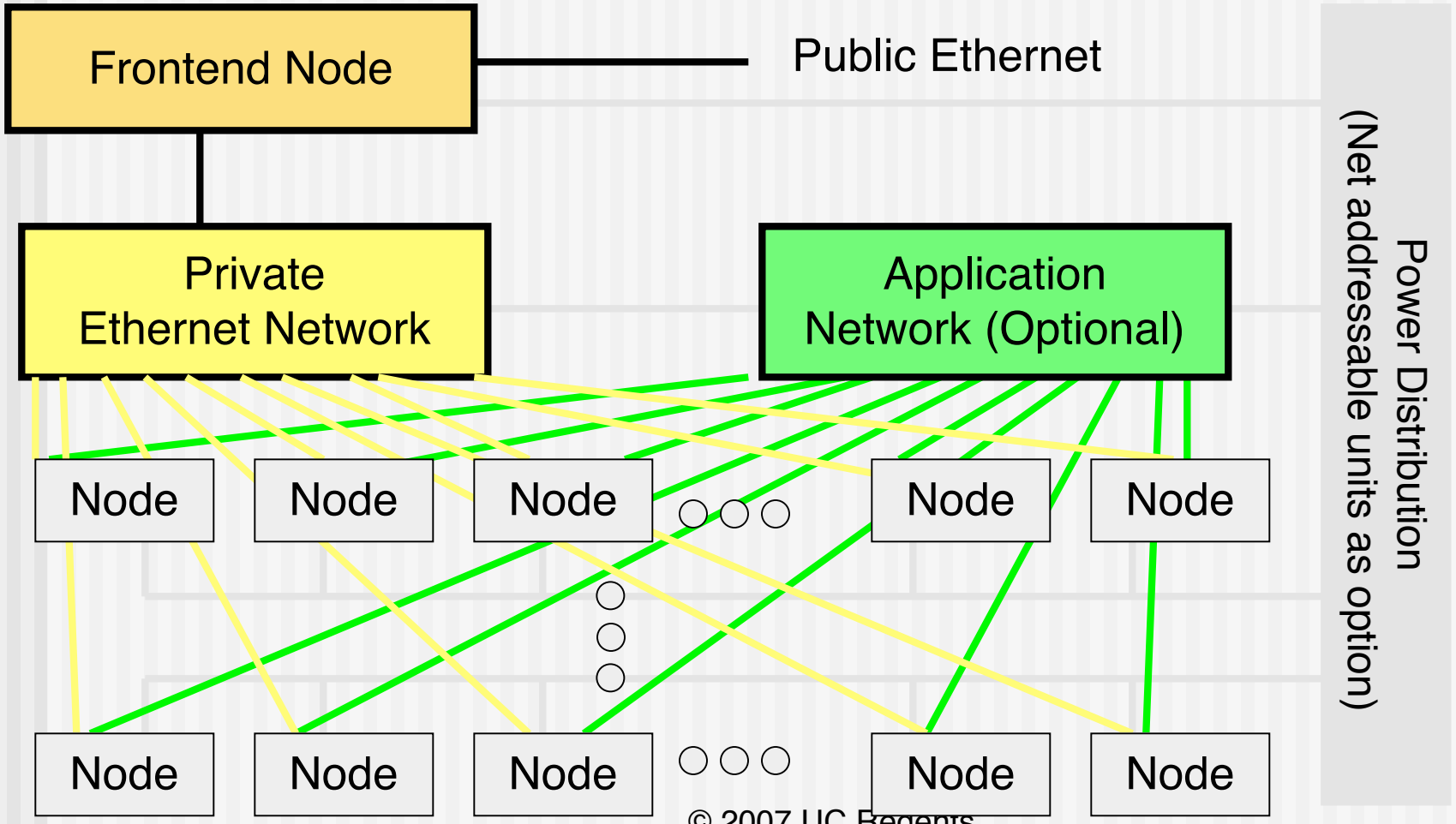
User Session 2
Introduction to Rocks

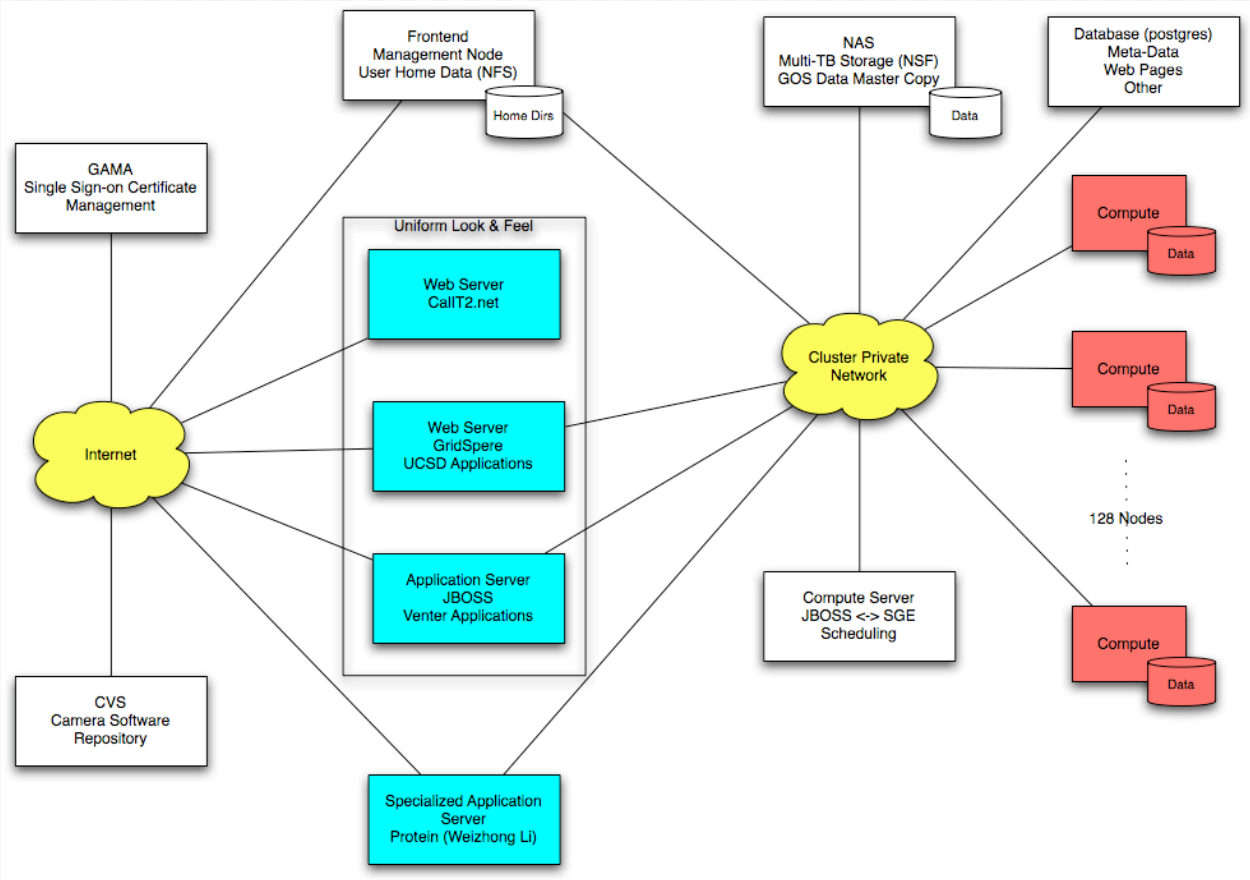
Rocks-A-Palooza III

SDSC

Traditional Cluster Architecture

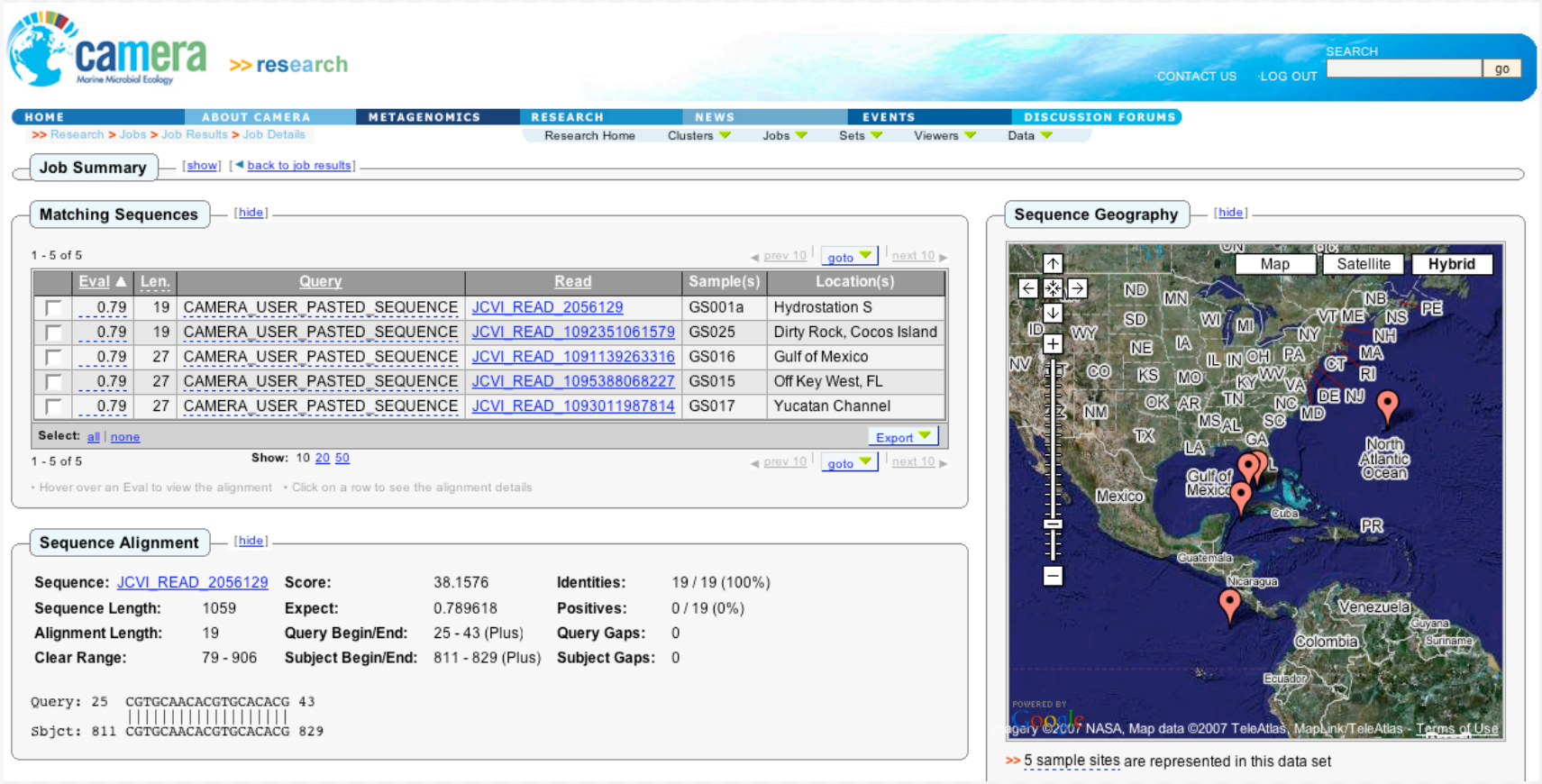


Modern Cluster Architecture



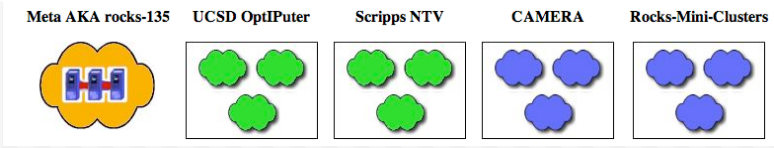
User View of Cluster

www.camera.calit2.net



Visualization Clusters

- Cluster of GPUs
  - OpenGL machine
  - Not an MPI machine
- Massive Pixel Walls
  - 60 MegaPixels
  - Full rate HDTV
- Software
  - SAGE
  - DMX
  - Chromium



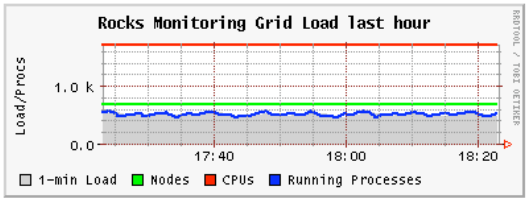
Diverse Grid End-Points

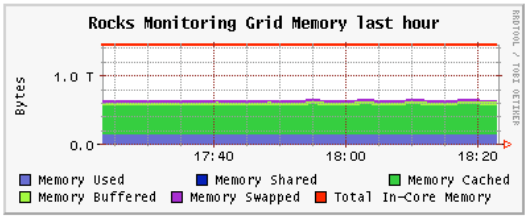
ganglia.sourceforge.net - UCB

Rocks Monitoring Grid (5 sources) (tree view)

CPUs Total: 1712
Hosts up: 689
Hosts down: 37

Avg Load (15, 5, 1m):
29%, 29%, 28%
Localtime:
2007-03-15 18:23

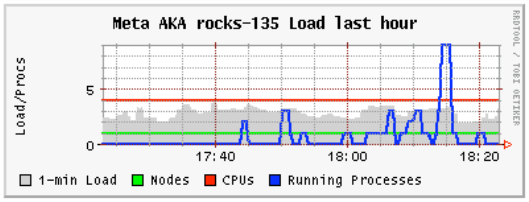


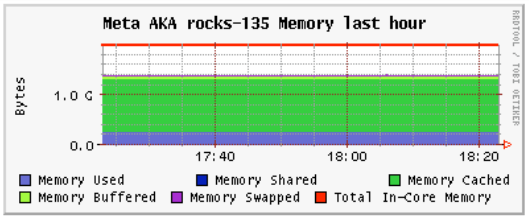


Meta AKA rocks-135 (physical view)

CPUs Total: 4
Hosts up: 1
Hosts down: 0

Avg Load (15, 5, 1m):
74%, 67%, 80%
Localtime:
2007-03-15 18:22

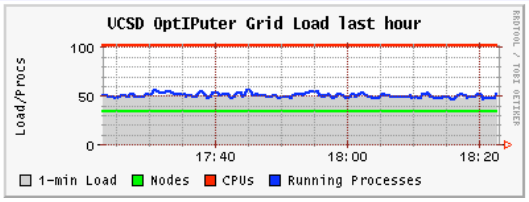


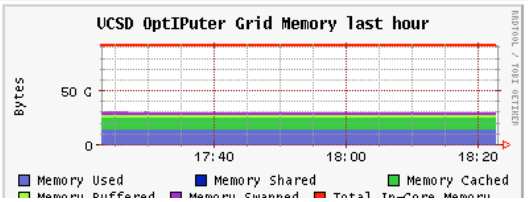


UCSD OptIPuter Grid (tree view)

CPUs Total: 102
Hosts up: 34
Hosts down: 35

Avg Load (15, 5, 1m):
48%, 48%, 49%
Localtime:
2007-03-15 18:22

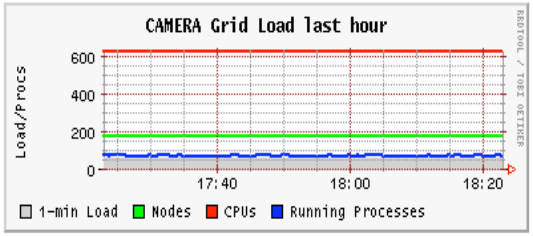


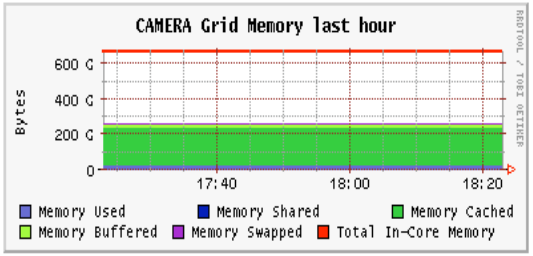


CAMERA Grid (tree view)

CPUs Total: 624
Hosts up: 172
Hosts down: 0

Avg Load (15, 5, 1m):
9%, 9%, 9%
Localtime:
2007-03-15 18:22





Monitoring / Management

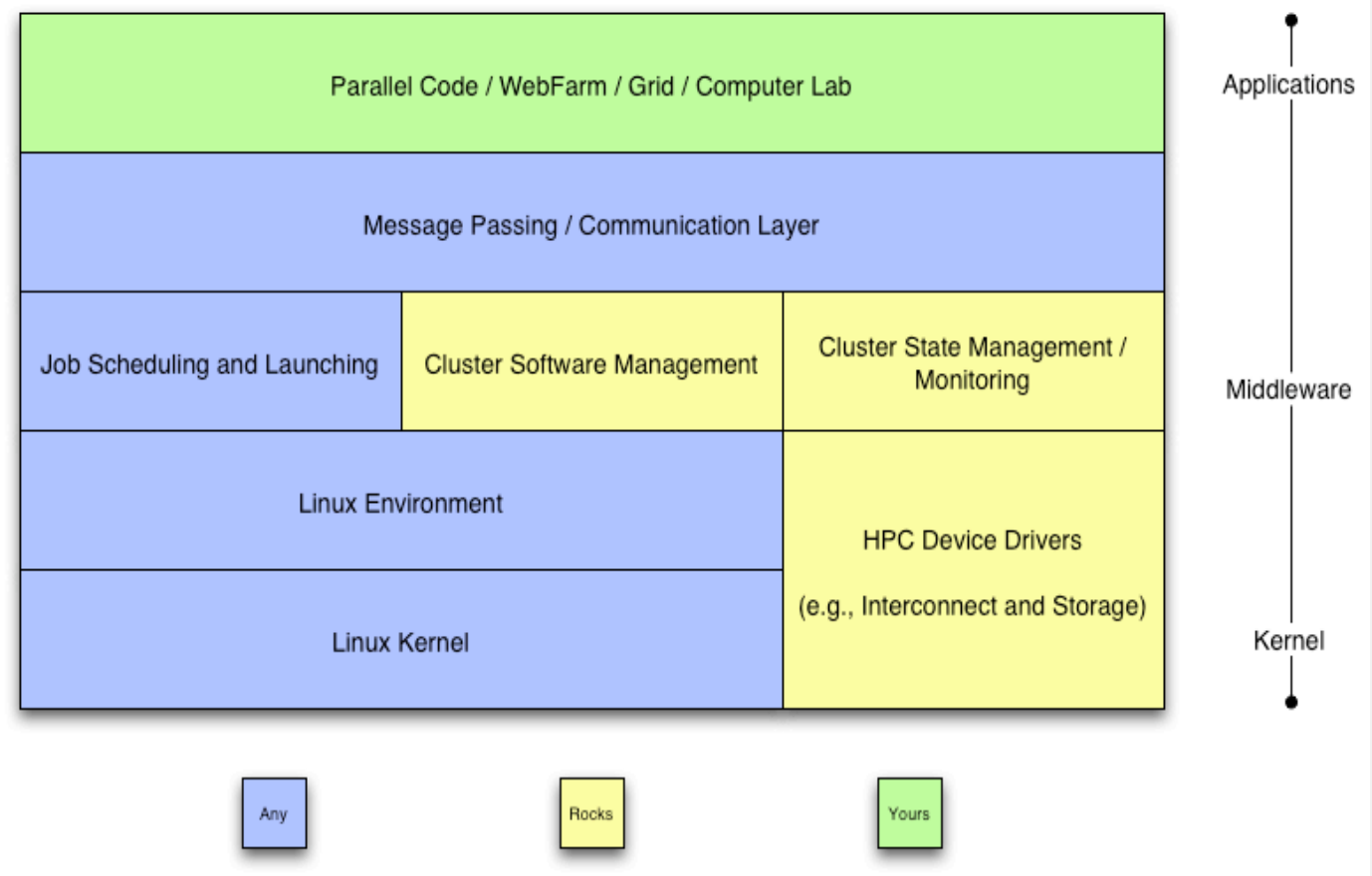
Visualization Cluster

Portal / Compute Cluster

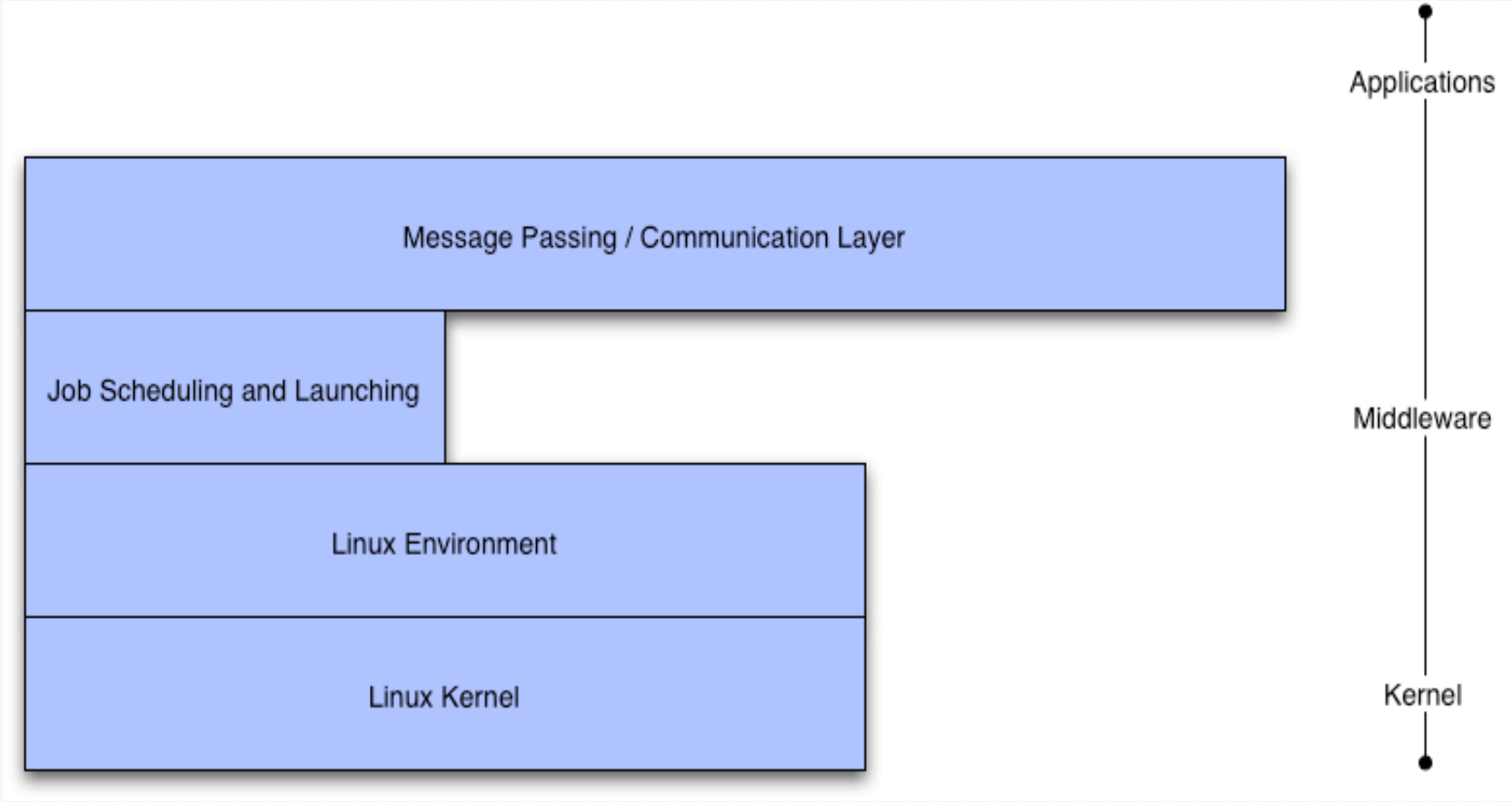
key point

Rocks builds more than just MPI machines

Basic Cluster Software Stack



Common to Any Cluster



Red Hat

- Enterprise Linux 4.0
  - Recompiled from public SRPMS, including errata updates (source code)
  - No license fee required, redistribution is also fine
  - Recompiled for all CPU types (x86, Opteron, Itanium)
  - *Rocks 5.0 will be based on RHEL 5.0 (Centos, or RHEL)*
- Standard Red Hat Linux kernel
  - No Rocks added kernel patches
- No support for other distributions
  - Red Hat is the market leader for Linux
    - In the US
    - And becoming so in Europe
  - Trivial to support any Anaconda-based system
  - Others would be harder, and require vendor support (SuSe ~ 12 months work)
- Excellent support for automated installation
  - Scriptable installation (Kickstart)
  - Very good hardware detection

Dell Invests in Red Hat

Michael Dell puts $99.5M in Red Hat

Billionaire chairman of No. 1 PC maker places big bet on Microsoft competitor.

May 10, 2005: 1:41 PM EDT

NEW YORK (CNN/Money) - Red Hat is getting a $99.5 million boost from Michael S. Dell, billionaire founder and chairman of Dell Inc., according a regulatory filing.

Through his private investment firm, MSD, Dell bought the largest share of $600 million in debentures offered by the software developer in January 2004, a Securities Exchange Commission filing showed.

Red Hat's main product, the Linux operating system for PCs, is a direct competitor to Microsoft's Windows. The Raleigh, N.C.-based company also provides support services for "open source" technology, which is software developed by communities of programmers for free use.

Dell (Research) is the nation's largest PC maker.

Debentures are similar to bonds in that the issuer promises a fixed return for a stated period of time on the investment.

In the case of a public company, a debenture can also be converted into shares or equity. ■

COURTESY: DELL COMPUTER

Michael Dell, billionaire chairman of Dell Inc., has given Red Hat a $99.5M injection.

Tech

SEE MORE STORIES

advertiser links — what's this?

Accounting Research Manager
Find insightful interpretations on GAAP and Securities and Exchange Commission...
www.accountingresearchmanager.c

Securities Exchange Commission

Batch Systems

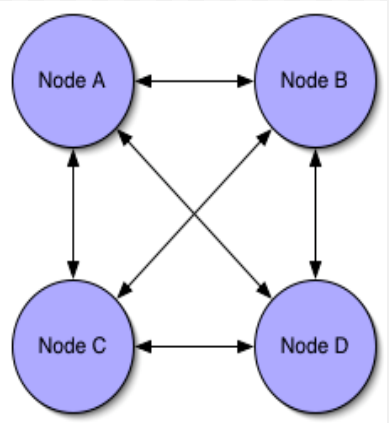
- Portable Batch System and Maui
  - Long time standard for HPC queuing systems
  - Maui provides backfilling for high throughput
  - PBS/Maui system can be fragile and unstable
  - Multiple code bases:
    - PBS
    - OpenPBS
    - PBSPro
    - Scalable PBS
- Sun Grid Engine
  - Rapidly becoming the new standard
  - Integrated into Rocks by Scalable Systems
  - Now the default scheduler for Rocks
  - Robust and dynamic

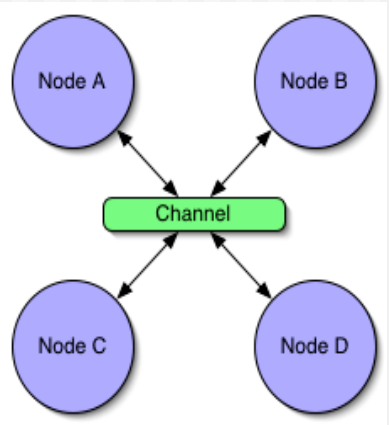
Communication Layer

- None
  - “Embarrassingly Parallel”
- Sockets
  - Client-Server model
  - Point-to-point communication
- MPI - Message Passing Interface
  - Message Passing
  - Static model of participants
- PVM - Parallel Virtual Machines
  - Message Passing
  - For Heterogeneous architectures
  - Resource Control and Fault Tolerance

Sockets are low level

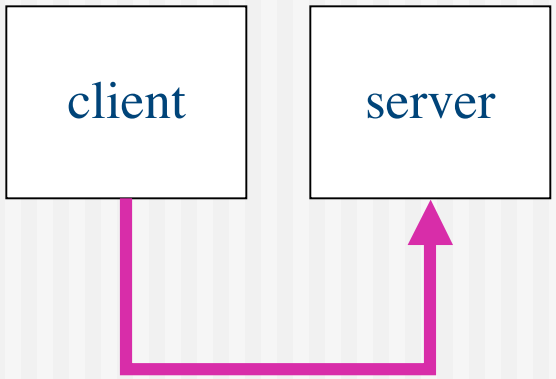
- Sockets
  - Point-to-Point
  - N machines = (n^2 - n)/2 connections
  - 1, 3, 6, 10, 15, …
- MPI/PVM
  - Shared virtual channel
  - Implementation could be sockets
  - Easier to program





Sockets

- Open an endpoint
- Specify IP address and port
- Send / receive messages
  - If TCP, only point-to-point messages
  - If UDP, option of point-to-point or multicast (broadcast)
- Shutdown connection



High-level TCP Example

```
/*
 * SERVER CODE
 */

fd = socket();
.
.
saddr.s_addr    = INADDR_ANY;
saddr.port      = 1234;
bind(fd, &saddr);
listen(fd);
accept(fd);
.
.
read(fd, buffer, size);
.
.
close(fd);
```

```
/*
 * CLIENT CODE
 */

fd = socket();
.
.
saddr.s_addr    = gethostbyname("c0-0");
saddr.port      = 1234;
.
.
write(fd, buffer, size);
.
.
close(fd);
```

Challenges with Sockets

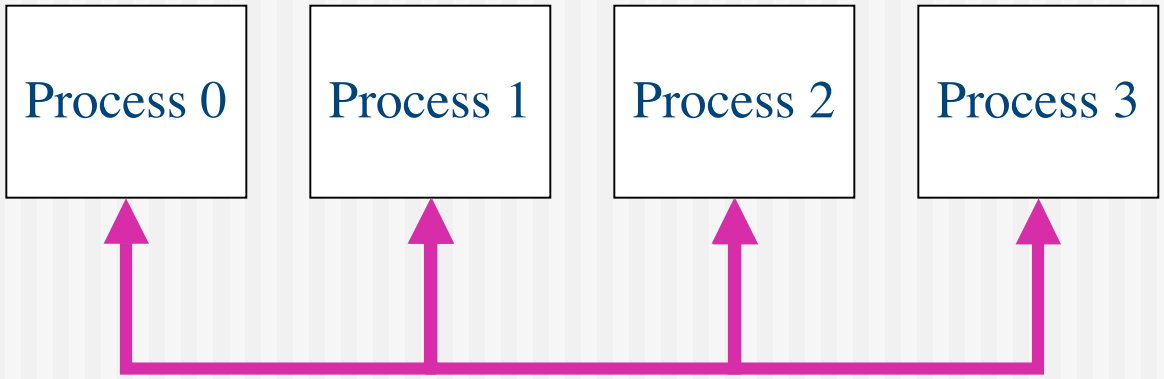
- TCP
  - Reliable, but byte oriented
  - Need to write code to send and receive *packets* (at the application level)
- UDP
  - Unreliable
  - Need to write code to reliably send packets

MPI

- Message Passing Interface
- *De facto* standard for message passing
  - Runs over many CPU architectures and many communication substrates
- There are (and were) lots of good messaging libraries
  - But, MPI is the most pervasive
  - Developed a practical, portable, efficient and flexible standard
  - In development since 1992

MPI

- Explicitly move data like sockets, but virtualizes the endpoints
  - Remote endpoints addressed by integer 0, 1, …, n
- Primitives to support point-to-point and broadcast



High-level MPI Example

```
MPI_Init();
.
.
MPI_Comm_rank(&my_mpi_id);
.
.
Remote_mpi_id = 1
MPI_Send(send_buffer, buf_size, remote_mpi_id)
.
.
MPI_Recv(recv_buffer, buf_size, remote_mpi_id)
.
.
MPI_Finalize()
```

Challenges with MPI

- If a node fails, no easy way to reconfigure and route around the problem
  - Basically, your program stops
- Hard to manage deployment
  - network X compiler = mpi binaries
  - Result is several versions of MPI / cluster

Compile

- MPICH with GNU Compilers and Ethernet

Compiler	Path
C:	/opt/mpich/ethernet/gcc/bin/mpicc
C++:	/opt/mpich/ethernet/gcc/bin/mpiCC
F77:	/opt/mpich/ethernet/gcc/bin/mpif77

- MPICH with GNU Compilers and Myrinet

Compiler	Path
C:	/opt/mpich/myrinet/gcc/bin/mpicc
C++:	/opt/mpich/myrinet/gcc/bin/mpiCC
F77:	/opt/mpich/myrinet/g77/bin/mpif77

Compile

- MPICH with Intel Compilers and Ethernet

Compiler	Path
C:	/opt/mpich/ethernet/ecc/mpicc
C++:	/opt/mpich/ethernet/ecc/mpiCC
F77:	/opt/mpich/ethernet/ecc/mpif77
F90:	/opt/mpich/ethernet/ecc/mpif90

- MPICH with Intel Compilers and Myrinet

Compiler	Path
C:	/opt/mpich/myrinet/ecc/mpicc
C++:	/opt/mpich/myrinet/ecc/mpiCC
F77:	/opt/mpich/myrinet/efc/mpif77
F90:	/opt/mpich/myrinet/efc/mpif90

PVM

- Parallel Virtual Machines v3.4.3
  - Message passing interface for heterogeneous architectures
    - Supports over 60 variants of UNIX
    - Supports Windows NT
  - Resource control and meta computing
  - Fault tolerance
  - http://www.csm.ornl.gov/pvm/

NFS

- User account are served over NFS
  - Works for small clusters (<= 128 nodes)
  - Will not work for large clusters (>1024 nodes)
  - NAS is better than Linux
    - Rocks uses the Frontend machine to server NFS
    - We have deployed NAS on several clusters
- Applications are not served over NFS
  - /usr/local/ does not exist
  - All software is installed locally from RPM

SNMP

- Enabled on all compute nodes
- Great for point-to-point use
  - Good for high detail on a single end-point
  - Does not scale to full cluster wide use
- Supports Linux MIB
  - Uptime, Load, Network statistics
  - Install Software
  - Running Processes

Syslog

- Native UNIX system event logger
  - Logs events to local dist
    - /var/log/message
    - Rotates logs daily, eventually historic data is lost
  - Forwards all message to the frontend
- Scalable
  - Can add additional loghosts
  - Can throttle verbosity of loggers
- Uses
  - Predicting hardware and software failures
  - Post Mortem on crashed nodes
  - Debugging System startup

eKV

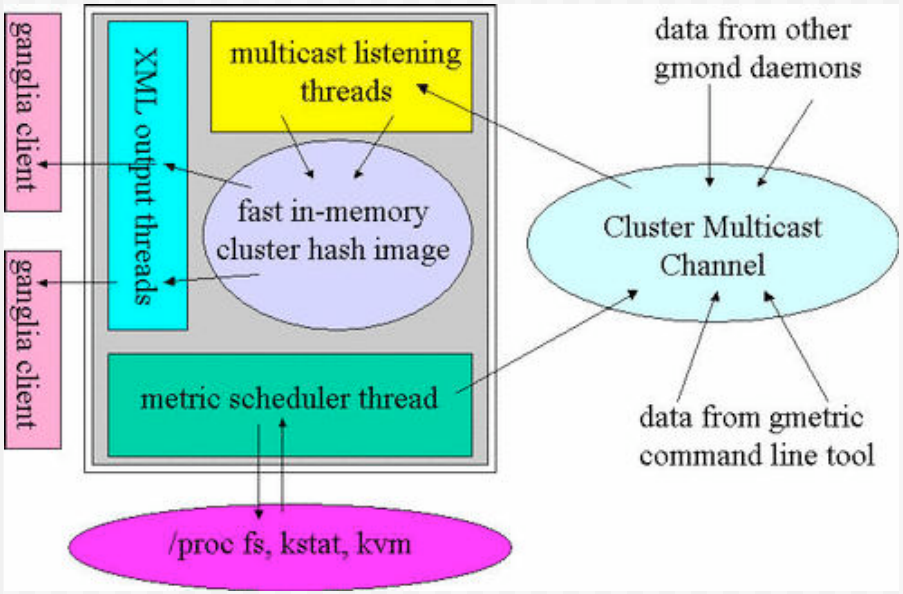
- Remotely Interact with Installation
  - Initial kickstart
  - Re-Installation
- Shoot-node
  - Reinstall OS and brings up eKV
- eKV
  - Ssh to node while it is installing
  - See the console output over Ethernet
- Newer versions of Rocks (4.0+) use VNC
  - Graphical
  - Works on headless machines

Optional Drivers

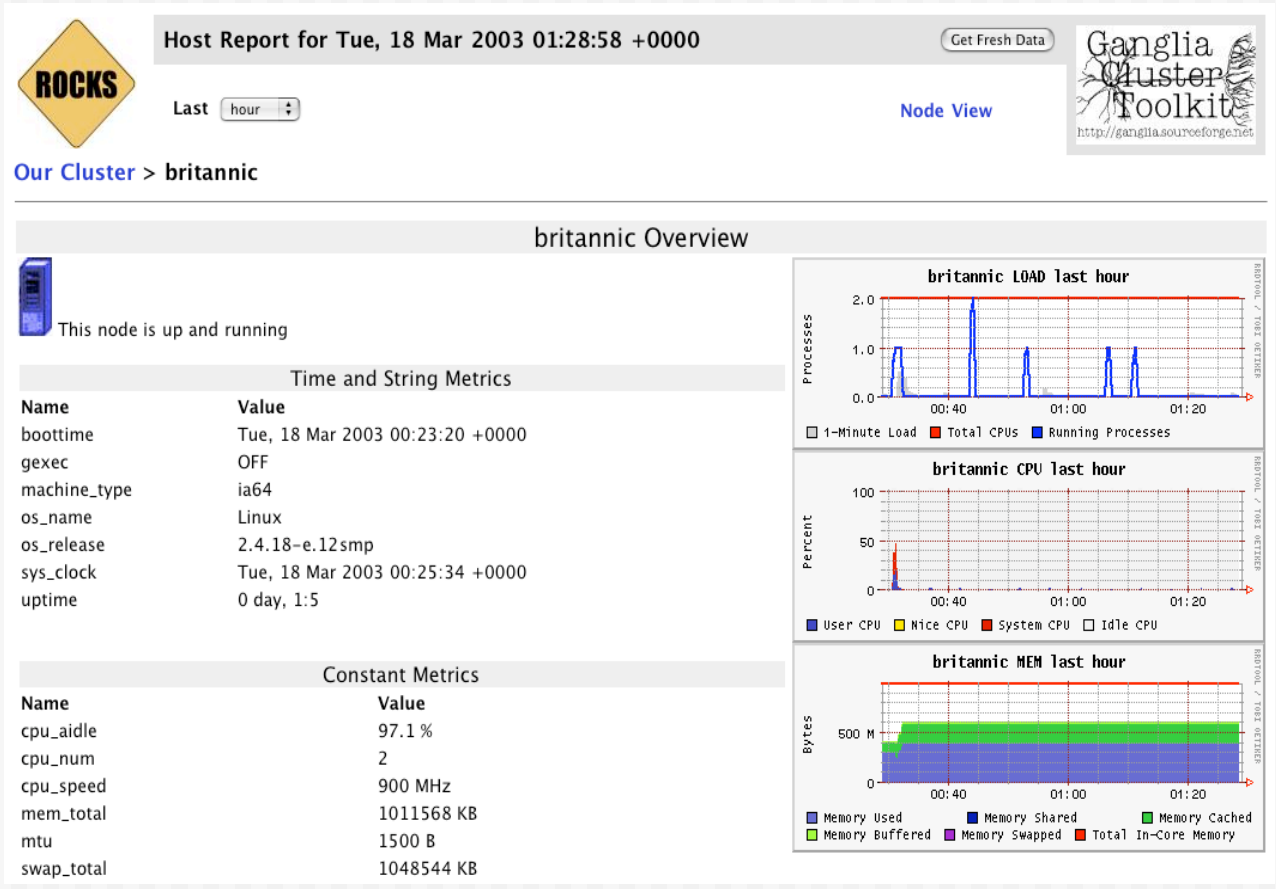
- PVFS
  - Parallel Virtual File System
  - Kernel module built for all nodes
  - User must decide to enable
- Myrinet
  - High Speed and Low Latency Interconnect
  - GM/MPI for user Applications
  - Kernel module built for all nodes with Myrinet cards
- Video
  - nVidia (from Viz Roll)
- Add your own
  - Cluster Gigabit Ethernet driver
  - Infiniband driver
- Kernel Modules are dynamically built
- No need to manage binary Kernel Modules
- Burn CPU time, not human time

Ganglia

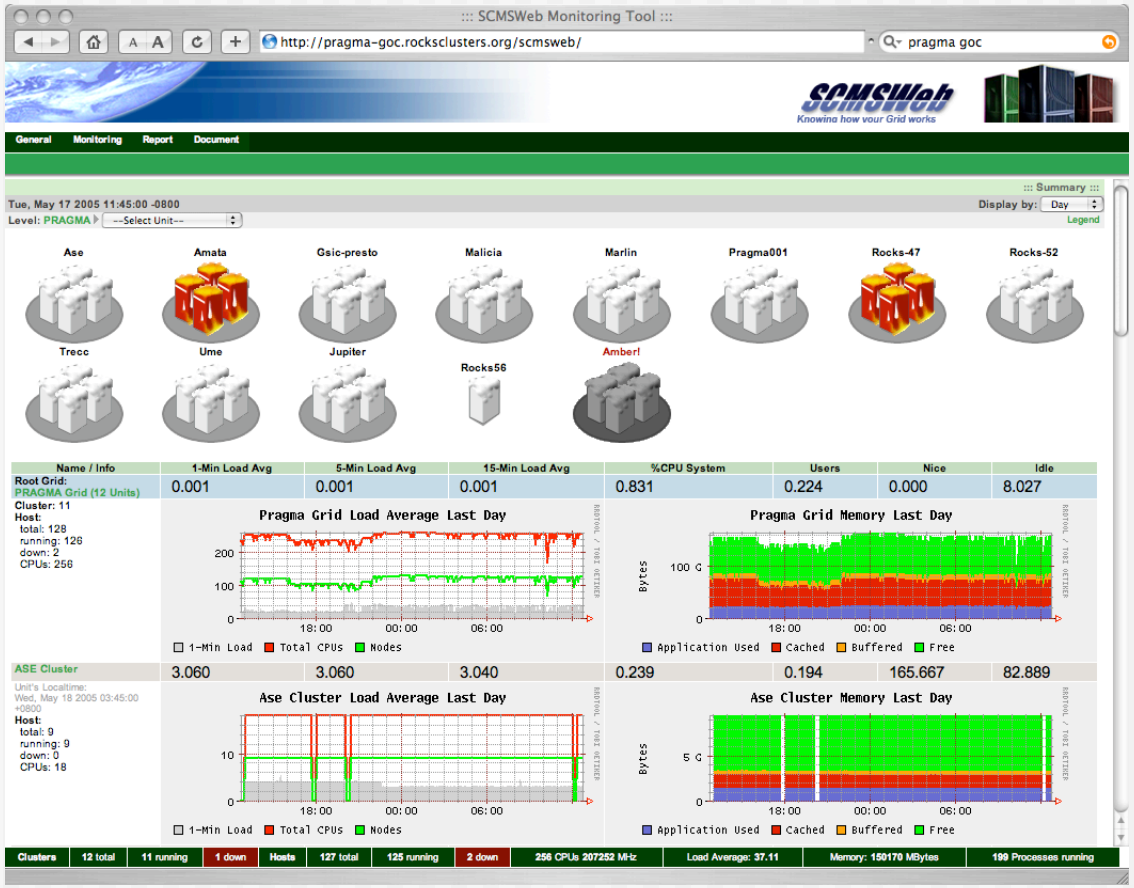
- Scalable cluster monitoring system
  - Based on ip multi-cast
  - Matt Massie, et al from UCB
  - http://ganglia.sourceforge.net
- Gmon daemon on every node
  - Multicasts system state
  - Listens to other daemons
  - All data is represented in XML
- Ganglia command line
  - Python code to parse XML to English
- Gmetric
  - Extends Ganglia
  - Command line to multicast single metrics



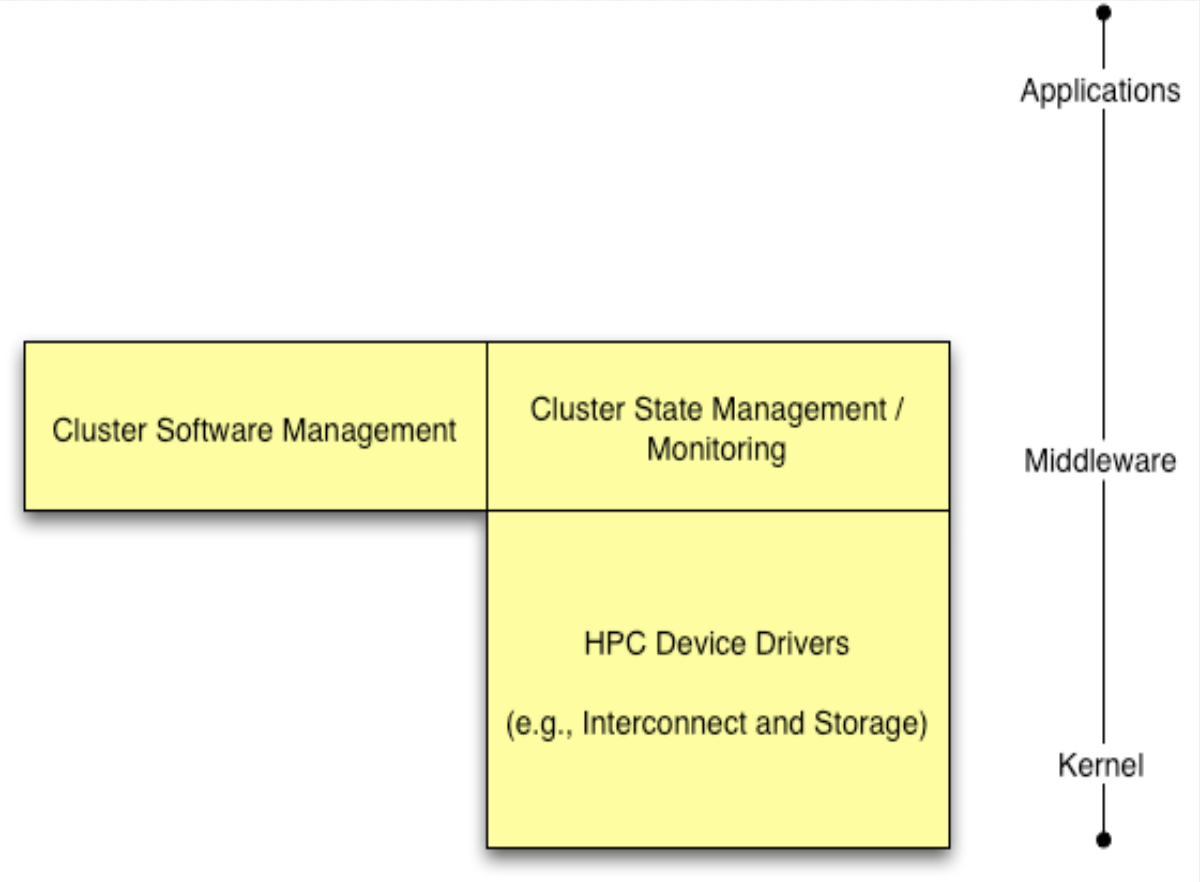
Ganglia Screenshot



SCMSWeb Screenshot



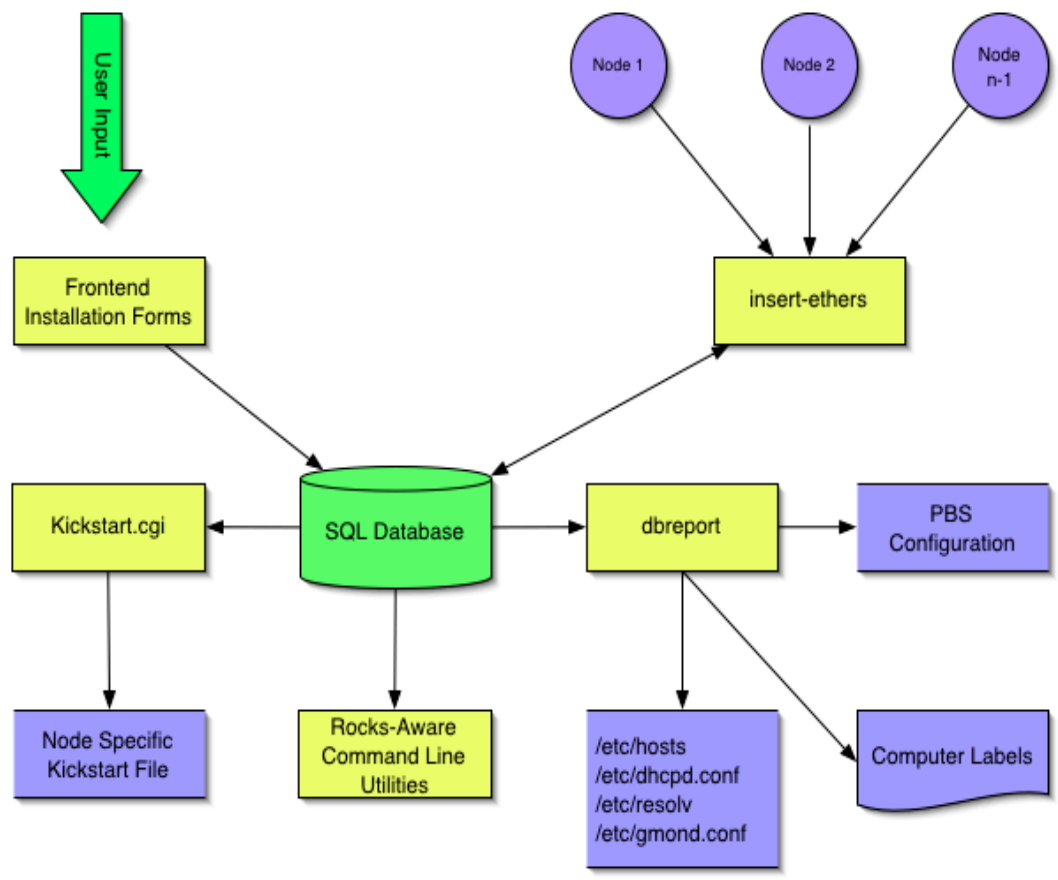
Rocks Cluster Software



Cluster State Management

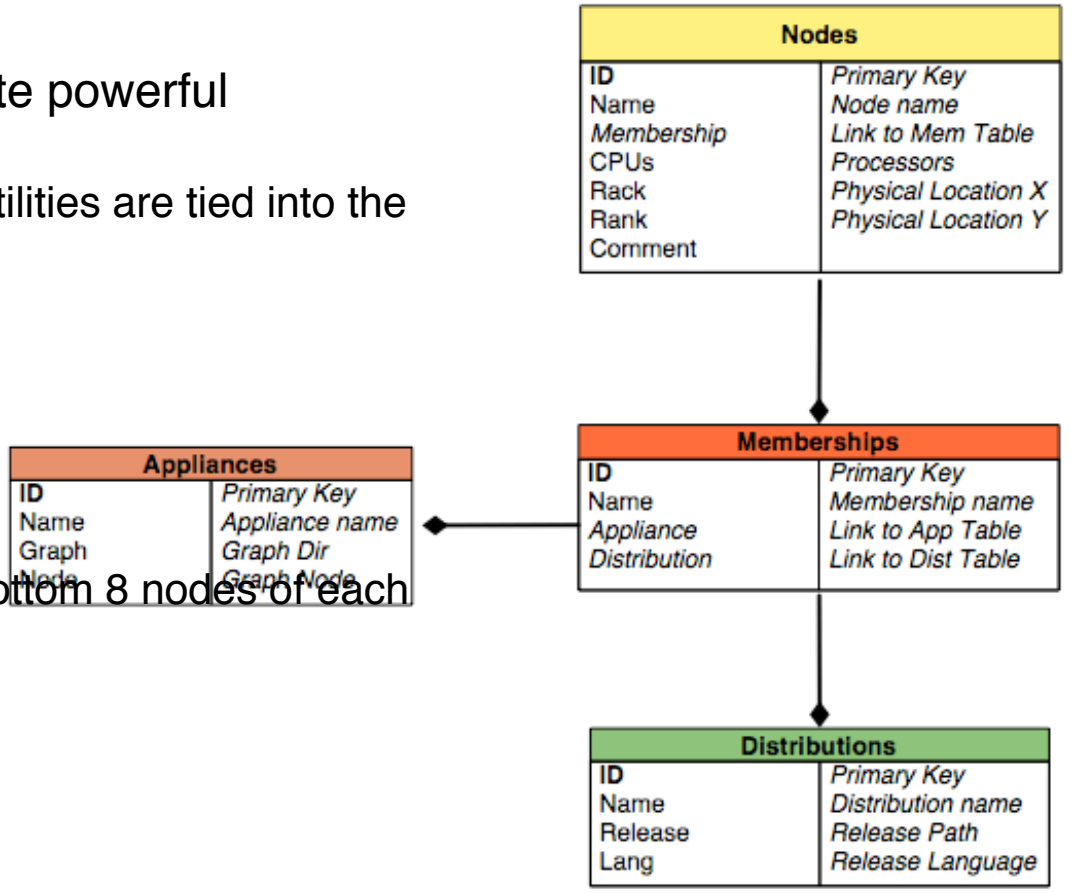
- Static Information
  - Node addresses
  - Node types
  - Site-specific configuration
- Dynamic Information
  - CPU utilization
  - Disk utilization
  - Which nodes are online

Cluster Database



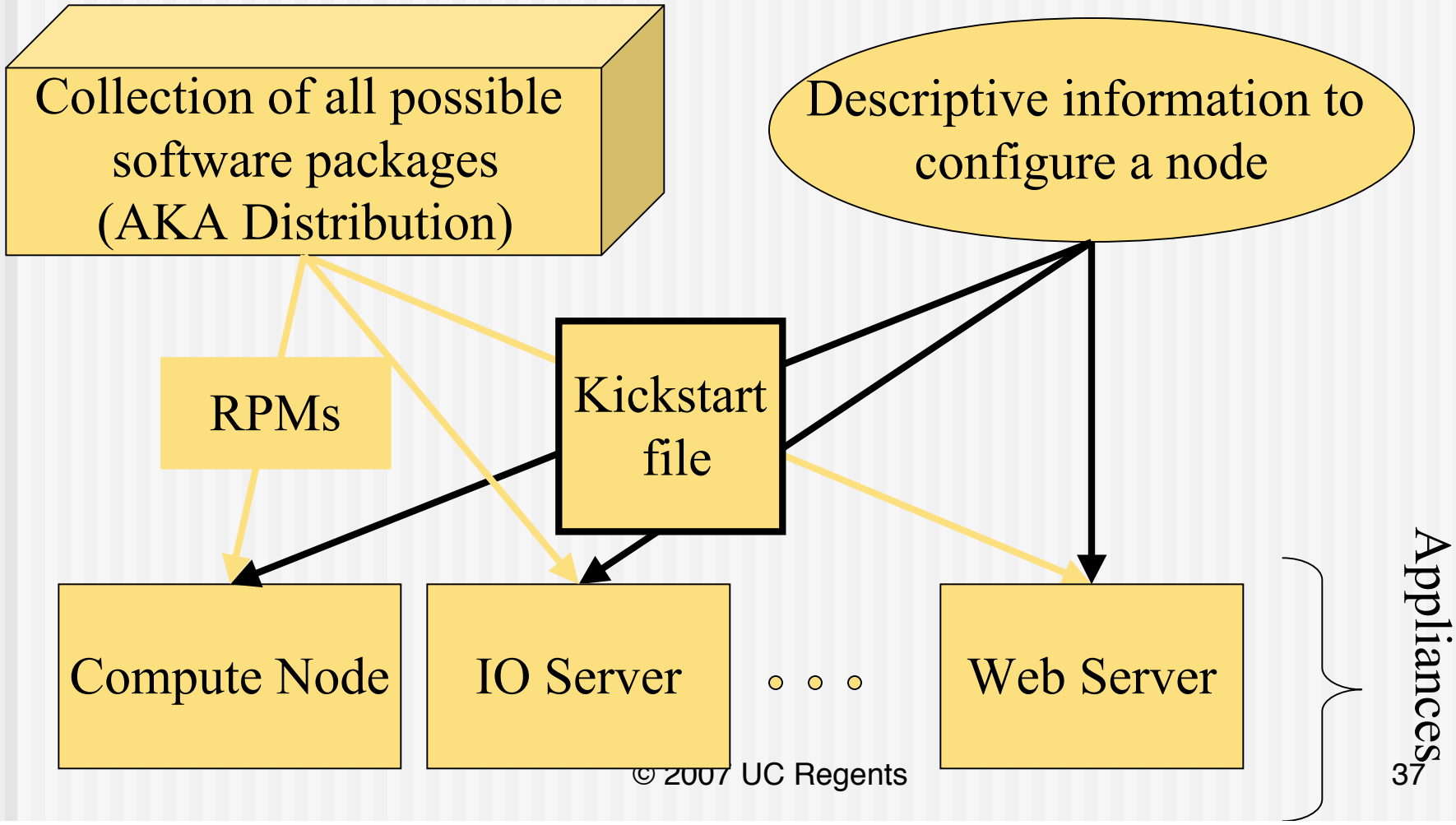
Node Info Stored In A MySQL Database

- If you know SQL, you can execute powerful commands
  - Rocks-supplied command line utilities are tied into the database
  - E.g., get the hostname for the bottom 8 nodes of each cabinet:

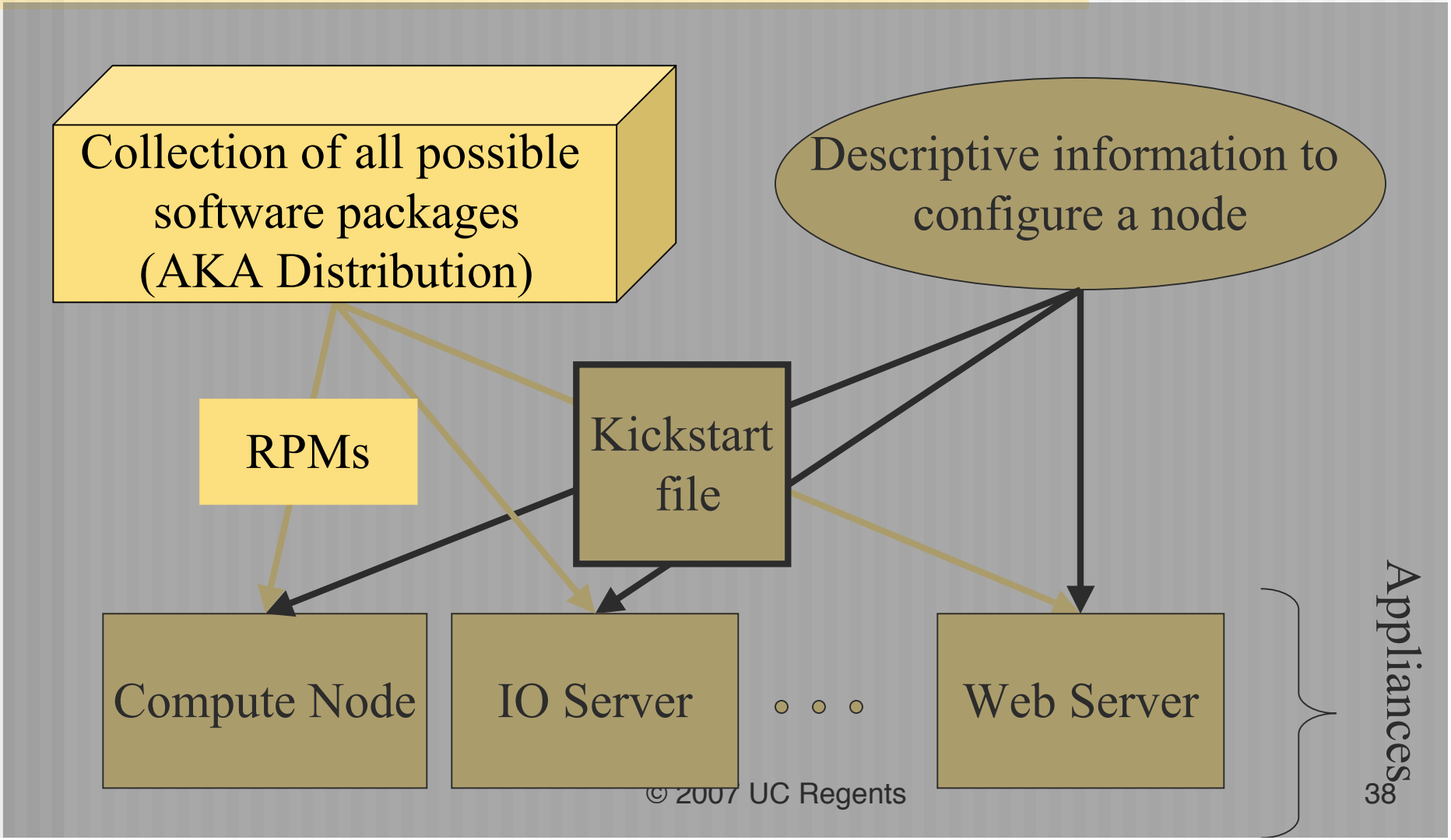



```
# cluster-fork --query="select name from nodes where rank<8" hostname
```

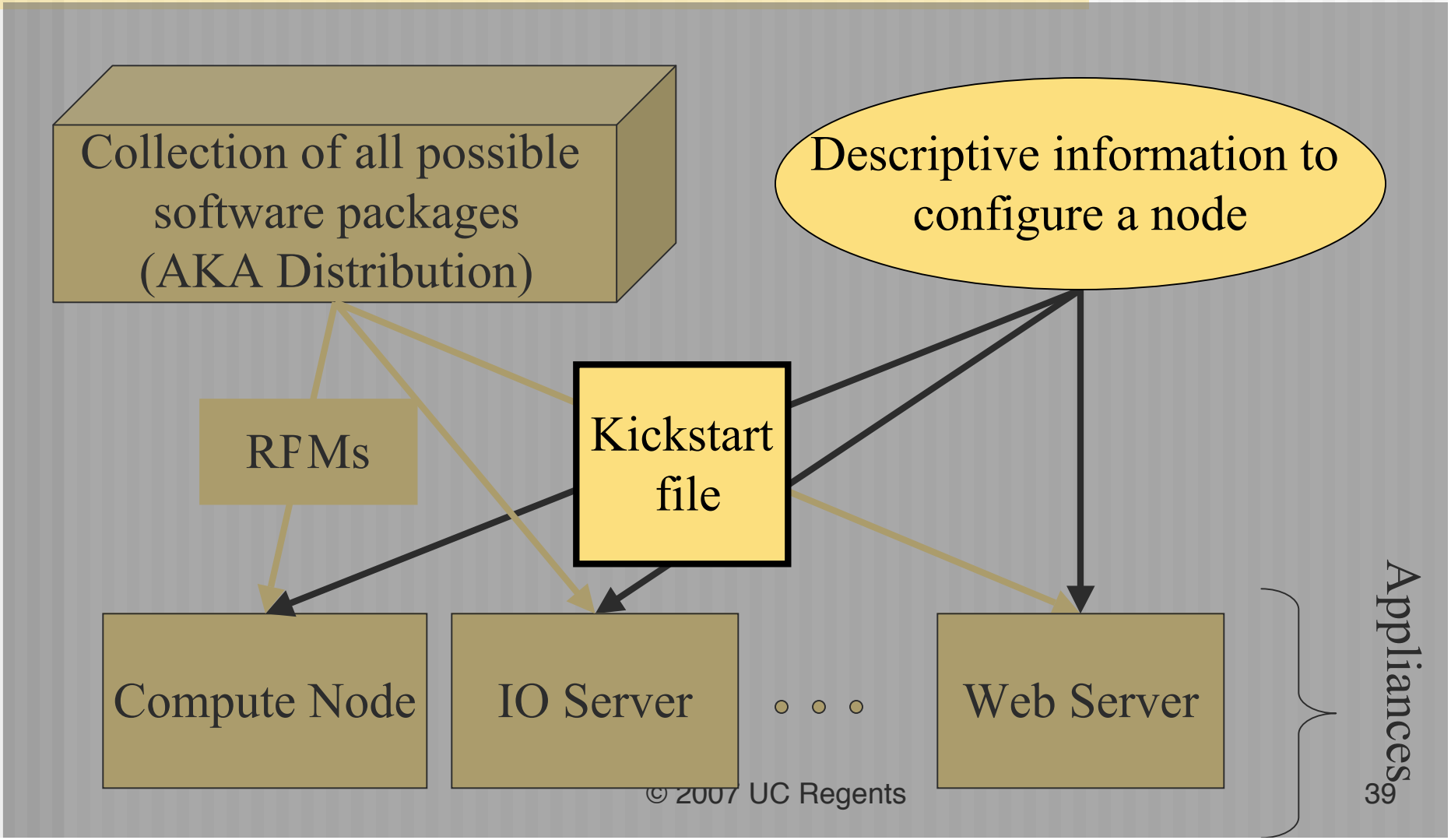
Software Installation



Software Repository



Installation Instructions



Cluster Software Management

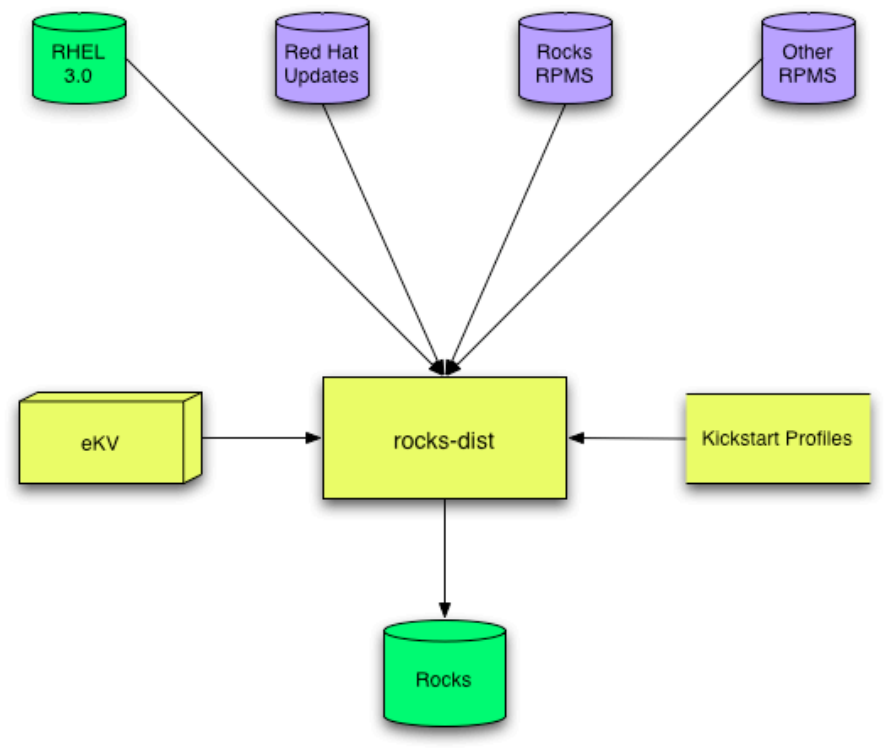
Software Packages

- RPMs
  - Standard Red Hat (desktop) packaged software
  - Or your own addons
- Rocks-dist
  - Manages the RPM repository
  - This is the distribution

Software Configuration

- Tuning RPMs
  - For clusters
  - For your site
  - Other customization
- XML Kickstart
  - Programmatic System Building
  - Scalable

Building a Rocks Distribution



- Start with Red Hat
- Add updates, Rocks (and optional other) software
- Add Kickstart profiles
- Modify Red Hat installation boot image
- Resulting in a Red Hat compatible Rocks distribution

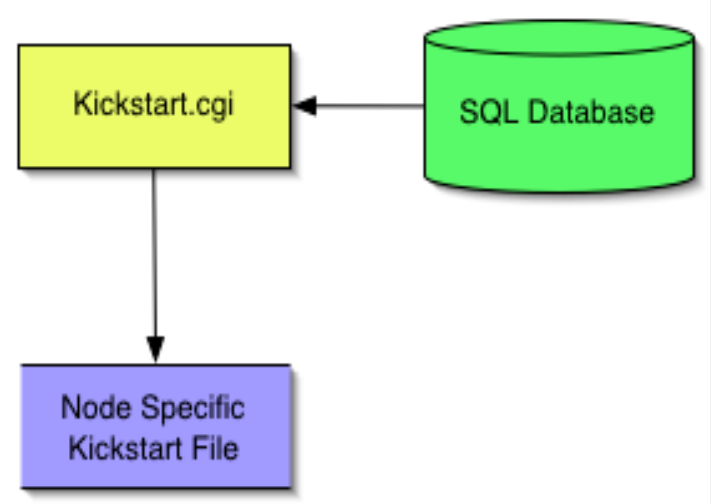
Nodes Main: Partitioning

- `<main>`
  - `<part> / --size 8000 --ondisk hda </part>`
  - `<part> swap --size 1000 --ondisk hda </part>`
  - `<part> /mydata --size 1 --grow --ondisk hda </part>`
- `</main>`

```
part / --size 8000 --ondisk hda
part swap --size 1000 --ondisk hda
part /mydata --size 1 --grow --ondisk hda
```

Kickstart

- Red Hat's Kickstart
  - Monolithic flat ASCII file
  - No macro language
  - Requires forking based on site information and node type.
- Rocks XML Kickstart
  - Decompose a kickstart file into nodes and a graph
    - Graph specifies OO framework
    - Each node specifies a service and its configuration
  - Macros and SQL for site configuration
  - Driven from web cgi script



Kickstart File Sections

- Main
  - Disk partitioning
  - Root password
  - RPM repository URL
  - …
- Packages
  - List of RPMs (w/o version numbers)
  - The repository determines the RPM versions
  - The kickstart file determines the set of RPMs
- Pre
  - Shell scripts run before RPMs are installed
  - Rarely used (Rocks uses it to enhance kickstart)
- Post
  - Shell scripts to cleanup RPM installation
  - Fixes bugs in packages
  - Adds local information

What is a Kickstart File?

- Setup & Packages (20%)

```
cdrom
zerombr yes
bootloader --location mbr --useLilo
skipx
auth --useshadow --enablemd5
clearpart --all
part /boot --size 128
part swap --size 128
part / --size 4096
part /export --size 1 --grow
lang en_US
langsupport --default en_US
keyboard us
mouse genericps/2
timezone --utc GMT
rootpw --iscrypted nrDq4Vb42jjQ.
text
install
reboot

%packages
@Base
@Emacs
@GNOME
```

- Post Configuration (80%)

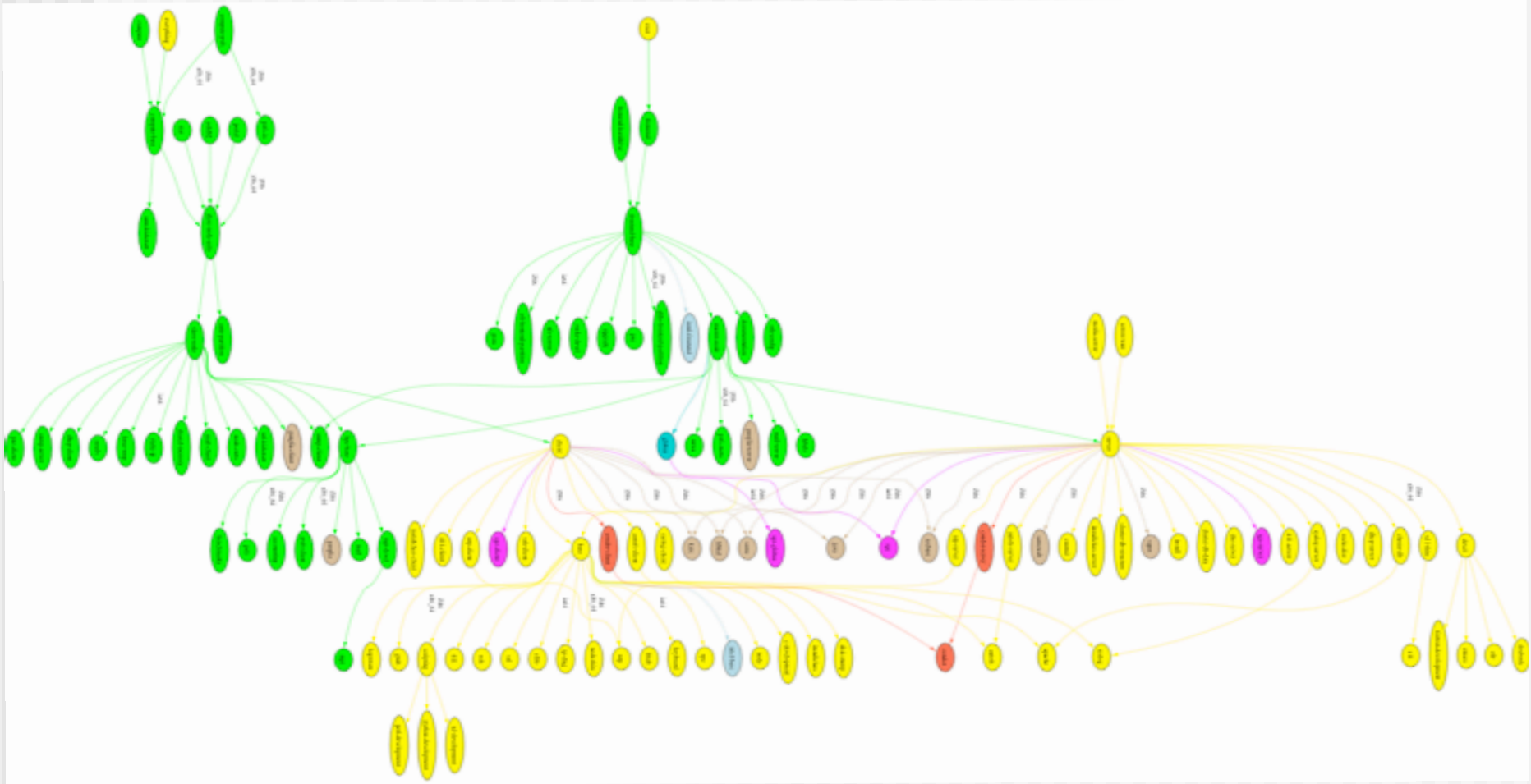
```
%post

cat > /etc/nsswitch.conf << 'EOF'
passwd:     files
shadow:     files
group:      files
hosts:      files dns
bootparams: files
ethers:     files
EOF

cat > /etc/ntp.conf << 'EOF'
server ntp.ucsd.edu
server       127.127.1.1
fudge        127.127.1.1 stratum 10
authenticate no
driftfile /etc/ntp/drift
EOF

/bin/mkdir -p /etc/ntp
cat > /etc/ntp/step-tickers << 'EOF'
ntp.ucsd.edu
EOF

/usr/sbin/ntpdate ntp.ucsd.edu
/sbin/hwclock --systohc
```

Issues

- High level description of software installation
  - List of packages (RPMs)
  - System configuration (network, disk, accounts, …)
  - Post installation scripts
- *De facto* standard for Linux
- Single ASCII file
  - Simple, clean, and portable
  - Installer can handle simple hardware differences
- Monolithic
  - No macro language
  - Differences require forking (and code replication)
  - Cut-and-Paste is not a code re-use model

XML Kickstart

It looks something like this

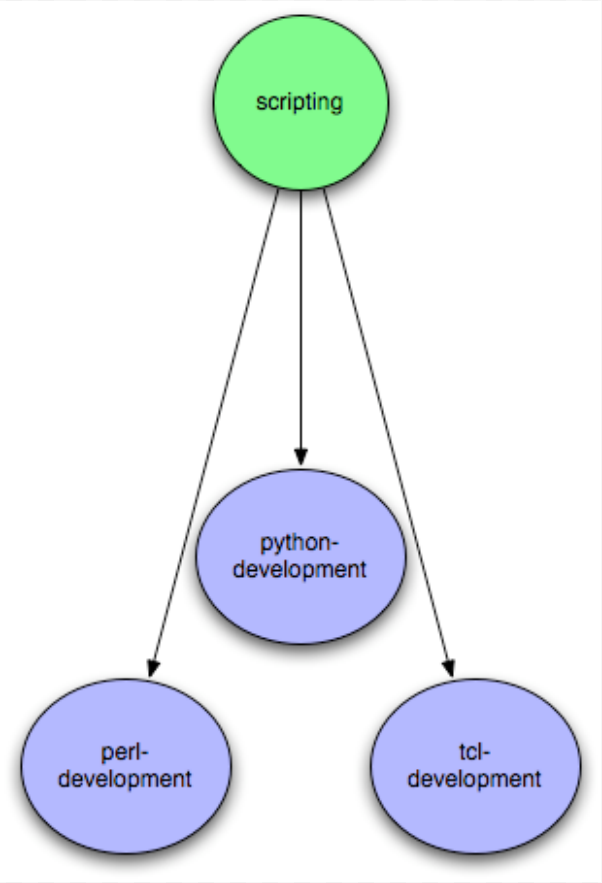


Implementation

- Nodes
  - Single purpose modules
  - Kickstart file snippets (XML tags map to kickstart commands)
  - Approximately 200 node files in Rocks
- Graph
  - Defines interconnections for nodes
  - Think OOP or dependencies (class, #include)
  - A single default graph file in Rocks
- Macros
  - SQL Database holds site and node specific state
  - Node files may contain <var name="state"/> tags

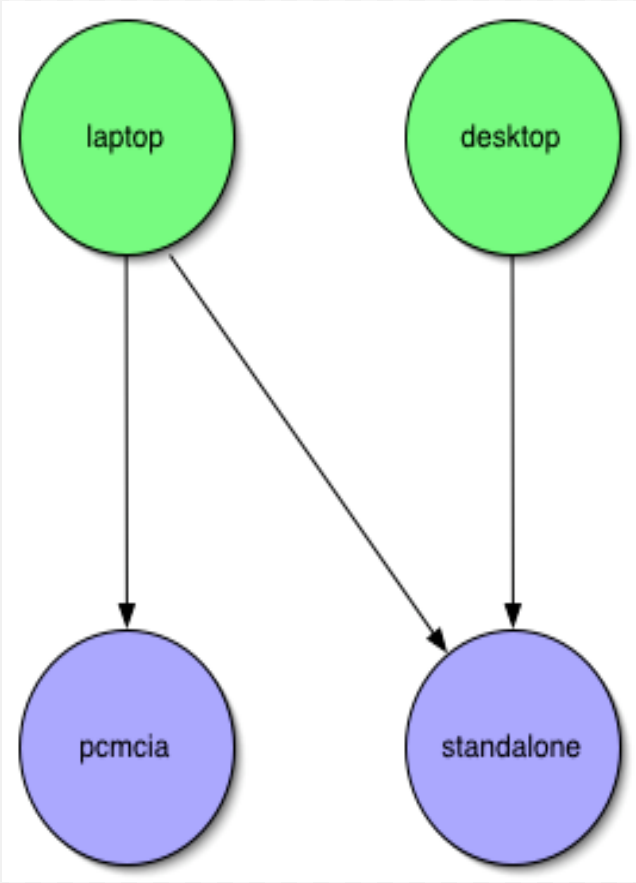
Composition

- Aggregate Functionality
- Scripting
  - IsA perl-development
  - IsA python-development
  - IsA tcl-development



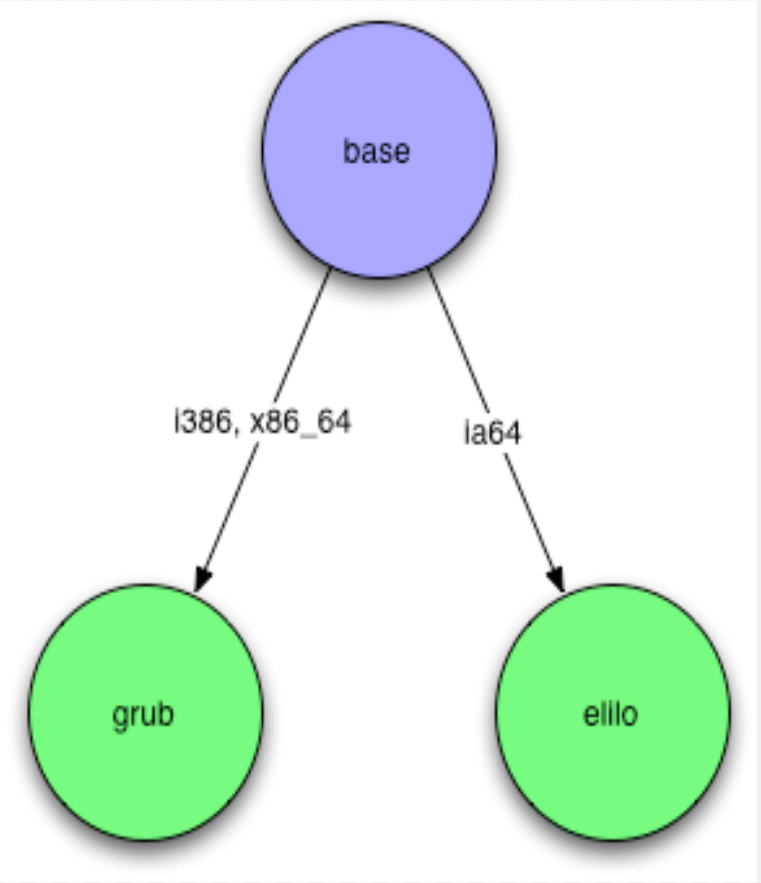
Appliances

- Laptop / Desktop
  - Appliances
  - Final classes
  - Node types
- Desktop IsA
  - standalone
- Laptop IsA
  - standalone
  - pcmcia
- Specify only the differences

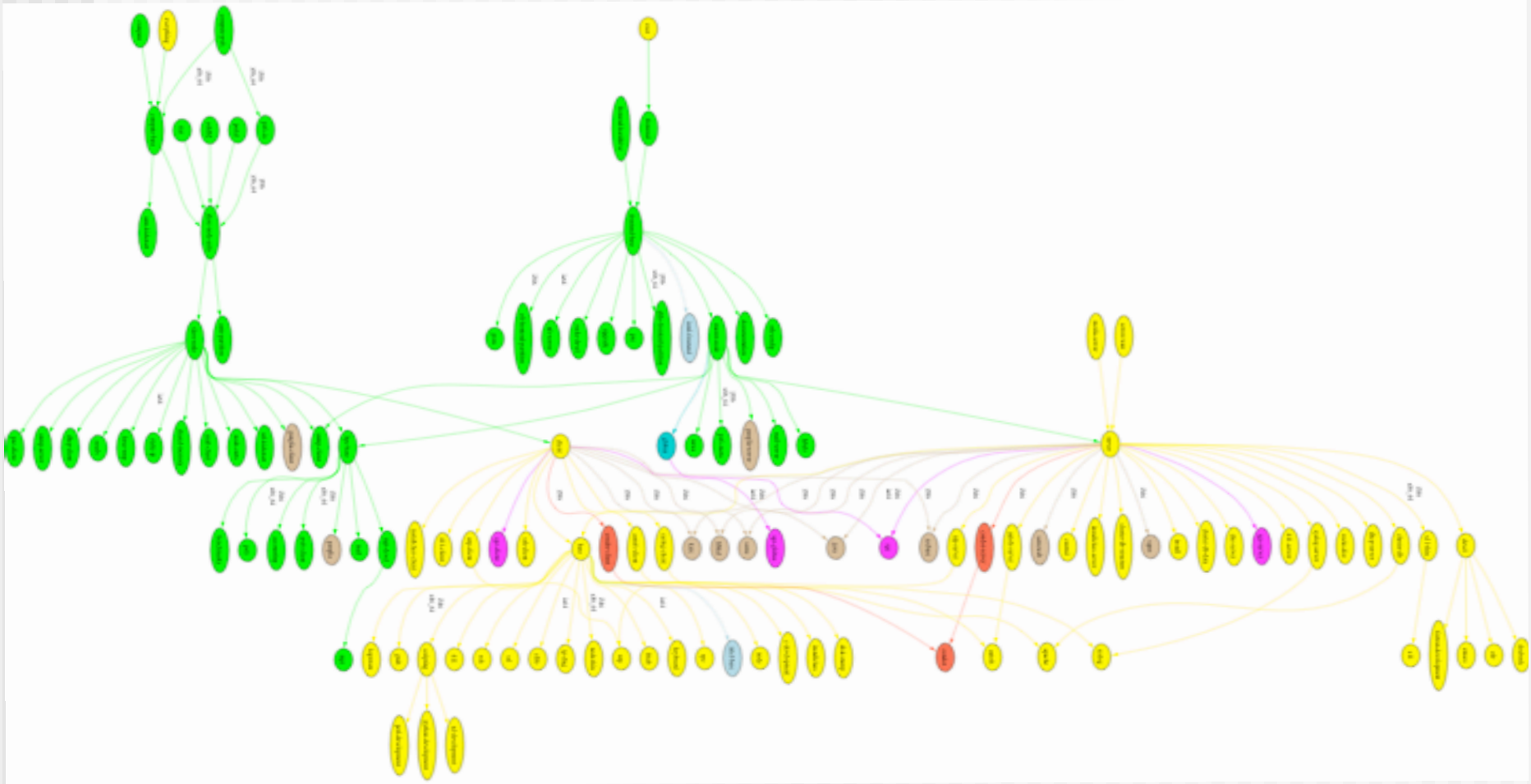


Architecture Differences

- Conditional inheritance
- Annotate edges with target architectures
- if i386
  - Base IsA grub
- if ia64
  - Base IsA elilo
- One Graph, Many CPU Architectures
  - Heterogeneity becomes easy
  - Not true for SSI or Imaging



Putting in all together



key point

The XML Graph is the DNA of the cluster

Sample Node File

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE kickstart SYSTEM "@KICKSTART_DTD@" [<!ENTITY ssh "openssh">]>
<kickstart>
            <description>
            Enable SSH
            </description>

            <package>&ssh;</package>
            <package>&ssh;-clients</package>
            <package>&ssh;-server</package>
            <package>&ssh;-askpass</package>
<post>

cat &gt; /etc/ssh/ssh_config &lt;&lt; 'EOF' <!-- default client setup -->
Host *
        ForwardX11 yes
        ForwardAgent yes
EOF

chmod o+rx /root
mkdir /root/.ssh
chmod o+rx /root/.ssh

</post>
</kickstart>>
```

Sample Graph File

```
<?xml version="1.0" standalone="no"?>

<graph>
            <description>
            Default Graph for NPACI Rocks.
            </description>

            <edge from="base" to="scripting"/>
            <edge from="base" to="ssh"/>
            <edge from="base" to="ssl"/>
            <edge from="base" to=”grub"  arch="i386"/>
            <edge from="base" to="elilo" arch="ia64"/>
                        …
            <edge from="node" to="base"/>
            <edge from="node" to="accounting"/>
            <edge from="slave-node" to="node"/>
            <edge from="slave-node" to="nis-client"/>
            <edge from="slave-node" to="autofs-client"/>
            <edge from="slave-node" to="dhcp-client"/>
            <edge from="slave-node" to="snmp-server"/>
            <edge from="slave-node" to="node-certs"/>
            <edge from="compute" to="slave-node"/>
            <edge from="compute" to="usher-server"/>
            <edge from="master-node" to="node"/>
            <edge from="master-node" to="x11"/>
            <edge from="master-node" to="usher-client"/>
</graph>
```

Cluster SQL Database

Nodes and Groups

ID	MAC	Name	Membership	Hardware	Rack	Rank	IP	C
1		frontend-0	1	0	0	0	10.1.1.1	
2	00:30:c1:d8:59:00	network-1-0	6	0	1	0	10.255.255.254	
3	00:01:e7:1a:be:00	network-0-0	6	0	0	0	10.255.255.253	
4	00:30:c1:d8:ac:80	network-3-0	6					
5	00:50:8b:a5:4d:b1	nfs-0-0	12					
6	00:50:8b:c5:c3:72	nfs-0-1	12					
7	00:50:8b:a5:57:ff	nfs-1-0	12					
8	00:50:8b:a5:4c:f4	nfs-1-1	12					
9	00:50:8b:e0:3a:a7	compute-0-0	2					
10	00:50:8b:e0:44:5e	compute-0-1	2					
11	00:50:8b:e0:40:95	compute-0-2	2					
12	00:50:8b:e0:40:93	compute-0-3	2					
13	00:50:8b:e0:42:df	compute-0-4	2					

Nodes Table

ID	Name	Appliance	Distribution	Compute
1	Frontend	1	1	no
2	Compute	2	1	yes
3	PVFS I/O Node	3	1	no
4	Compute with PVFS	4	1	yes
5	Laptop	5	1	no
6	Ethernet Switches	6		no
7	Myrinet Switches	6		no
8	Power Units	7		no
9	Remote Management	8		no
10	DTF Compute	9	1	yes
11	Web Portal	10	1	no
12	NFS Server	11	1	no

Memberships Table

Groups and Appliances

ID	Name	Appliance	Distribution	Compute
1	Frontend	1	1	no
2	Compute	2	1	yes
3	PVFS I/O Node	3		no
4	Compute with PVFS	4		yes
5	Laptop	5		no
6	Ethernet Switches	6		no
7	Myrinet Switches	6		no
8	Power Units	7		no
9	Remote Management	8		no
10	DTF Compute	9		yes
11	Web Portal	10		no
12	NFS Server	11	1	no

Memberships Table

ID	Name	ShortName	Graph	Node
1	frontend	f	default	frontend
2	compute	c	default	compute
3	pvfs	pv	default	pvfs-io
4	comp-pvfs	cp	default	compute-pvfs
5	laptop		default	laptop
6	network	n		
7	power	p		
8	manager			
9	dtf	d	default	dtf-compute
10	portal	pl	default	portal
11	nfs	n	default	nfs

Appliances Table

Simple key - value pairs

ID	Membership	Service	Component	Value
1	0	Kickstart	PublicNTPHost	ntp.ucsd.edu
2	0	Kickstart	ZeroMBR	yes
3	0	Kickstart	PrivateKickstartCGI	kickstart.cgi
4	0	Kickstart	PublicNetmask	255.255.255.0
5	0	Kickstart	PublicNetwork	192.31.21.0
6	0	Kickstart	PrivateNISMaster	frontend-0
7	0	Kickstart	PrivateHostname	frontend-0
8	0	Kickstart	PrivateIPForwarding	yes
9	0	Kickstart	PrivateGateway	10.1.1.1
10	0	Kickstart	PublicKickstartBasedir	install
11	0	Kickstart	Lang	en_US

- Used to configure DHCP and to customize appliance kickstart files

Nodes XML Tools: <var>

- Get Variables from Database
  - `<var name="Kickstart_PrivateGateway"/>`
  - `<var name="Node_Hostname"/>`

```
10.1.1.1
compute-0-0
```

  - Can grab any value from the *app_globals* database table

Nodes XML Tools: <eval>

- Do processing on the frontend:
  - **<eval shell="bash">**
- To insert a fortune in the kickstart file:

```
<eval shell="bash">
/usr/games/fortune
</eval>
```

"Been through Hell? Whaddya bring back for me?"
-- A. Brilliant

Nodes XML Tools: <eval>

- Inside <eval> variables are not accessed with <var>: use the environment instead.

```
<eval shell="sh">
echo "My NTP time server is
$Kickstart_PublicNTPHost"
echo "Got it?"
</eval>
```

My NTP time server is time.apple.com
Got it?

```
<eval shell="python">
import os
print "My NTP time server is",
 os.environ['Kickstart_PublicNTPHost']
print "Got it?"
</eval>
```

My NTP time server is time.apple.com
Got it?

Nodes XML Tools <include>

- Auto-quote XML characters in a file
  - **`<include file="foo.py"/>`**
- Quotes and includes file

  `sweetroll/include/foo.py`
- foo.py (native) → foo.py (quoted xml):

```
#!/usr/bin/python

import sys

def hi(s):
        print >> sys.stderr, s
```

```
#!/usr/bin/python

import sys

def hi(s):
        print &gt;&gt; sys.stderr, s
```

Nodes XML Tools <file>

- Create a file on the system:
  - **<file name="/etc/hi-mom" mode="append">**
    - **How are you today?**
  - **</file>**
- Used extensively throughout Rocks post sections
  - Keeps track of alterations automatically via RCS.

```
<file name="/etc/hi" perms="444">
How are you today?
I am fine.
</file>
```

```
…RCS checkin commands...
cat > /etc/hi << 'EOF'
How are you today?
I am fine.
EOF
chmod 444 /etc/hi-mom
…RCS cleanup commands…
```

Fancy <file>: nested tags

```
<file name="/etc/hi">

Here is your fortune for today:
<eval>
date +"%d-%b-%Y"
echo ""
/usr/games/fortune
</eval>

</file>
```

…RCS checkin commands...
cat > /etc/hi << 'EOF'

Here is your fortune for today:
13-May-2005

"Been through Hell? Whaddya bring back for me?"
-- A. Brilliant

EOF
…RCS cleanup commands…

Nodes Main

- Used to specify basic configuration:
  - timezone
  - mouse, keyboard types
  - install language
- Used more rarely than other tags
- Rocks main tags are usually a straight translation:

```
<main>

  <timezone>America/Mission_Beach
  </timezone>

</main>
```

```
…
timezone America/Mission_Beach
…
rootpw --iscrypted sndk48shdlwis
mouse genericps/2
url --url http://10.1.1.1/install/rocks-dist/..
```

Nodes Packages

- **`<package>java</package>`**
  - Specifies an RPM package. Version is automatically determined: take the *newest* rpm on the system with the name 'java'.
- **`<package arch="x86_64">java</package>`**
  - Only install this package on x86_64 architectures
- **`<package arch="i386,x86_64">java</package>`**

```
<package>newcastle</package>
<package>stone-pale</package>
<package>guinness</package>
```

```
%packages
newcastle
stone-pale
guinness
```

Nodes Packages

- RPM name is a basename (not fullname of RPM)
  - For example, RPM name of package below is ‘kernel’

```
# rpm -qip /home/install/rocks-dist/lan/i386/RedHat/RPMS/kernel-2.6.9-22.EL.i686.rpm
Name        : kernel                       Relocations: (not relocatable)
Version     : 2.6.9                           Vendor: CentOS
Release     : 22.EL                        Build Date: Sun 09 Oct 2005 03:01:51 AM WET
Install Date: (not installed)              Build Host: louisa.home.local
Group       : System Environment/Kernel    Source RPM: kernel-2.6.9-22.EL.src.rpm
Size        : 25589794                        License: GPLv2
Signature   : DSA/SHA1, Sun 09 Oct 2005 10:44:40 AM WET, Key ID a53d0bab443e1821
Packager    : Johnny Hughes <johnny@centos.org>
Summary     : the linux kernel (the core of the linux operating system)
Description :
The kernel package contains the Linux kernel (vmlinuz), the core of any
Linux operating system
```

Nodes Post

- `<post>` for *Post-Install* configuration scripts
- Configuration scripts in <post> section run after *all* RPMs have been installed.
  - Useful: you have all your software available
  - Scripts run in “target” environment: /etc in <post> will be /etc on the final installed system
- Scripts are always non-interactive
  - No Human is driving

Nodes Post

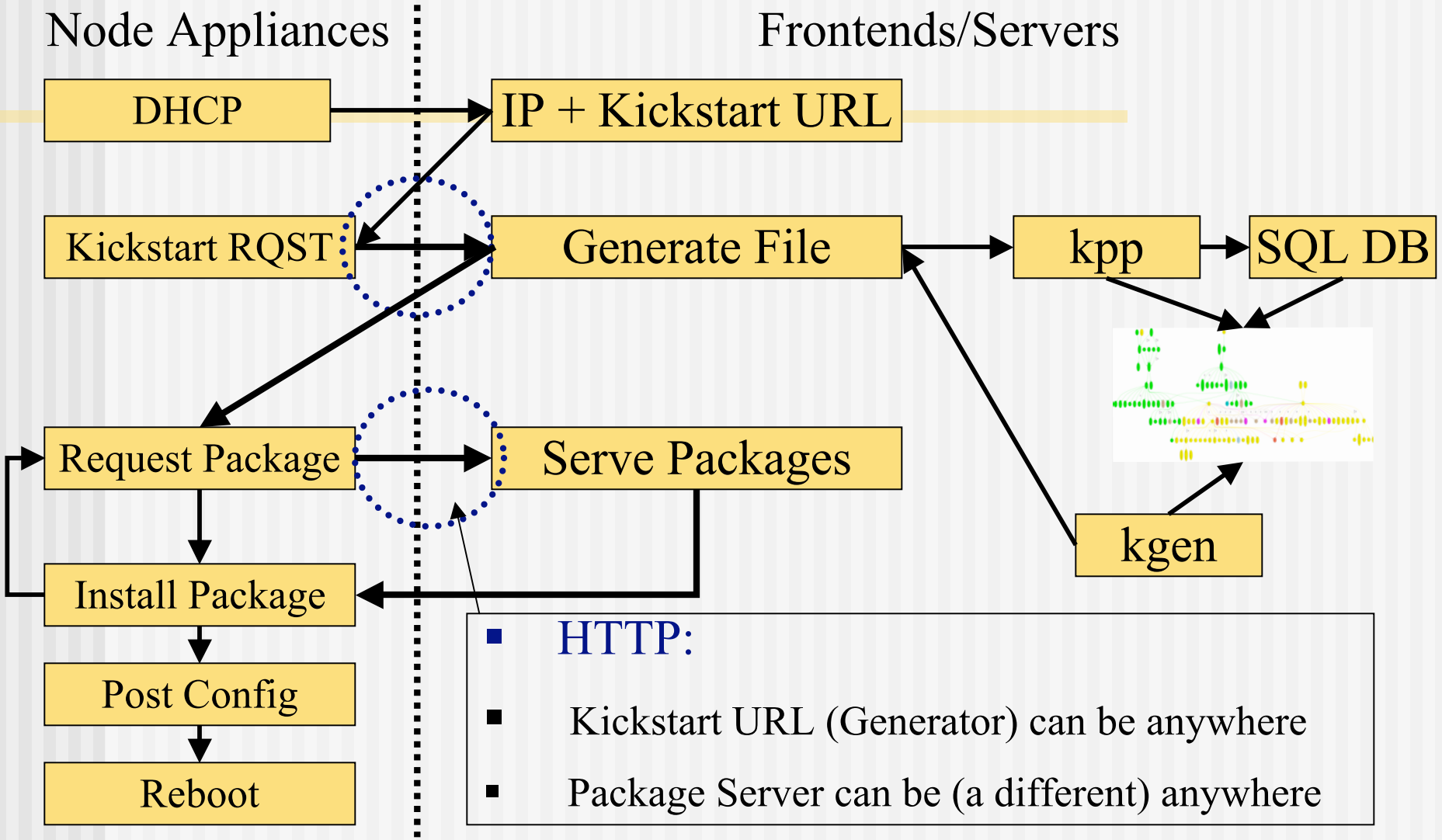
ntp-client.xml

```
<post>

/bin/mkdir -p /etc/ntp
/usr/sbin/ntpdate <var name="Kickstart_PrivateNTPHost"/>
/sbin/hwclock --systohc

</post>
```

```
%post

/bin/mkdir -p /etc/ntp
/usr/sbin/ntpdate 10.1.1.1
/sbin/hwclock --systohc
```

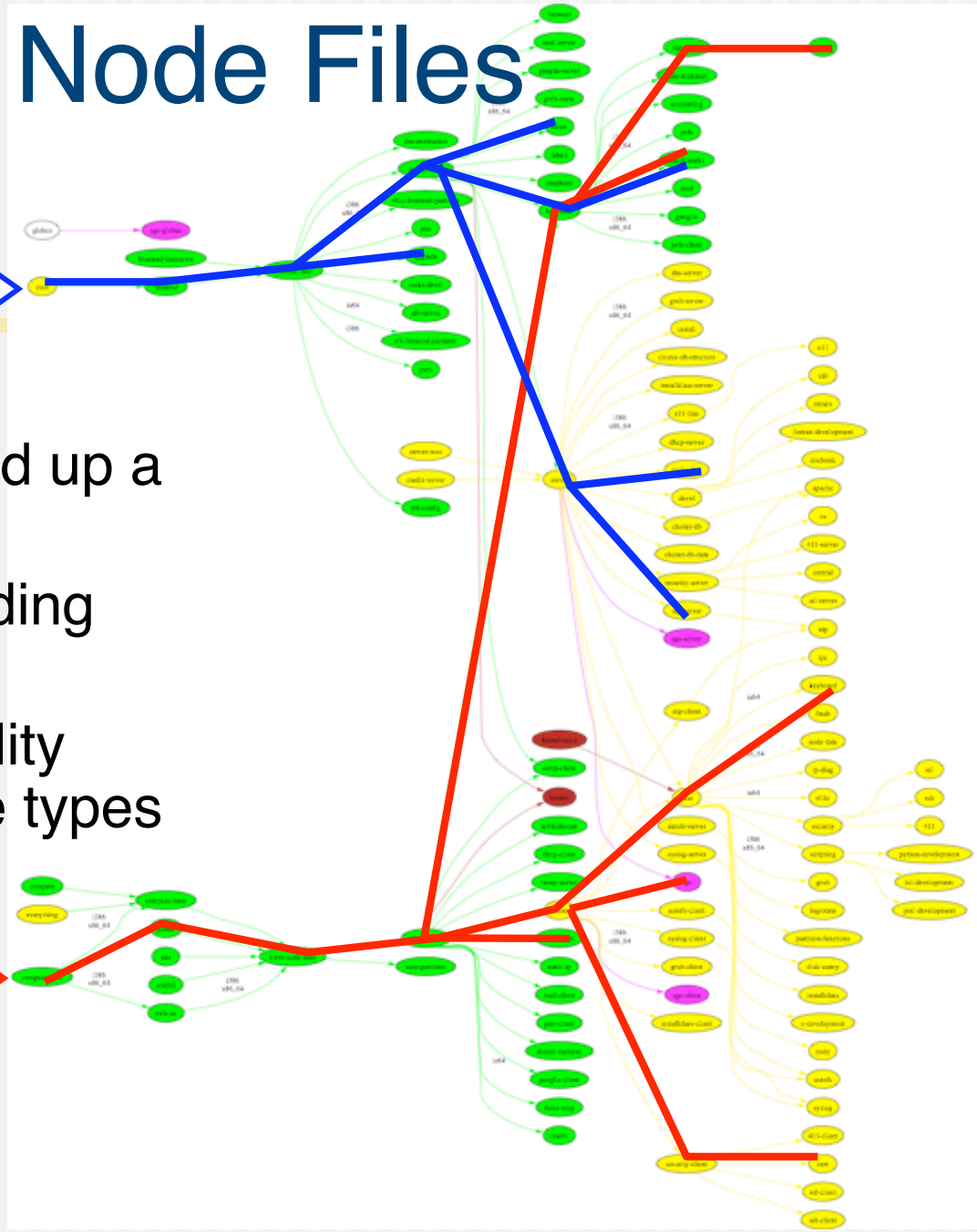
Putting it together

Space-Time and HTTP



- HTTP:
  - Kickstart URL (Generator) can be anywhere
  - Package Server can be (a different) anywhere

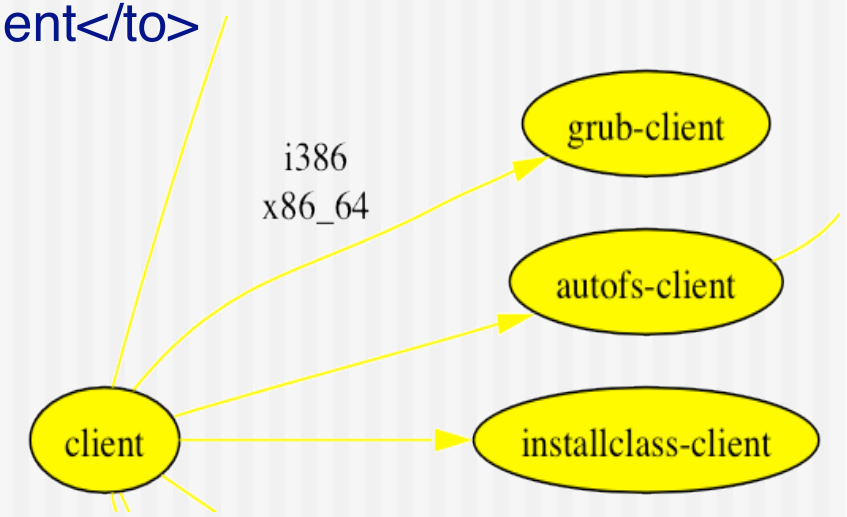
Coalescing Node Files



- Traverse a graph to build up a kickstart file
- Makes kickstart file building flexible
- Easy to share functionality between disparate node types

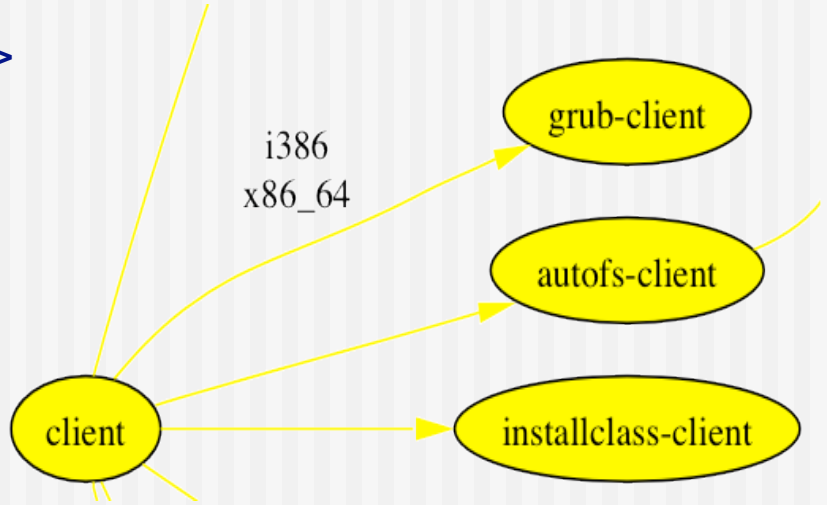
Another Look at XML

```
<graph>

    <edge from="client">
        <to arch="i386,x86_64">grub-client</to>
        <to>autofs-client</to>
        <to>installclass-client</to>
    </edge>

</graph>
```



Partial Ordering

```
<graph>
        <order head="autofs-client" tail="client"/>
        <edge from="client">
                <to arch="i386,x86_64">grub-client</to>
                <to>autofs-client</to>
                <to>installclass-client</to>
        </edge>

</graph>
```



- Forces autofs-client <post> section to run before client's <post> section
- In order graph traversal enforces a partial ordering
- Applying standard graph theory to system installation

Application Layer

- Rocks Rolls
  - Optional component
  - Created by SDSC
  - Created by others
- Example
  - Bio (BLAST)
  - Chem (GAMESS)
  - Visualization Clusters