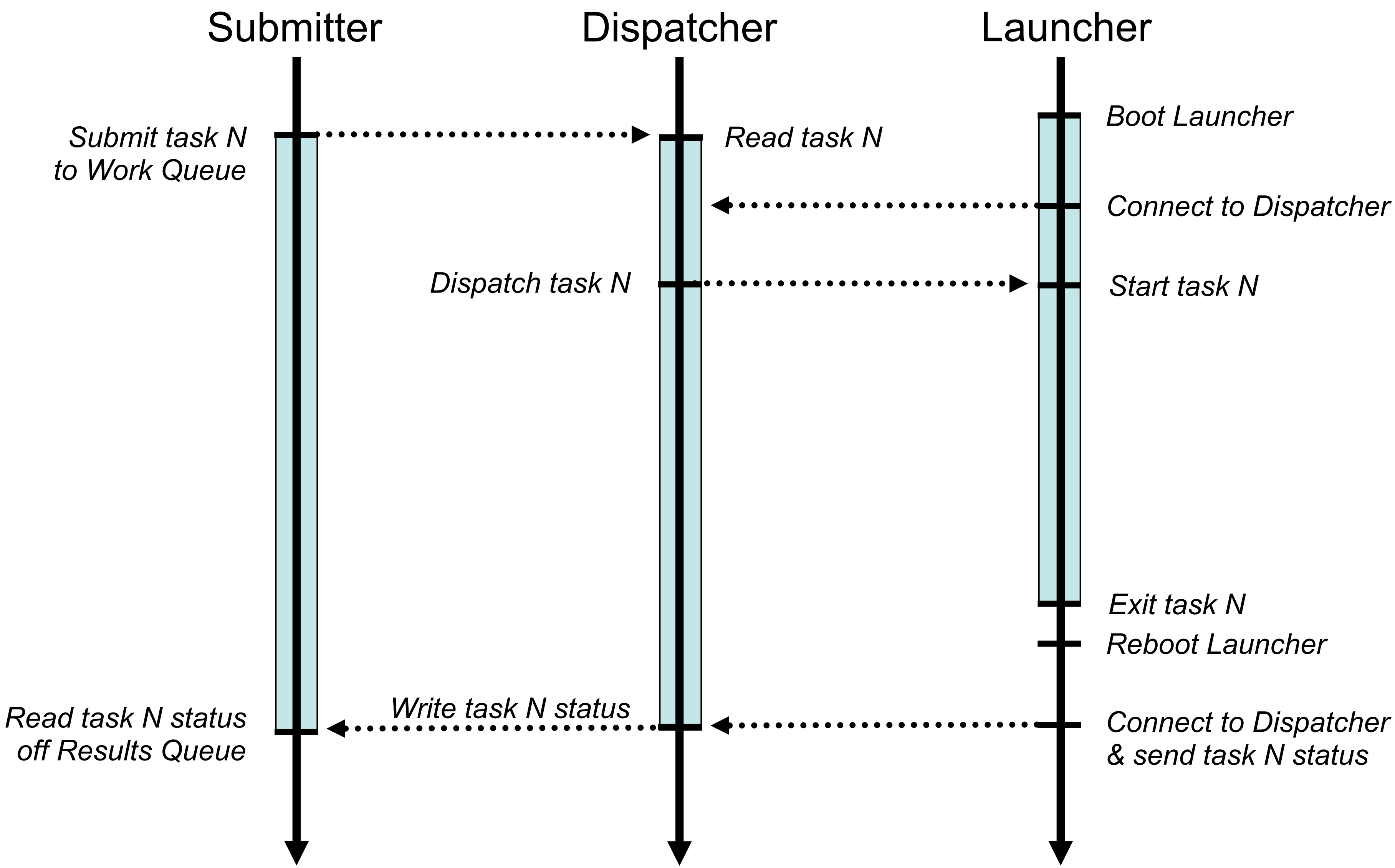
- **http://www.ibm.com/servers/deepcomputing/bluegene.html**
- **http://www.research.ibm.com/bluegene**
- **http://www.redbooks.ibm.com/cgi-bin/searchsite.cgi?query=blue+gene**

Backup Slides

Blue Gene Software Stack




Task submission
Thread
Task verification
Thread
Client
Work
queue
BG Partition
I/O
Nodes
Compute
Nodes
Dispatcher
Result
queue


Submitter
Dispatcher
Launcher
Boot Launcher
Submit task N
to Work Queue
Read task N
Connect to Dispatcher
Dispatch task N
Start task N
Exit task N
Reboot Launcher
Read task N status
off Results Queue
Write task N status
Connect to Dispatcher
& send task N status

Node Resiliency

An Overview of the Architecture of the BlueGene/L Compute Chip (BLC)