

Chiba City

**An Open Source
Computer Science Testbed**



<http://www.mcs.anl.gov/chiba/>
Mathematics & Computer Science Division
Argonne National Laboratory

The Chiba City Project

- Chiba City is a Linux cluster built of 314 computers. It was installed at MCS in October of 1999.
- The primary purpose of Chiba City is to be a *scalability testbed*, built from *open source* components, for the High Performance Computing and Computer Science communities.
- Chiba City is a first step towards a many-thousand node system.

Supercomputing / High Performance Computing

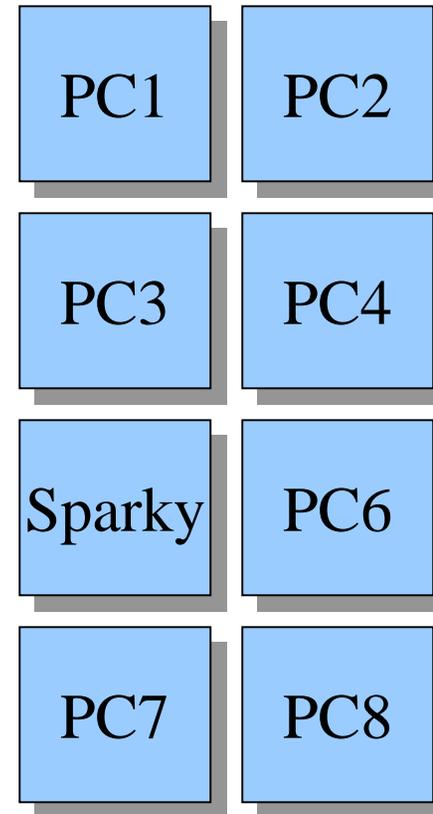
- Supercomputers are used when the problem you need to solve is Big.
 - Scientific simulations: weather, molecular interactions,
 - Business use: mostly data mining and simulation.
 - Spy stuff.
 - Weapons stuff.
- Supercomputing evolution:
 - Crays
 - Parallel vs. Vector Machines
 - Shared Memory vs Distributed Machines
 - Commodity clusters
 - (New architectures)
- But how are they really used?

Joe's Cluster

Joe picks up a used copy of “Extremelinux” and connects a bunch of PCs in his garage.

Using Joe's Cluster:

1. Pick a PC and log in.
2. Edit & compile code.
3. Run the program.
4. Analyze results.



Joe's Cluster - Programming

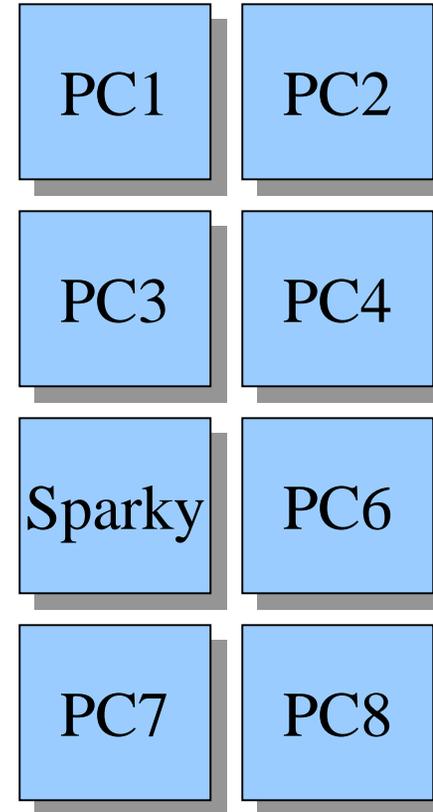
Reality sets in:

Parallel programming isn't easy. You really have to do that yourself.

Network speed matters.

There's a good reason for security patches.

Ugh. PC7 doesn't have floating point.



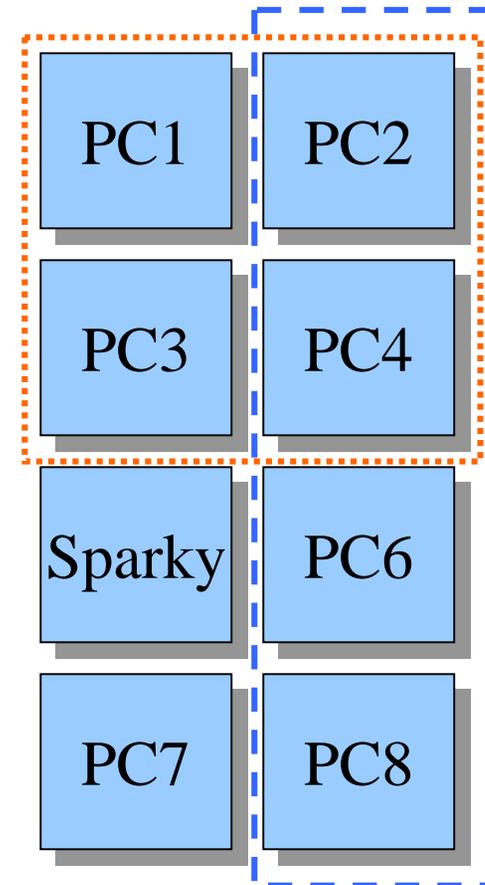
Joe's Cluster Attracts Users

Joe's neighbors want to use the cool cluster. They each only need half... (a half without PC7).

Joe's neighbors discover that using the same PC at the same time is incredibly bad.

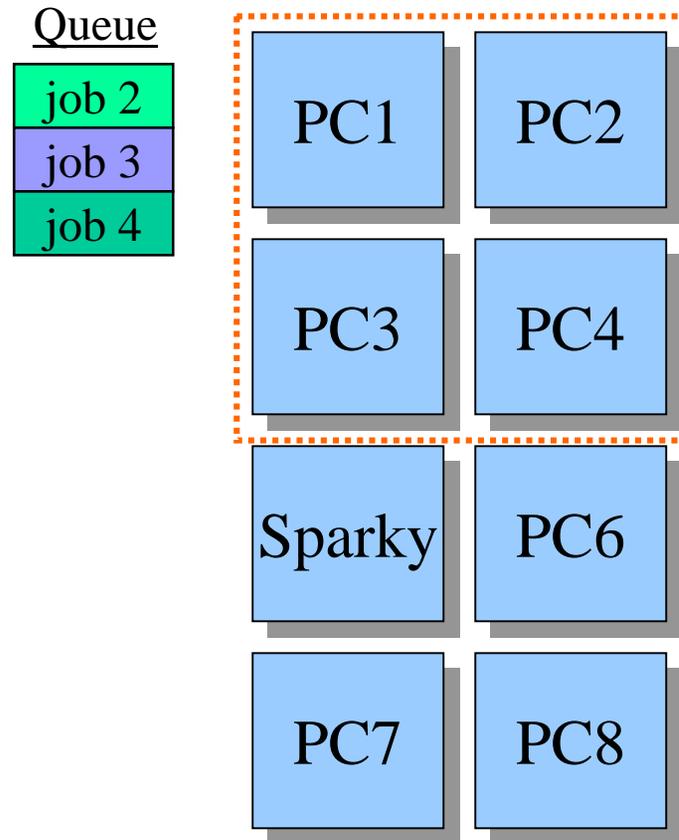
A solution would be to use parts of the cluster exclusively for one job at a time.

And so...



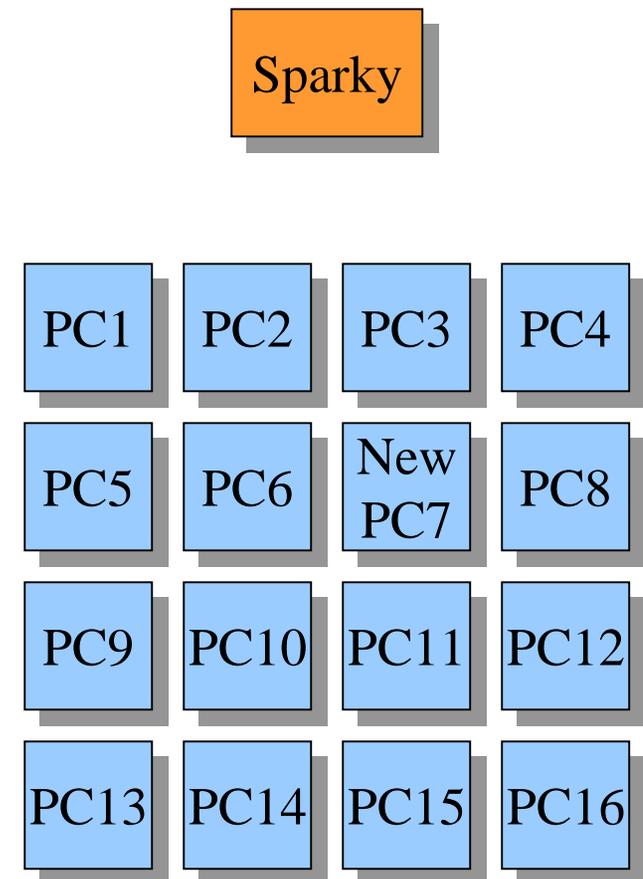
... Joe Discovers Scheduling

- Joe tries:
 - a sign up sheet
 - yelling across the yard
 - a mailing list
 - `finger schedule`
 - cool PHP script that at least looks great
 - ...
 - a program that does scheduling



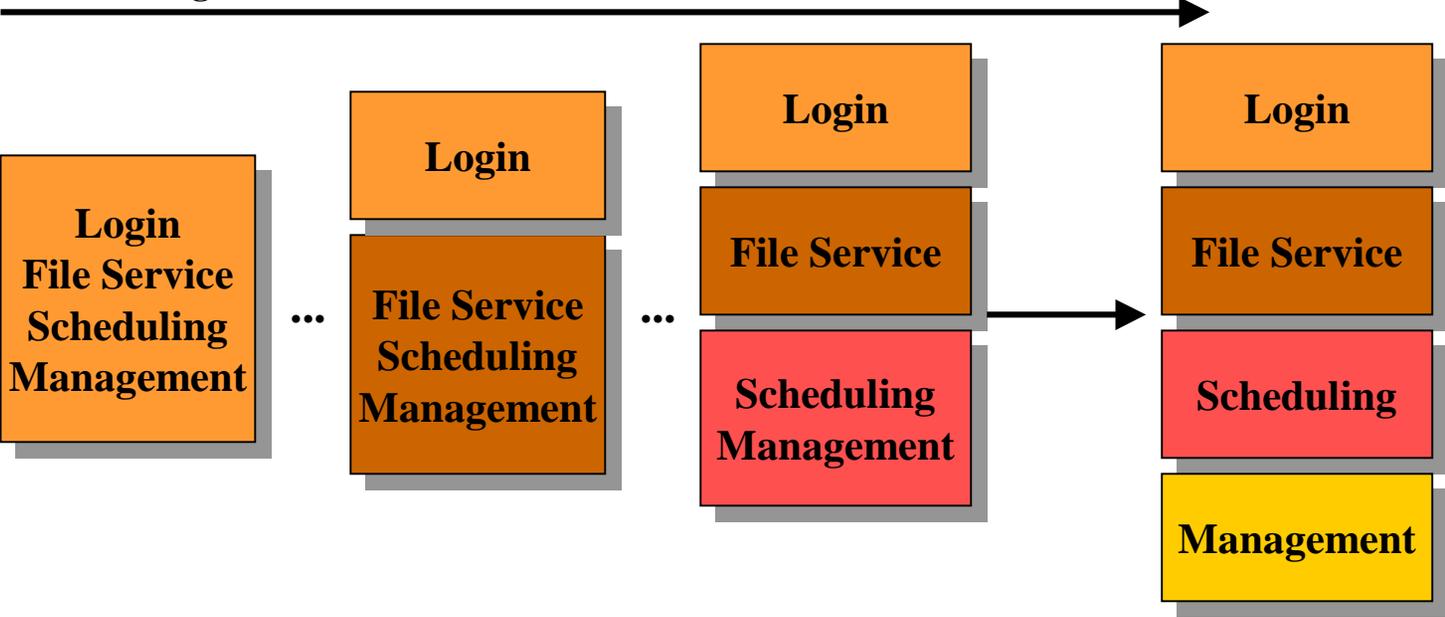
ComputeAtJoes.Com

- Joe expands, adding more users and more systems.
- He uses Sparky for:
 - Somewhere to login.
 - File services.
 - Scheduling services.
 - ...
- All the others are for computing.



Evolution of Cluster Services

The cluster grows...



Basic Goal:

Improve computing performance.



Improve system reliability and manageability.

Server Strategy:

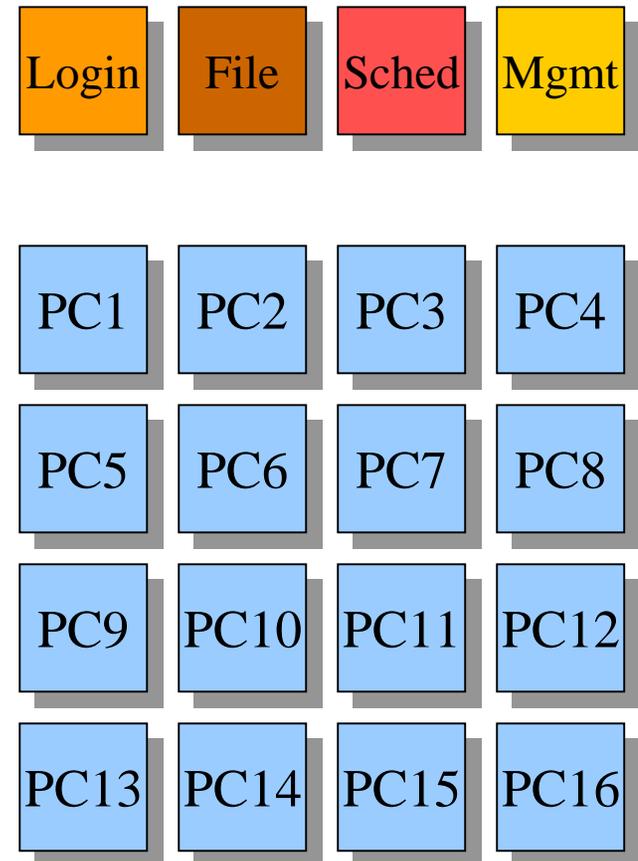
1 computer that is distinct from the computing nodes.



1 or more computers per service.

ComputeAtJoes.Com - The Franchise

- Usage Model:
 - Login to “login” node.
 - Compile and test code.
 - Schedule a test run.
 - Schedule a serious run.
 - Collect data after the run.
 - Analyze the data.
- Management Model:
 - The compute nodes are identical.
 - Run the cluster from a small set of machines.
 - Users use the login and compute nodes.



Using Supercomputers

- Typically it's a scarce resource. Lots of users, 1 machine.
- Thus you operate in batch mode.
 - You login to the "front end".
 - You prepare a job.
 - You put it in a queue of jobs.
 - You wait until it runs.
 - You get your output.
- Interactivity is possible, but difficult to arrange for.
- Programming model is typically message passing. (MPI)
- Performance depends on node speed and interconnect speed.
- One of the most difficult parts is scaling up without losing performance.
- Better tools would help, but computer scientists typically want interactivity and don't have access to the really large systems.
 - Hence Chiba City's goal: scalability and computer science.

Scalability - an Unrecognized Crisis

- Scalability is hard:
 - Complexity of solutions.
 - Fault tolerance.
 - Understanding program behavior is hard - huge amounts of data.
 - Lack of available scalability testbeds.
- Scalability research is important:
 - Improve real performance of applications.
 - The size of the average system is growing. We need better scalability now. Everyone will need it in 3-5 years.
- To have a scalable system, we need scalable...
 - ... algorithms.
 - ... development tools.
 - ... middleware.
 - ... systems administration methods.

What Facility Support Does Computer Science Need?

- Interactivity
 - Edit, Compile, Run, Debug/Run, Repeat.
 - In many cases those runs are very short and very wide.
- Flexible Systems Software
 - A specific OS, kernel, or a specific set of libraries and compilers.
 - (Which frequently conflict with some other user's needs.)
 - Re-configurable hardware.
 - Access to hardware counters.
 - Permission to crash the machine.
 - In some cases, root access.
- Ability to test at Scale
- Non-requirements
 - Exclusivity. "Performance" is an issue, but typically only on timing runs.

The Motivation Behind Chiba City

- Scalability Testbed
 - As part of the call to action.
- Open Source
 - Take advantage of the fact that the market had just discovered how the research community has released software for decades.
 - Expand our mission beyond supporting computational science to supporting open source system software.
- Computer Science Support
 - Explore why this issue is difficult and try to fix it.
 - Re-unite the division with the large divisional computing facilities.
- Computational Science Support
 - Continue to work with our research partners.
 - Continue to apply the results of our CS work to scientific problems..

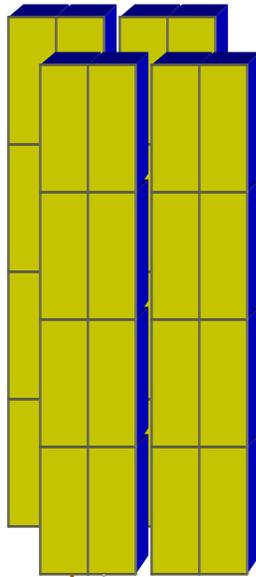
Building Chiba City

- We designed Chiba City ourselves.
 - The design was based on all sort of things:
 - Existing cluster work - Sterling's Beowulfs, Sandia's C-Plant, LANL's Avalon, and our own work on smaller clusters.
 - Our experiences with the SP and large sets of workstations.
 - Analysis of needs of the CS community.
 - Optimization of resources: \$\$, machine room space, computing speed...
 - The purchasing process was complicated, taking nearly a year.
 - No vendors were selling anything close to what we needed at the time.
 - Eventually, the cluster was purchased from IBM, with components from VA Linux, Myricom, Cisco, and others.
 - We dragged IBM into the cluster and open source world.
 - VA Linux started selling a cluster product based on our design.
- And then we built it ourselves, with a little help from VA.

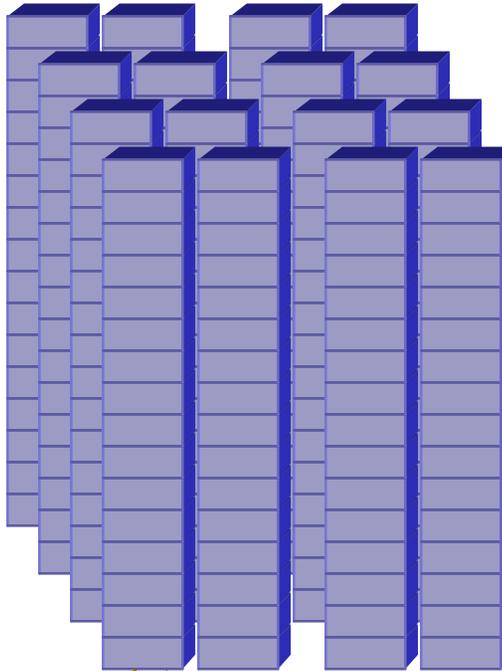
Chiba City

The Argonne Scalable Cluster

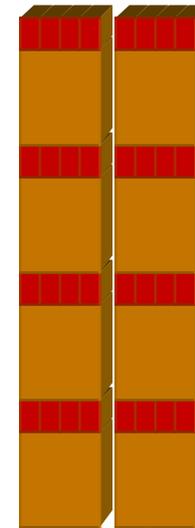
1 Visualization Town
32 Pentium III systems
with Matrox G400 cards



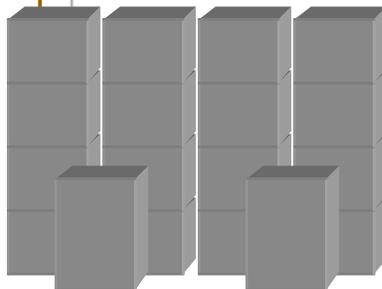
8 Computing Towns
256 Dual Pentium III systems



1 Storage Town
8 Xeon systems
with 300G disk each



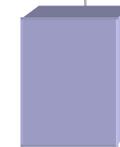
Cluster Management
12 PIII Mayor Systems
4 PIII Front End Systems
2 Xeon File Servers
3.4 TB disk



High Performance Net
64-bit Myrinet

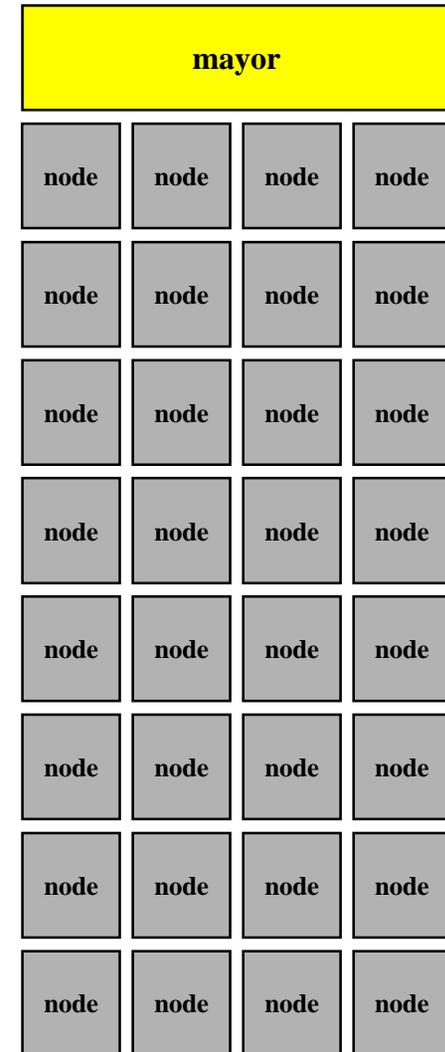


Management Net
Gigabit and Fast Ethernet
Gigabit External Link

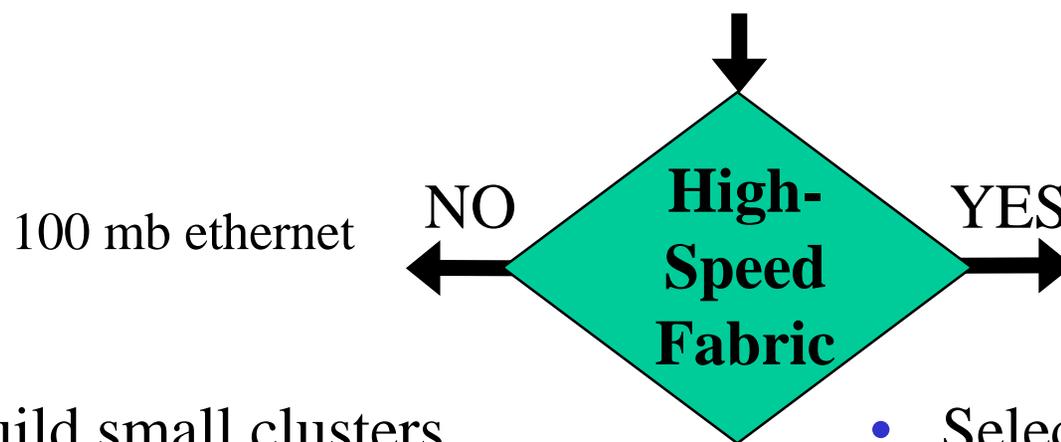


Chiba Computing Systems

- A “town” is the basic cluster building unit.
 - 8-32 systems, for the actual work.
 - 1 mayor, for management.
 - OS loading, monitoring, file service
 - Network and management gear.
- 8 compute towns:
 - 32 dual PIII 500 compute nodes that run user jobs.
- 1 storage town:
 - 8 Xeon systems with 300G disk
 - For storage-related research, eventually for production global storage.
- 1 visualization town:
 - 32 nodes for dedicated visualization experiments.



Interconnect - High speed or Low cost?

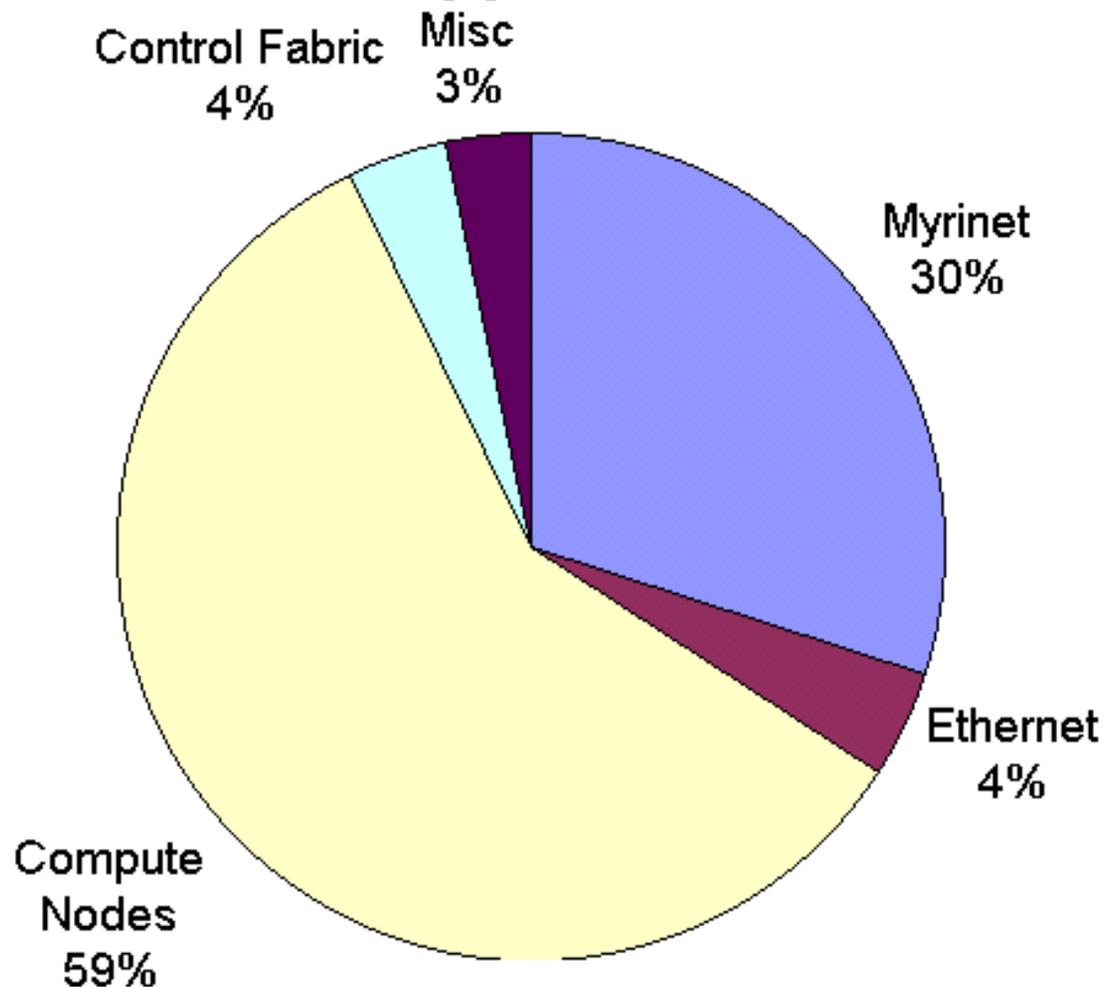


- Build small clusters
- Do mostly embarrassingly parallel computation

- Select scalable tech:
 - Myrinet, GigE, Giganet, Quadrics, etc
 - Expect order of magnitude improvement.
- Evaluate fast MPI
 - SMP?
- Open wallet, listen to giant sucking sound.

Bandwidth is Never Cheap

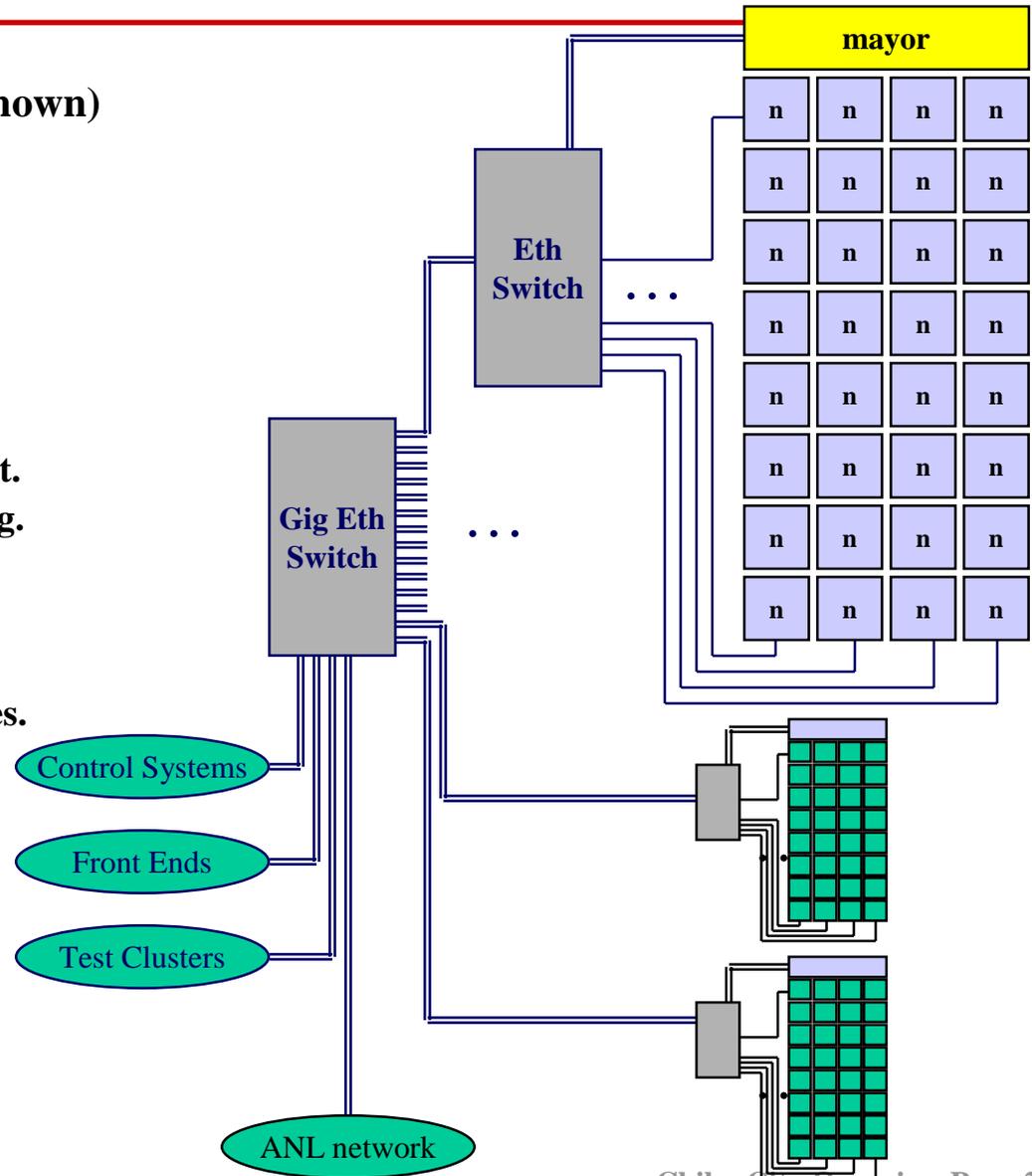
ACL: Rockhopper Hardware Costs



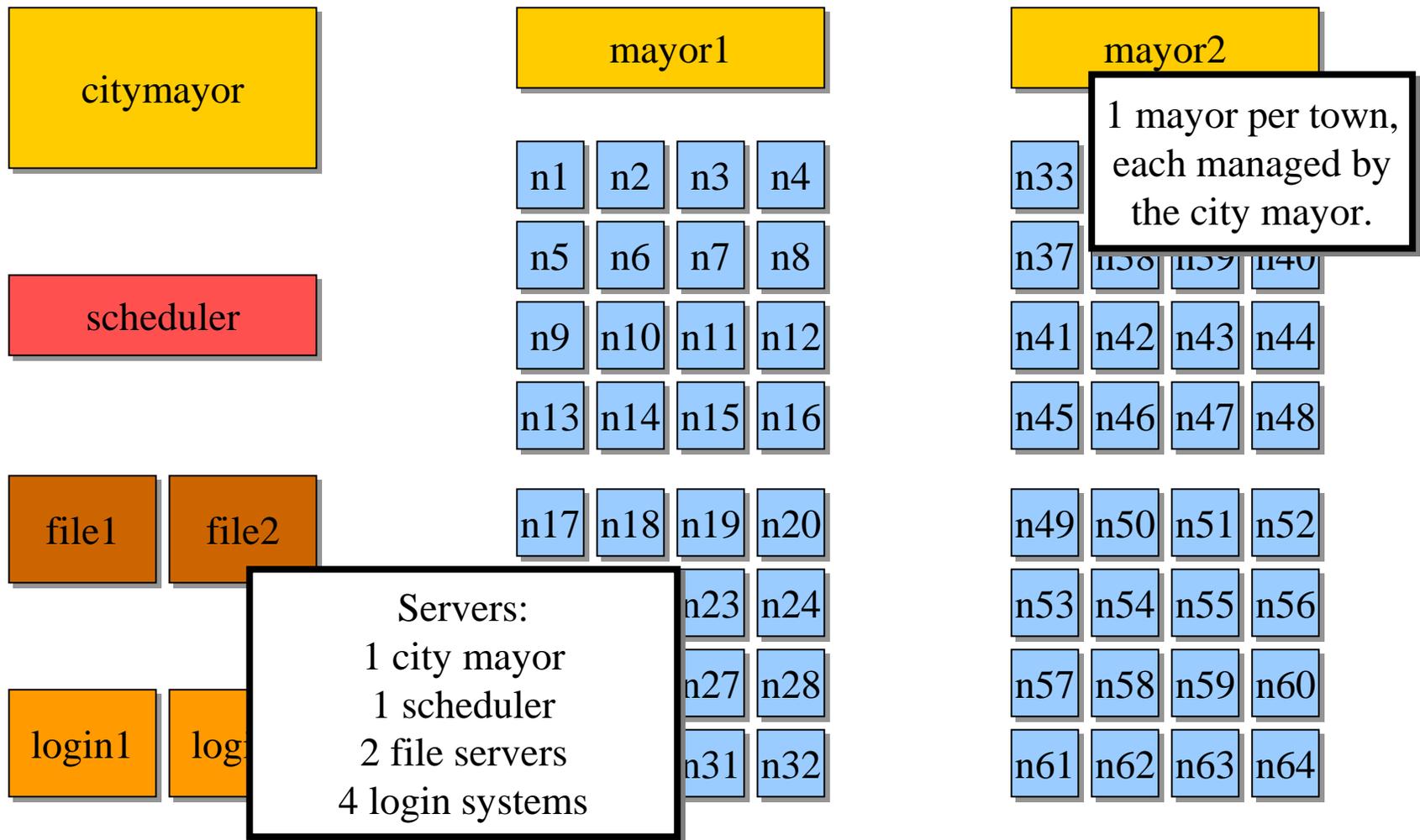
Chiba Networks

- **High Performance Network (Not Shown)**
 - 64-bit Myrinet.
 - All systems connected.
 - Flat topology.

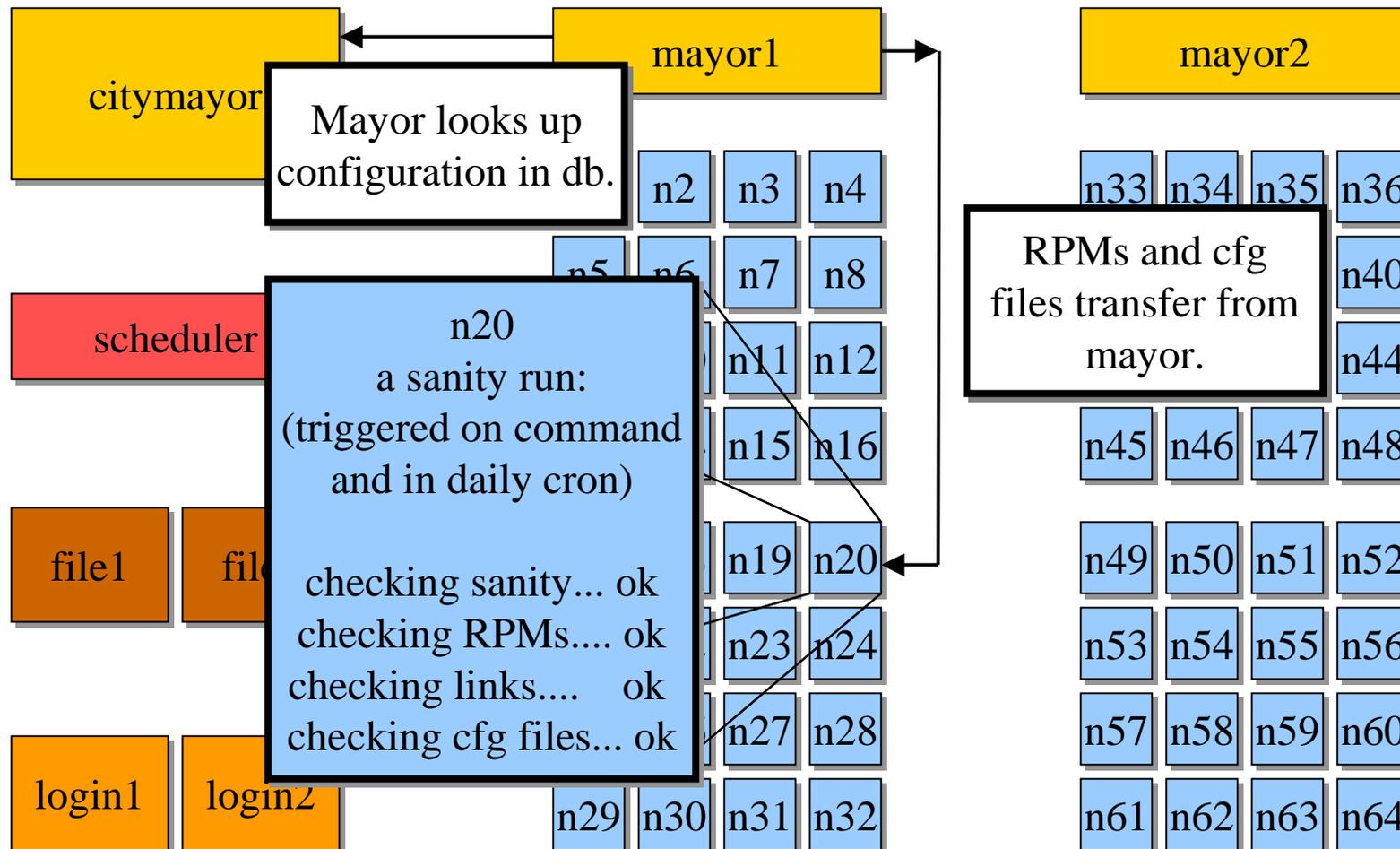
- **Management Network**
 - Switched Fast and Gigabit Ethernet.
 - Primarily for builds and monitoring.
 - Fast ethernet:
 - Each individual node.
 - Bonded gigabit ethernet:
 - Mayors, servers, & login nodes.
 - Town interconnects.
 - External links.
 - IP Topology:
 - 1 flat IP subnet.



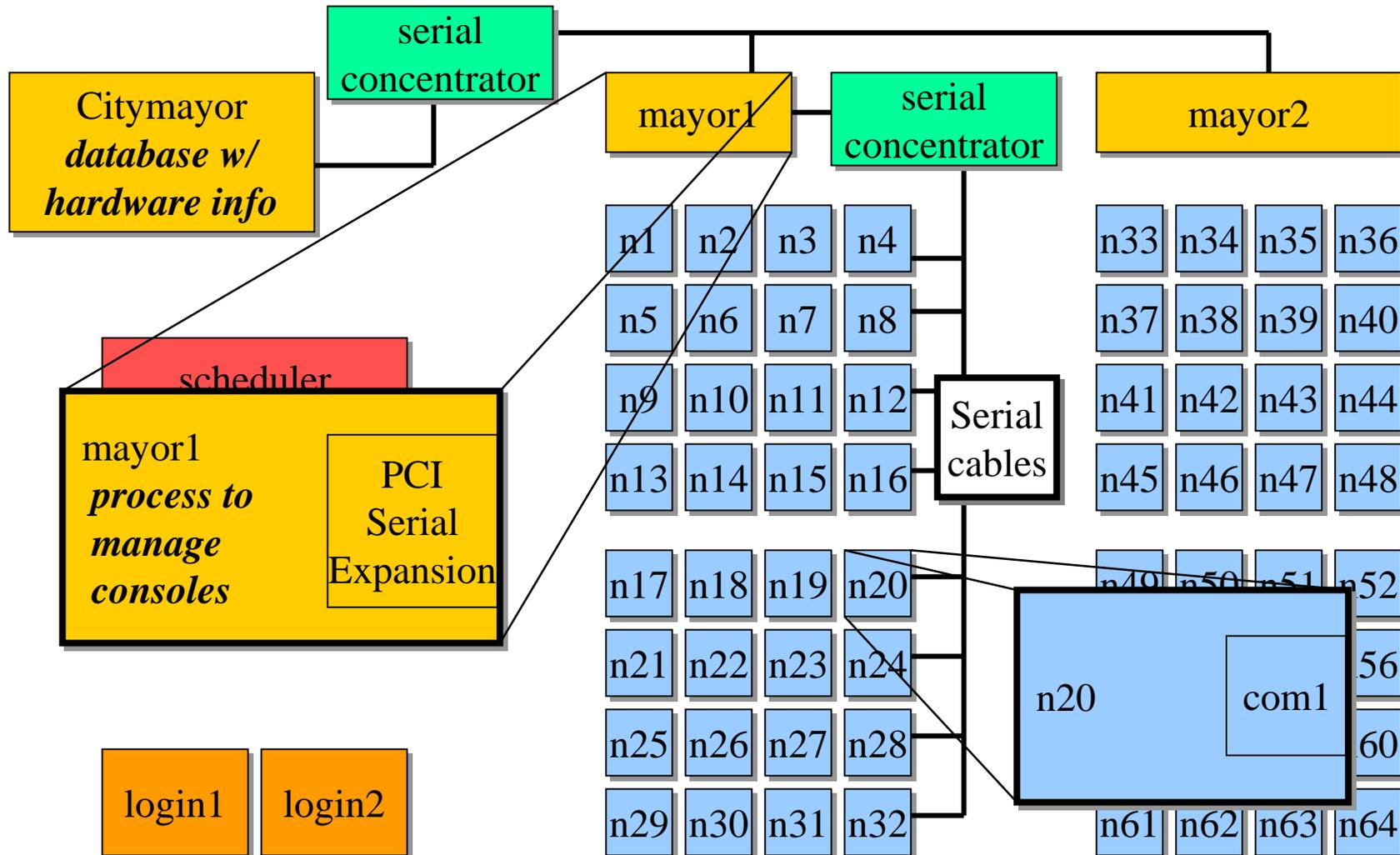
The Chiba Management Infrastructure



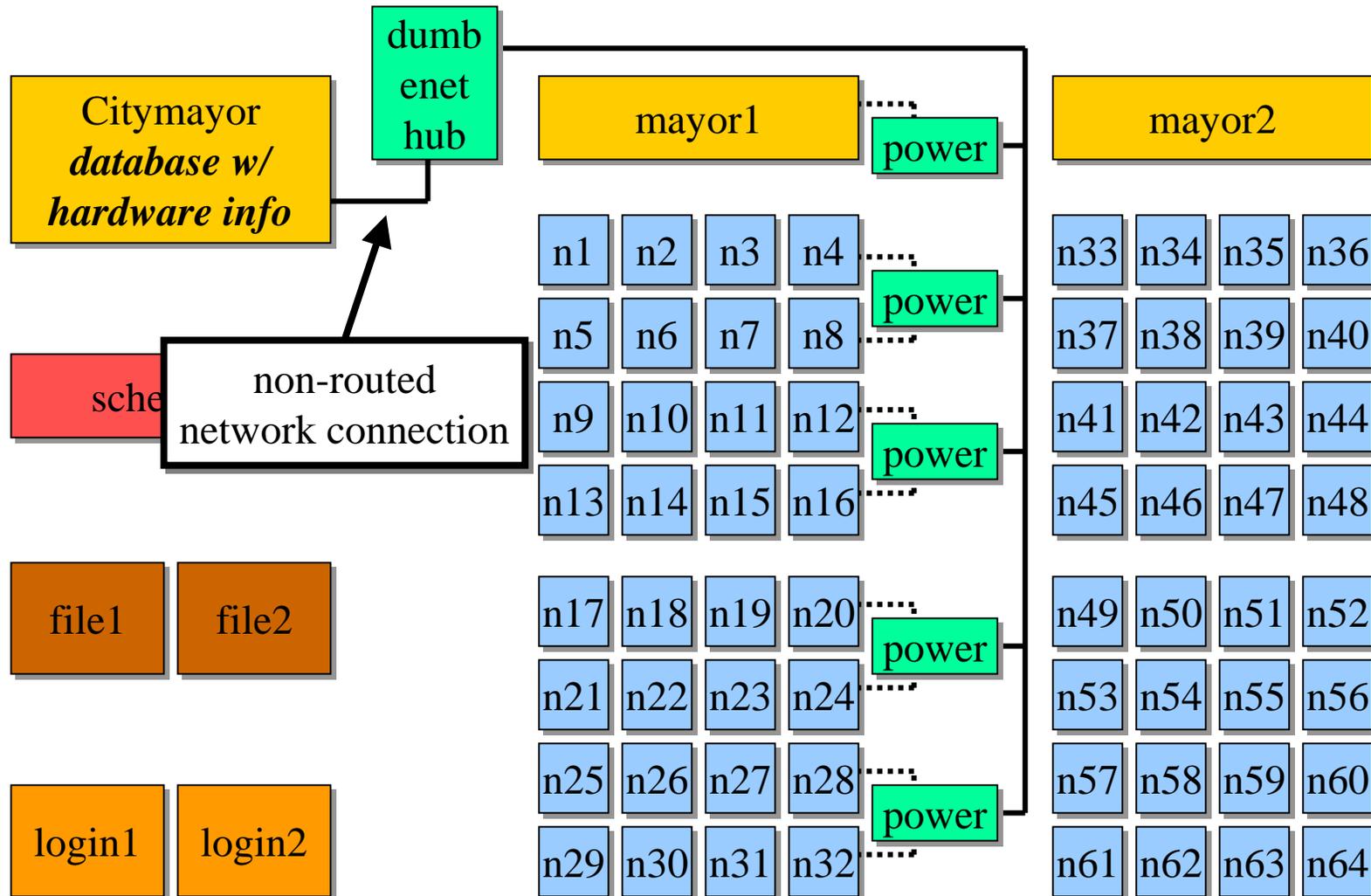
OS Image Management



Serial Infrastructure



Power Infrastructure

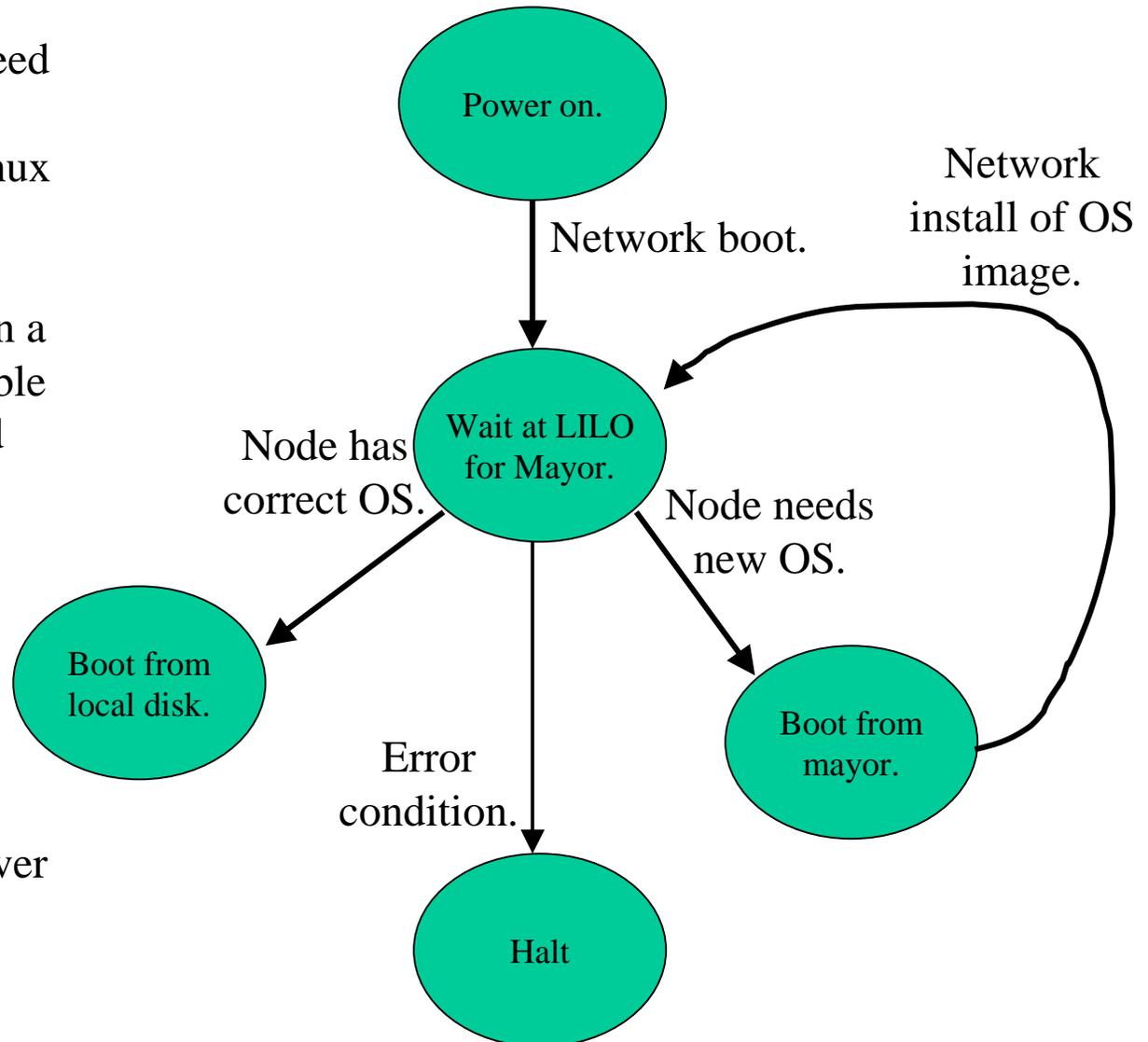


Node Image Management

Some users of Chiba City need to install their own OS, ranging from a modified Linux to Windows2000.

The mayor decides, based on a database of nodes and possible images, which image should be installed on a node. It installs the image via a controlled boot process.

All that is necessary to recover a node is to power cycle it.



Chiba Software Environment

- Default node OS is Linux, based loosely on RedHat 6.2.
 - All the usual pieces of Linux software.
- Programming model is MPI messages, using MPICH with GM drivers for Myrinet.
- C, C++, and Fortran are the primary languages in use.
- Job Management:
 - PBS resource manager, Maui scheduler, MPD for launching
- No production shared file system at present.
 - ... getting very close, though!

Chiba City Time Line

- Phase I – Install and Learning: October 1999 – May 2001
 - Installation
 - Management software development
 - Early user testing
 - Many lessons learned - memory, BIOS, stress factor, ...
- Phase II – Production Support while developing: June 2001 – December 2001
 - Quantum Physics
 - Climate Simulation
 - Communications Libraries
 - Many others
- Phase III – Focus on Large-scale and CS Projects: January 2002+
 - Scalable Systems Software
 - Corporate interaction: Myricom, Scyld, Linux Networx, IBM
 - Many large-scale jobs

Learning Experiences

- Barnraising:
 - Building these things by hand with lots of volunteers is fun - our scarcest resource was space.
 - If you have remote power control, make sure you have your volunteers install the power cables correctly...
- Configuration
 - The hierarchical, database-driven approach has worked very well.
 - Remote power and remote console are awesome.
- Pain:
 - Replacing all the memory in the cluster.
 - Upgrading the BIOS on every node.
 - We stress hardware far more than vendors do - AGP lossage, memory lossage, PCI card lossage...
- Scalability Challenges
 - Myrinet
 - Took a little while for us to get all of the nodes using myrinet happily (early driver releases, mapper, ...)
 - Very small error rates can kill in the large.
 - RSH
 - RSH is used by default to launch jobs, but can only invoke 256. Boom.
 - Network gear
 - Get very confused when 32 nodes all try to boot through them at once.
 - PBS uses UDP for internal communication. UDP loses badly in big, congested networks.

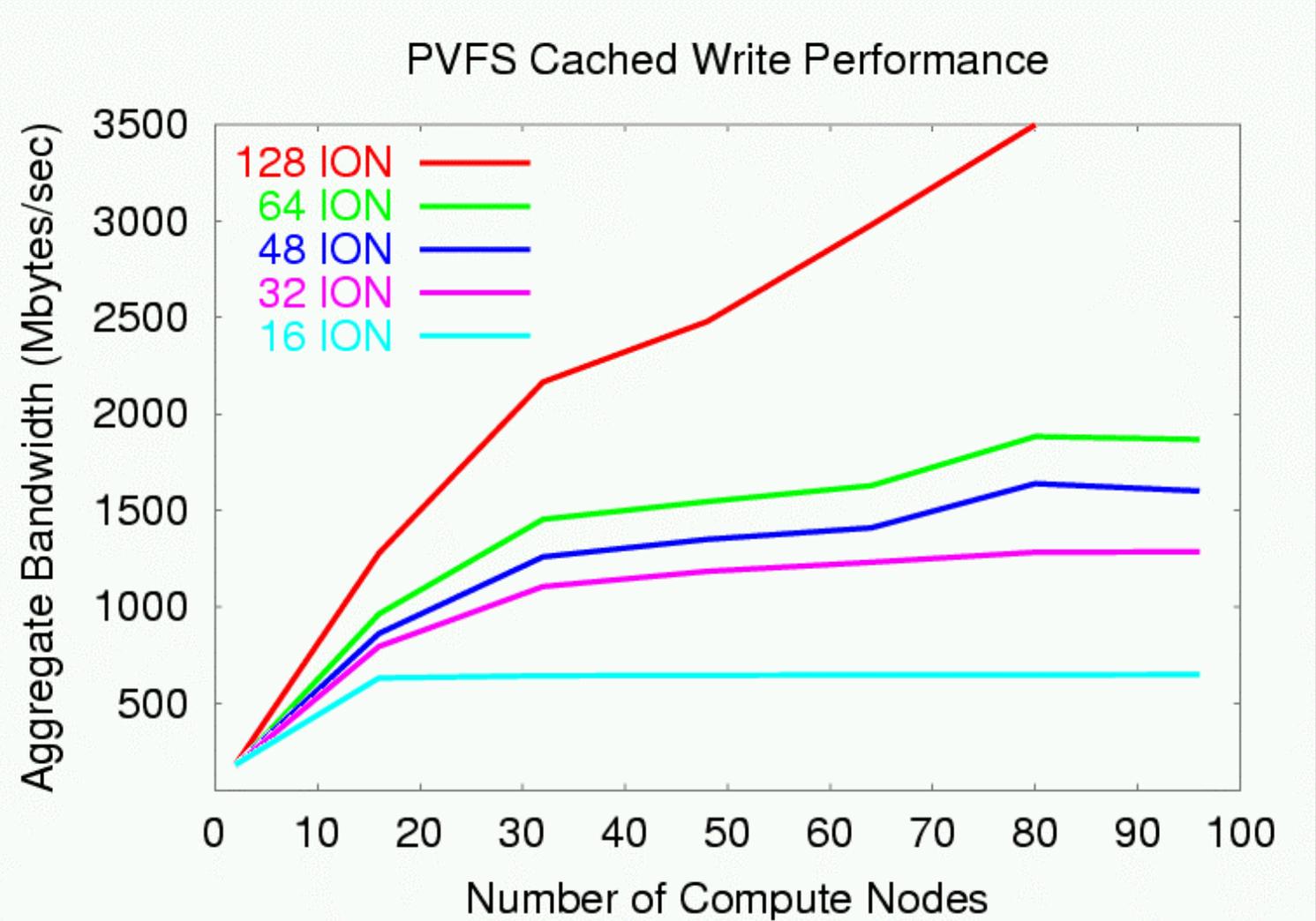
Computer Science Activities

- System Software and Communication Software
 - PVFS - a parallel file system
 - MPD – a scalable MPI job handling daemon
 - MPICH development
 - Minute Sort
- Data Management and Grid Services
 - Globus Services on Linux (w/LBNL, ISI)
- Visualization
 - Parallel OpenGL server (w/Princeton, UIUC)
 - vTK and CAVE Software for Linux Clusters
 - Scalable Media Server (FL Voyager Server on Linux Cluster)
 - Xplit - distributed tiled displays
- System Administration
 - Practical Scalability Tests
 - Myrinet Scaling
 - Scyld Beowulf Scaling
 - Scalable Management Tools
- Many other MCS Research Projects

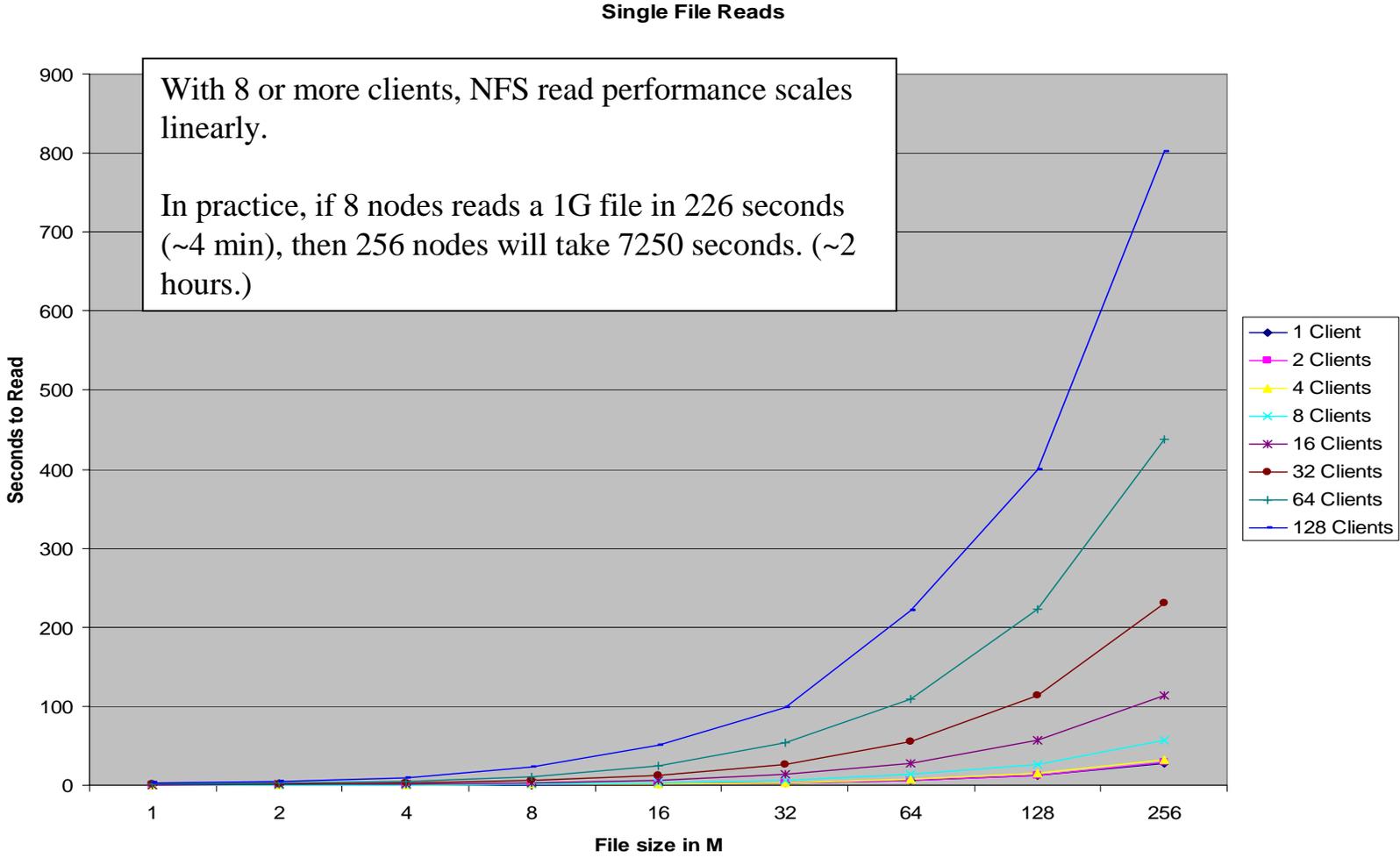
PVFS – an open source parallel file system

- Parallel file systems are used to allow multiple processes to simultaneously read and write files – typically very large files.
- PVFS is widely used in the cluster world.
- Latest benchmarks:
 - Using 48 I/O Nodes, a single 20GB file, and reading on 112 nodes:
 - 3.1 GB/sec writes
 - 2.9 GB/sec reads
- We're beginning to use PVFS as part of the production infrastructure.

PVFS Performance



NFS Single File Read Timings



MPD – the MPICH multi-purpose daemon

- MPD is an experiment into the architecture for job management.
 - Job launching.
 - Signal propagation.
 - Other job management functions.
 - Dual-linked ring topology for speed and reliability.
- Performance.
 - Can currently launch 100 processes/second
 - 2.6 seconds from pressing return on a front-end node until all processes have started
 - Have tested up to 2000 processes.
- It started as an experiment, but has become a critical part of the production infrastructure.

Scyld Beowulf System Scaling

- Scyld (www.scyld.com) is a small company founded by Donald Becker, one of the original developers of the Beowulf system.
- Scyld Beowulf is a new model for running clusters:
 - 1 single process space across the entire system.
 - Cluster nodes are not directly accessible and do not need to be managed.
 - Compared to an SMP:
 - Shares the same “single system image” of an SMP system.
 - Uses message passing rather than shared memory.
- Work on Chiba City includes:
 - Testing Scyld at large scale.
 - Providing a Myrinet test environment.
- All results are open source, thus this advances the cluster field.

The Msys and City Toolkits

Msys

A toolkit of system administration programs, including:

- `cfg` - centralized management of configuration files
- `sanity` - a tool for automatic configuration checking and fixing
- `pkg` - tools and mechanisms for flexible software installation
- `softenv` - tools for setting up the user's environment that adapt to software changes
- `hostbase` - a database of hostname information and scripts for driving all name-related services
- `clan`, `whatami` - utilities
- `anlpasswd` - a `passwd` replacement that catches guessable passwords

City

Cluster-specific tools that build on top of Msys.

- `chex` - the node console management system
- `citydb` - the database used to manage `cfgs`
- `chex` - the node management system
- `city_transit` - file distribution scripts
- filesystem images
- power utilities
- account management

Msys and City are both open source.

Both toolkits are available at:

<http://www.mcs.anl.gov/systems/software/>

Computational Science Updates

- We're slowly adding users over time. Right now there are about 60 groups with access to it.
- Primary users:
 - Quantum physics calculations
 - First ab initio computations of 10-body nuclei.
 - Optimization computation
 - The metaNEOS project solved the NUG30 quadratic assignment problem (an event reported in many press outlets).
 - The ASCI FLASH project
 - Material sciences
 - Computational biology
 - Climate simulation

Future Cluster-Related Plans

- Chiba II
 - To be installed this summer.
- TeraGrid
 - A joint project with NCSA, SDSC, and Caltech.
 - 4 large clusters will be operated in a tightly-interwoven grid.
 - The ANL cluster will have 256 IA-64 (McKinley) nodes.
 - Install begins in August of 2002.
- ANL Computation Cluster
 - A production cluster of 256 nodes focused on computational science.
 - Purchase likely to be complete by June.
- Multi-DOE Lab PetaFlops computing initiative
 - Just ramping up.

Scalable Cluster Planned Capabilities

	Chiba I	Chiba II	Chiba III	Chiba IV
Year	1999	2002	2004	2006
Nodes	256	1024	2048	4096
CPUs	512	1024	2048	4096
Virtual nodes	256	8192	40960	163840
FLOPs/cpu	500MF	2 GF	3.5 GF	6.3 GF
RAM/cpu	256MB	1 GB	2.5 GB	5 GB
Myrinet BW	100MB	200MB	500MB	1GB
Storage/node	9GB	100MB	400GB	1TB

Chiba II System Components

- User Nodes
 - 1024 nodes.
 - Single 2.4 GHz Pentium IV. 1G RAM. 100G IDE disk.
- Virtual Computing Subsystem
 - 8 virtual nodes / node (using VMware and/or USL): 8192 nodes
 - Virtual nodes are independently addressable and schedulable, each having access to 1/8 the total system resource (I.e. 1/8th of Myrinet, 128M of RAM, etc)
- Network
 - Myrinet 2000, 1 NIC / compute node.
 - Fast ethernet.
- Persistent Storage: 18 TB
 - 18 servers.
 - Dual 2.4 GHz Pentium IV. 1G RAM. 1 TB SCSI disk.
 - Note that user nodes can be configured as an I/O system.
- Reliability and Management Subsystem
 - 16 servers for configuration management and infrastructure services.
 - Dual 2.4 GHz Pentium IV. 1G RAM. 1 TB SCSI disk.
 - Remote console, power, monitoring.

Food for Thought

- How would you build a cluster with one million CPUs?
 - Would you really build a “cluster”?
- What would the software environment look like?
- How would you program it?
- Assuming you didn't have the money to build such a thing, how would you simulate it?

Chiba City

**An Open Source
Computer Science Testbed**



<http://www.mcs.anl.gov/chiba/>
Mathematics & Computer Science Division
Argonne National Laboratory