Extending Functionality Through the Rocks Command Line

© 2008 UC Regents

It's Python Time!

© 2008 UC Regents

Motivation

- Lack of consistency in Rocks commands
  - add-extra-nic (15 flags)
  - 411put
  - rocks-dist
  - dbreport (~ a dozen reports)
- Extensible to other groups
  - How do I add a flag to an existing command?
  - How do I add a new command?
  - How do I document my command?

© 2008 UC Regents

Motivation

```
Usage: add-extra-nic [-hvv] [-p password] [-u host] [-d database] [--help]
 [--list-rcfiles] [--list-project-info] [--verbose] [--dump] [--del] [--list]
 [--verbose] [--no-update] [--no-modify] [--dryrun] [--rcfile arg] [--host host]
 [--password password] [--db database] [--user host]
 [--if interface (default: eth1)] [--mac mac address]
 [--module linux driver module name] [--ip ip address]
 [--netmask netmask (default /24)] [--gateway ip address of gateway]
 [--name hostname on new interface] [--site client ip] node

Usage: rocks-dist [-hvcpv] [-p password] [-u host] [-d database] [-a arch]
 [-d dirname] [-g path] [-l lang] [-r release] [--help] [--list-rcfiles]
 [--list-project-info] [--verbose] [--copy] [--debug] [--graph-draw-invis-edges]
 [--graph-draw-order] [--graph-draw-edges] [--graph-draw-key] [--graph-draw-all]
 [--graph-draw-landscape] [--install] [--verbose] [--with-rolls-only] [--clean]
 [--notorrent] [--rcfile arg] [--host host] [--password password]
 [--db database] [--user host] [--arch architecture] [--comps path]
 [--dist dirname] [--graph-draw-size arg] [--graph-draw-format arg]
 [--mirror-dir dirname] [--mirror-host hostname] [--root dirname]
 [--cdrom /mnt/cdrom] [--with-roll rollname-rollversion]
 [--path single path item] command
Available commands:
dist dvd makecontrib makesitenodes copycd usb copyroll cdrom paths graph dist2mirror
```

© 2008 UC Regents

Goals

- Consistent
  - Interface
  - Argument parsing
  - Usage / Help
- Extensible
  - Easy to add commands (3rd party rolls)
  - Easy to modify commands
- Easy to guess the right command
- Purge all –flags from Rocks (see previous slide)
- Inspired by Trac

© 2008 UC Regents

Verb Based

- “add”, “set”, “enable”, …
  - Modify the cluster database
- “list”, “dump”, “report”
  - Inspect the cluster database
- About 20 verbs in the command line so far
- You can even add your own

© 2008 UC Regents

Grammar

- rocks <verb> <object…> <subject> <params…>
  - Object is general to specific
    - “host” “interface”
    - “network” “subnet”
    - “viz” “layout”
  - Subject is typed
    - host
    - appliance
    - Network
- Example:

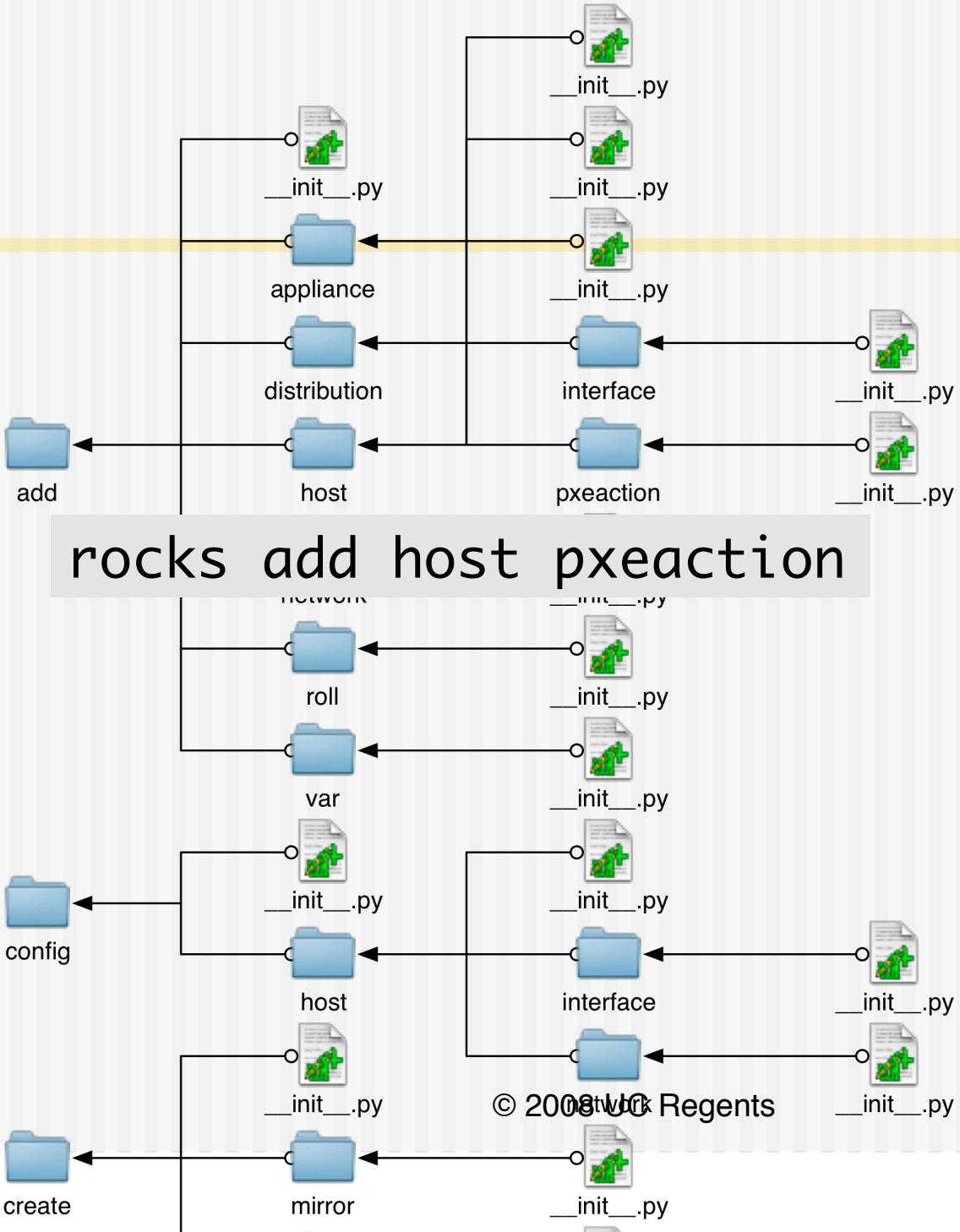
```
# rocks list host interface tile-0-0
SUBNET  IFACE MAC               IP             NETMASK   GATEWAY MODULE NAME
private eth0  00:13:72:ba:be:42 10.255.255.254 255.0.0.0 ------- tg3    tile-0-0
------- eth1  00:0a:5e:1a:6d:64 -------------- --------- ------- skge   --------
```

© 2008 UC Regents

Implementation

- Python
  - Similar to existing dbreport code
  - Very small modules
- Command line is identical to the directory hierarchy
  - Verbs are directories
  - Objects are directories
  - Subjects are __init__.py files
- Commands are added by adding directories

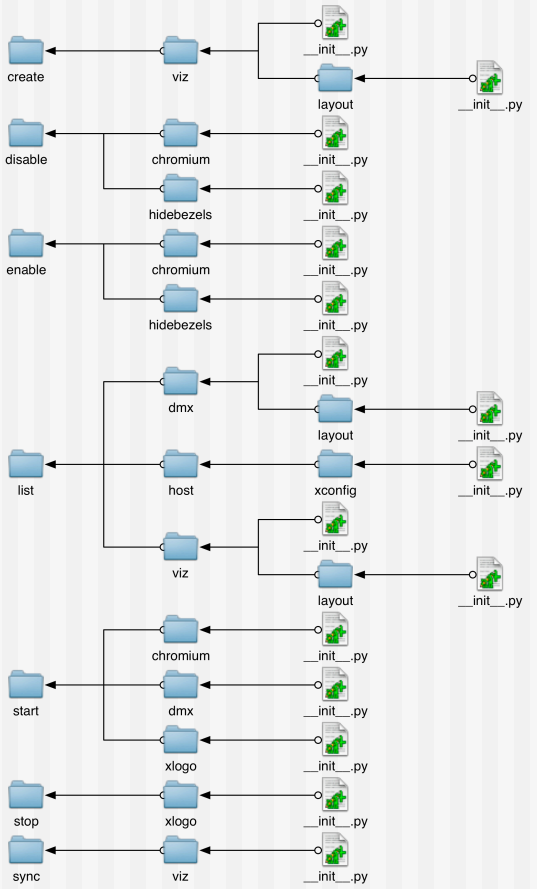
© 2008 UC Regents



© 2008 UC Regents

Rolls Can Add Commands

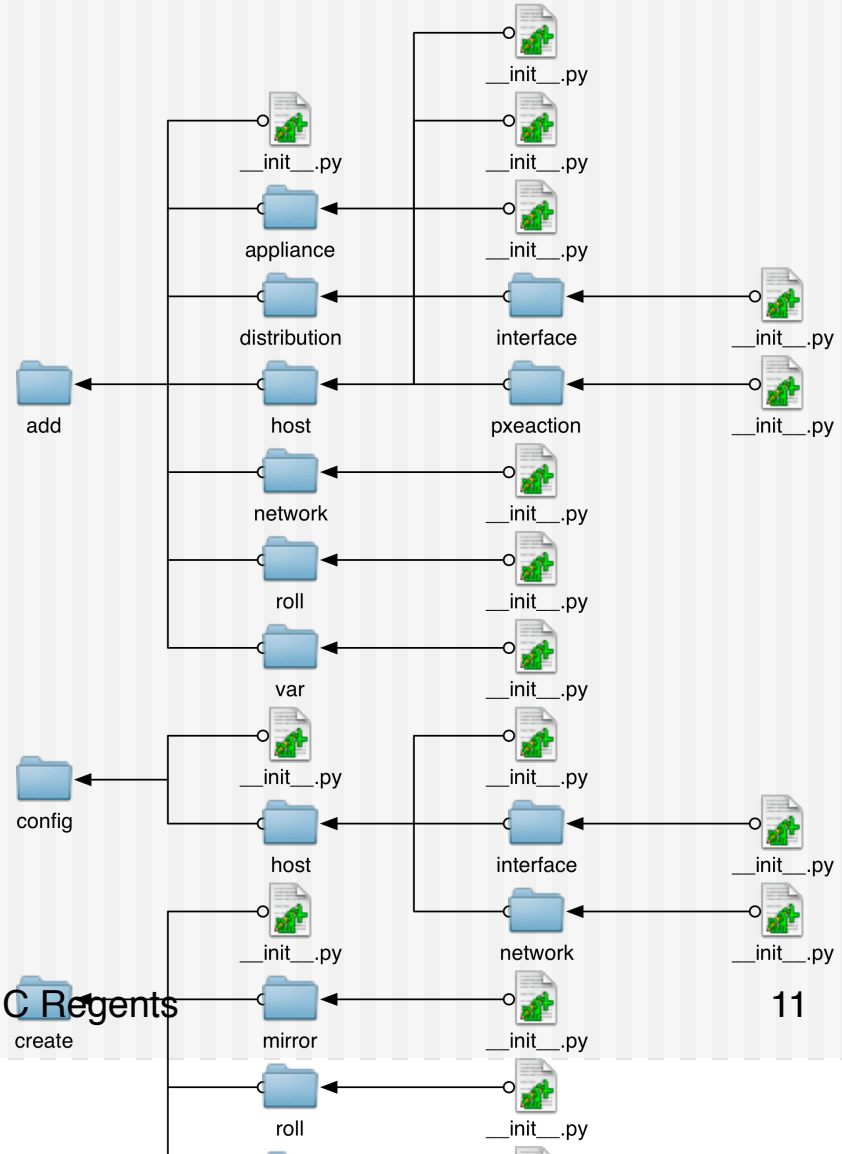
- Similar to the configuration graph
- Rolls can add command line
  - Files : commands
  - Directories : verbs and objects
- Think hard before adding another verb



© 2008 UC Regents

add

- Creates new entries in the cluster database
- Examples:
  - Hosts
  - Appliances
  - Rolls



© 2008 UC Regents

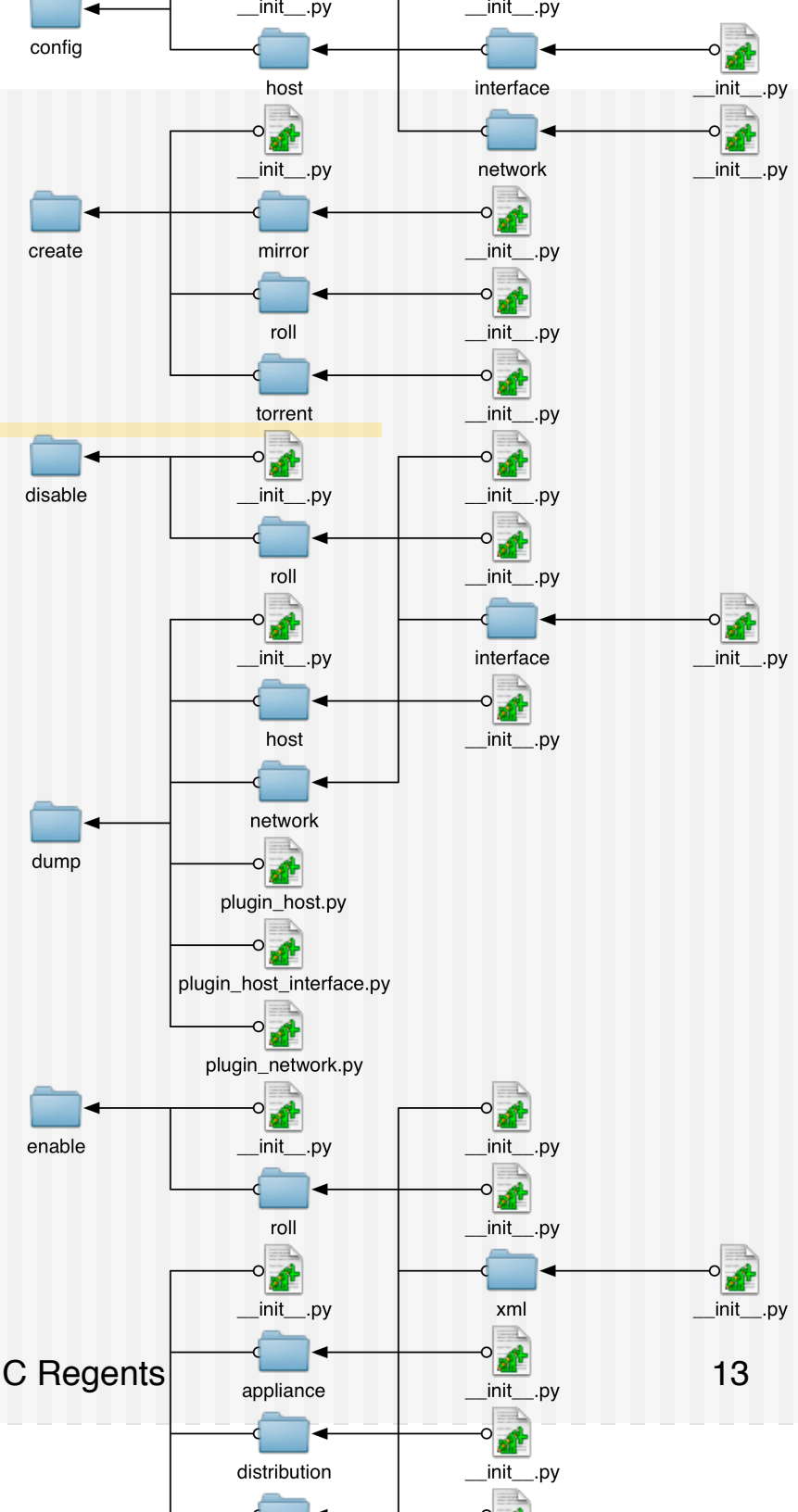
rocks add distribution

```
import rocks.commands

class Command(rocks.commands.DistributionArgumentProcessor,
        rocks.commands.add.command):
        """
        Add a distribution specification to the database.

        <arg type='string' name='distribution'>
        Name of the new distribution.
        </arg>

        <example cmd='add distribution rocks-dist'>
        Adds the distribution named "rocks-dist" into the database.
        </example>
        """

        def run(self, params, args):

                if len(args) != 1:
                        self.abort('must supply one distribution')
                dist = args[0]

                if dist in self.getDistributionNames():
                        self.abort('distribution "%s" exists' % dist)

                self.db.execute("""insert into distributions (name) values
                        ('%s')""" % dist)


```

© 2008 UC Regents

dump

- Returns cluster database information in the form of rocks command lines
- Examples:
  - Hosts
  - Network
- Same as –dump flag on insert-ethers



© 2008 UC Regents

rocks dump host

```
# rocks dump host
/opt/rocks/bin/rocks add host vizagra cpus=1 rack=0 rank=0 membership="Frontend"
/opt/rocks/bin/rocks add host tile-0-1 cpus=2 rack=0 rank=1 membership="Tile"
/opt/rocks/bin/rocks add host tile-0-0 cpus=2 rack=0 rank=0 membership="Tile"
/opt/rocks/bin/rocks add host tile-0-2 cpus=2 rack=0 rank=2 membership="Tile"
/opt/rocks/bin/rocks add host tile-0-3 cpus=2 rack=0 rank=3 membership="Tile"
/opt/rocks/bin/rocks add host tile-1-3 cpus=2 rack=1 rank=3 membership="Tile"
/opt/rocks/bin/rocks add host tile-1-2 cpus=2 rack=1 rank=2 membership="Tile"
/opt/rocks/bin/rocks add host tile-1-1 cpus=2 rack=1 rank=1 membership="Tile"
/opt/rocks/bin/rocks add host tile-1-0 cpus=2 rack=1 rank=0 membership="Tile"
/opt/rocks/bin/rocks add host tile-2-0 cpus=2 rack=2 rank=0 membership="Tile"
/opt/rocks/bin/rocks add host tile-2-1 cpus=2 rack=2 rank=1 membership="Tile"
/opt/rocks/bin/rocks add host tile-2-2 cpus=2 rack=2 rank=2 membership="Tile"
/opt/rocks/bin/rocks add host tile-2-3 cpus=2 rack=2 rank=3 membership="Tile"
/opt/rocks/bin/rocks add host tile-3-0 cpus=2 rack=3 rank=0 membership="Tile"
/opt/rocks/bin/rocks add host tile-3-1 cpus=2 rack=3 rank=1 membership="Tile"
/opt/rocks/bin/rocks add host tile-3-2 cpus=2 rack=3 rank=2 membership="Tile"
/opt/rocks/bin/rocks add host tile-3-3 cpus=2 rack=3 rank=3 membership="Tile"
/opt/rocks/bin/rocks add host tile-4-0 cpus=2 rack=4 rank=0 membership="Tile"
/opt/rocks/bin/rocks add host tile-4-1 cpus=2 rack=4 rank=1 membership="Tile"
/opt/rocks/bin/rocks add host tile-4-2 cpus=2 rack=4 rank=2 membership="Tile"
/opt/rocks/bin/rocks add host tile-4-3 cpus=2 rack=4 rank=3 membership="Tile"
```

© 2008 UC Regents

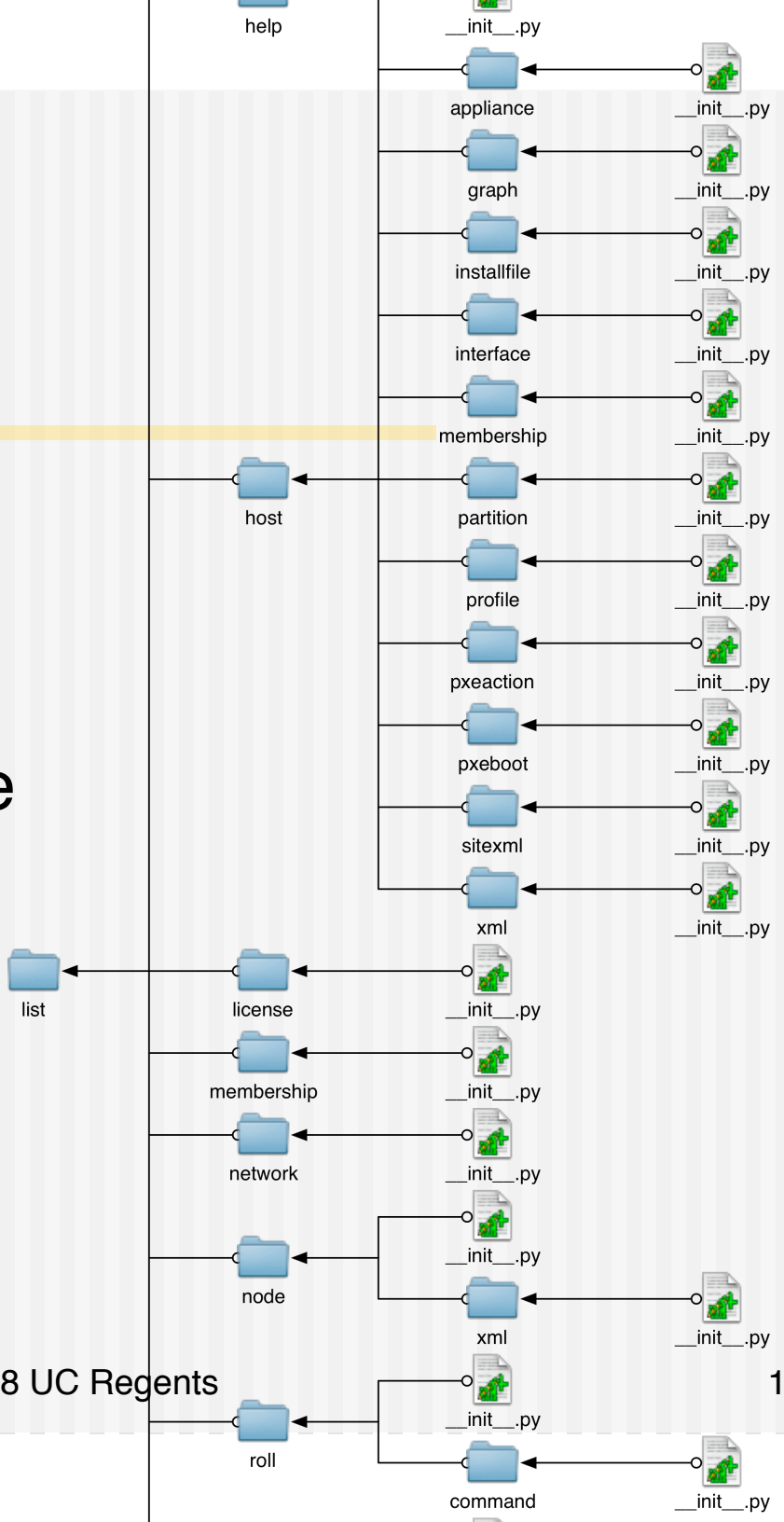
rocks dump host

```
1: import os
2: import sys
3: import string
4: import rocks.commands
5:
6: class command(rocks.commands.HostArgumentProcessor,
7:         rocks.commands.dump.command):
8:         pass
9:
10: class Command(command):
11:         """
12:         Dump the host information as rocks commands.
13:
14:         <arg optional='1' type='string' name='host' repeat='1'>
15:         Zero, one or more host names. If no host names are supplied,
16:         information for all hosts will be listed.
17:         </arg>
18:
19:         <example cmd='dump host compute-0-0'>
20:         Dump host compute-0-0 information.
21:         </example>
22:
23:         <example cmd='dump host compute-0-0 compute-0-1'>
24:         Dump host compute-0-0 and compute-0-1 information.
25:         </example>
26:
27:         <example cmd='dump host'>
28:         Dump all hosts.
29:         </example>
30:         """
31:
32:         def run(self, params, args):
33:                 for host in self.getHostnames(args):
34:                         self.db.execute("""select
35:                                 n.cpus, n.rack, n.rank, m.name
36:                                 from nodes n, memberships m where
37:                                 n.membership=m.id and n.name='%s'""" % host)
38:                         (cpus, rack, rank, membership) = self.db.fetchone()
39:                         self.dump('add host %s cpus=%s rack=%s rank=%s '
40:                                 'membership="%s"' %
41:                                 (host, cpus, rack, rank, membership))
42:
43:
```

© 2008 UC Regents

list

- Reports information in human readable format
- No side-effects on the database
- Examples:
  - Hosts
  - Appliances
  - Rolls



© 2008 UC Regents

rocks list host

```
# rocks list host
HOST      MEMBERSHIP CPUS RACK RANK COMMENT
vizagra:  Frontend   1    0    0    -------
tile-0-1: Tile       2    0    1    -------
tile-0-0: Tile       2    0    0    -------
tile-0-2: Tile       2    0    2    -------
tile-0-3: Tile       2    0    3    -------
tile-1-3: Tile       2    1    3    -------
tile-1-2: Tile       2    1    2    -------
tile-1-1: Tile       2    1    1    -------
tile-1-0: Tile       2    1    0    -------
tile-2-0: Tile       2    2    0    -------
tile-2-1: Tile       2    2    1    -------
tile-2-2: Tile       2    2    2    -------
tile-2-3: Tile       2    2    3    -------
tile-3-0: Tile       2    3    0    -------
tile-3-1: Tile       2    3    1    -------
tile-3-2: Tile       2    3    2    -------
tile-3-3: Tile       2    3    3    -------
tile-4-0: Tile       2    4    0    -------
tile-4-1: Tile       2    4    1    -------
tile-4-2: Tile       2    4    2    -------
tile-4-3: Tile       2    4    3    -------
```

© 2008 UC Regents

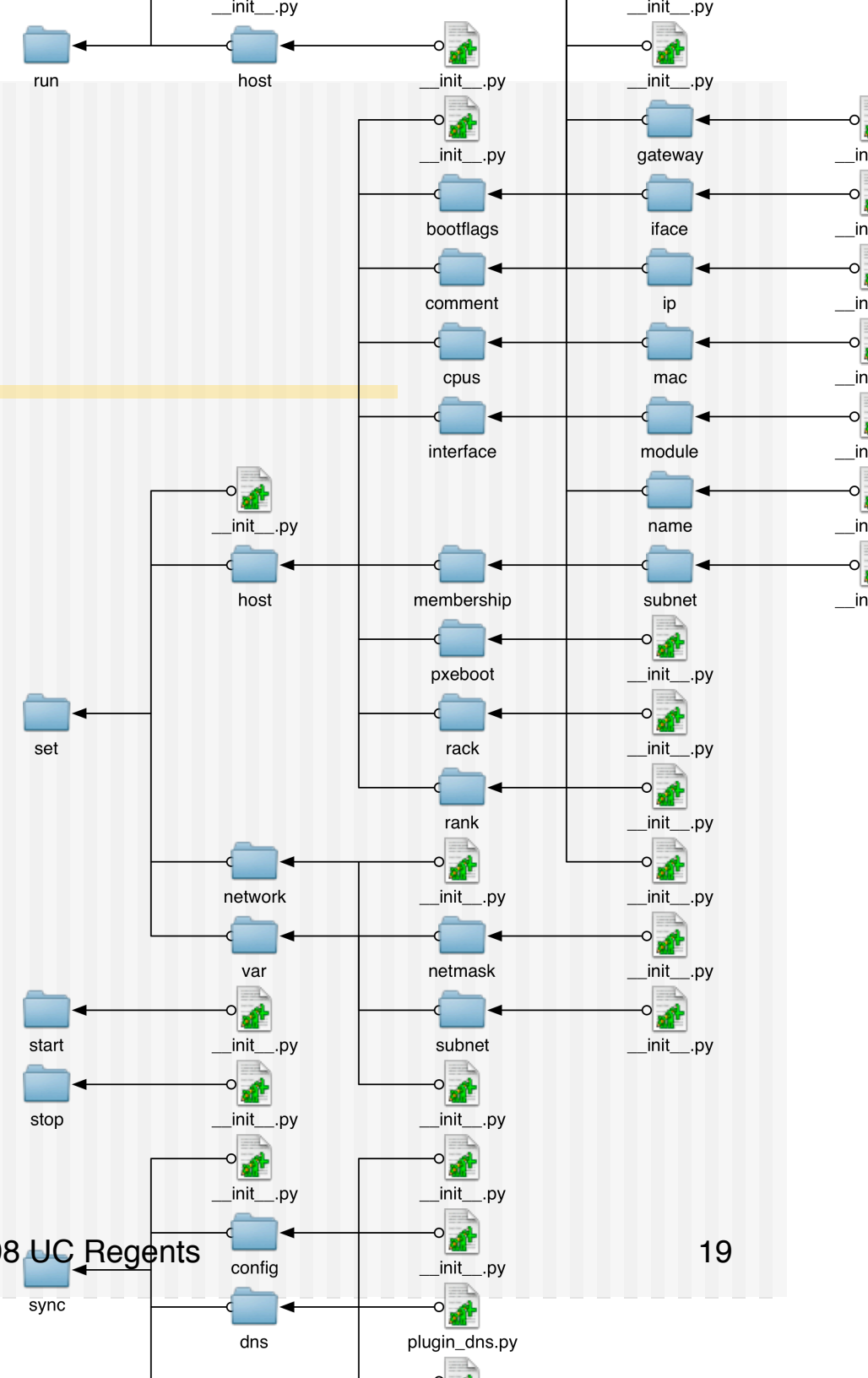
rocks list host

```
1: import rocks.commands
2:
3: class command(rocks.commands.HostArgumentProcessor,
4:         rocks.commands.list.command):
5:         pass
6:
7: class Command(command):
8:         """
9:         List the membership, CPU count, physical position info and comment for
10:        a list of hosts.
11:
12:        <arg optional='1' type='string' name='host' repeat='1'>
13:        Zero, one or more host names. If no host names are supplied, info about
14:        all the known hosts is listed.
15:        </arg>
16:
17:        <example cmd='list host compute-0-0'>
18:        List info for compute-0-0.
19:        </example>
20:
21:        <example cmd='list host'>
22:        List info for all known hosts.
23:        </example>
24:        """
25:
26:        def run(self, params, args):
27:                self.beginOutput()
28:
29:                for host in self.getHostnames(args):
30:                        self.db.execute("""select m.name, n.cpus,
31:                                n.rack, n.rank, n.comment from
32:                                nodes n, memberships m where
33:                                n.membership=m.id and n.name='%s'""" % host)
34:                        self.addOutput(host, self.db.fetchone())
35:
36:                self.endOutput(header=['host', 'membership',
37:                        'cpus', 'rack', 'rank', 'comment'])
38:
```

© 2008 UC Regents

set

- Modifies entries in the cluster database
- Examples:
  - Network Interfaces
  - Appliance Assignment
  - Rack / Rank
- add-extra-nic
  - Rocks add host interface
  - Rocks set host interface



© 2008 UC Regents

add-extra-nic is now …

- Start

```
# rocks list host interface compute-1-1
SUBNET  IFACE MAC               IP             NETMASK   GATEWAY MODULE NAME
private eth0  00:0e:0c:5d:7e:5e 10.255.255.251 255.0.0.0 ------- e1000  compute-1-1
------- eth1  00:30:1b:b2:ea:61 -------------- --------- ------- tg3    -----------
```

- Configure

```
# rocks set host interface ip compute-1-1 eth1 192.168.1.1
# rocks set host interface gateway compute-1-1 eth1 192.168.1.254
# rocks set host interface name compute-1-1 eth1 fast-1-1
# rocks set host interface subnet compute-1-1 eth1 public
```

- Verify

```
# rocks list host interface compute-1-1
SUBNET  IFACE MAC               IP             NETMASK       GATEWAY       MODULE NAME
private eth0  00:0e:0c:5d:7e:5e 10.255.255.251 255.0.0.0     ------------- e1000  compute-1-1
public  eth1  00:30:1b:b2:ea:61 192.168.1.1    255.255.255.0 192.168.1.254 tg3    fast-1-1
```

© 2008 UC Regents

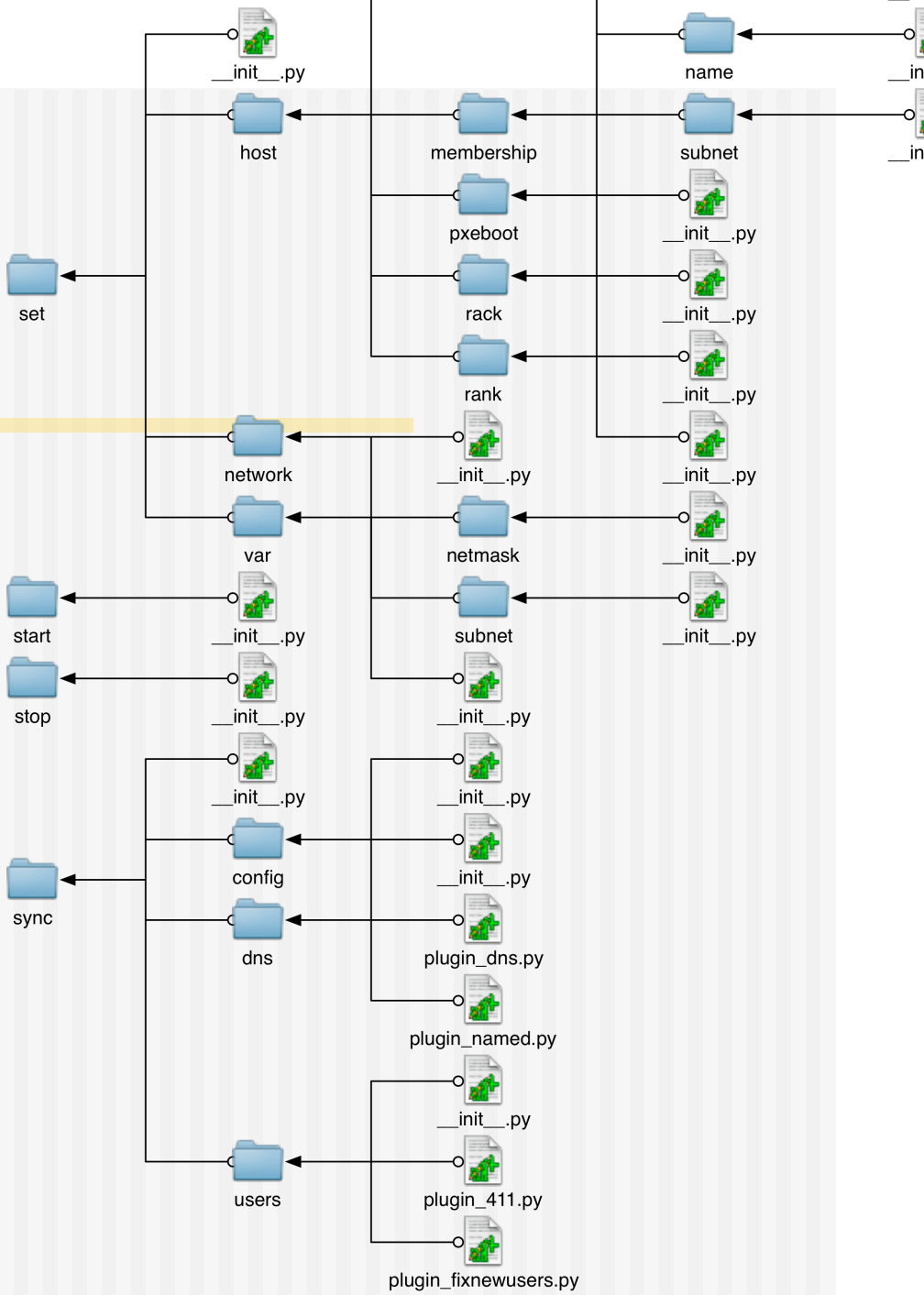
rocks set host interface ip

```
import rocks.commands

class Command(rocks.commands.set.host.command):
        """
        Sets the IP address for the named interface for one host.

        <arg type='string' name='host'>
        Host name.
        </arg>

        <arg type='string' name='iface'>
        Interface that should be updated. This may be a logical interface
        the IP address of the interface.
        </arg>

        <arg type='string' name='ip'>
        The IP address of the interface. Usually of the form nnn.nnn.nnn.nnn
        where n is a decimal digit. This format is not enforced. Use IP=NULL
        to clear.
        </arg>

        <param type='string' name='iface'>
        Can be used in place of the iface argument.
        </param>

        <param type='string' name='ip'>
        Can be used in place of the ip argument.
        </param>


        <example cmd='set host interface ip compute-0-0 eth1 192.168.0.10'>
        Sets the IP Address for the eth1 device on host compute-0-0.
        </example>

        <example cmd='set host interface ip compute-0-0 iface=eth1 ip=192.168.0.10'>
        Same as above.
        </example>

        <related>set host interface iface</related>
        <related>set host interface ip</related>
        <related>set host interface gateway</related>
        <related>set host interface module</related>
        <related>add host</related>
        """

        def run(self, params, args):

                (args, iface, ip) = self.fillPositionalArgs(('iface', 'ip'))

                hosts = self.getHostnames(args)

                if len(hosts) != 1:
                        self.abort('must supply one host')
                if not iface:
                        self.abort('must supply iface')
                if not ip:
                        self.abort('must supply ip')

                ip = ip.upper() # null -> NULL

                for host in hosts:
                        self.db.execute("""update networks, nodes set
                                networks.ip=NULLIF('%s','NULL') where
                                nodes.name='%s' and networks.node=nodes.id and
                                (networks.device='%s' or networks.mac='%s')""" %
                                (ip, host, iface, iface))
```

© 2008 UC Regents

start / stop

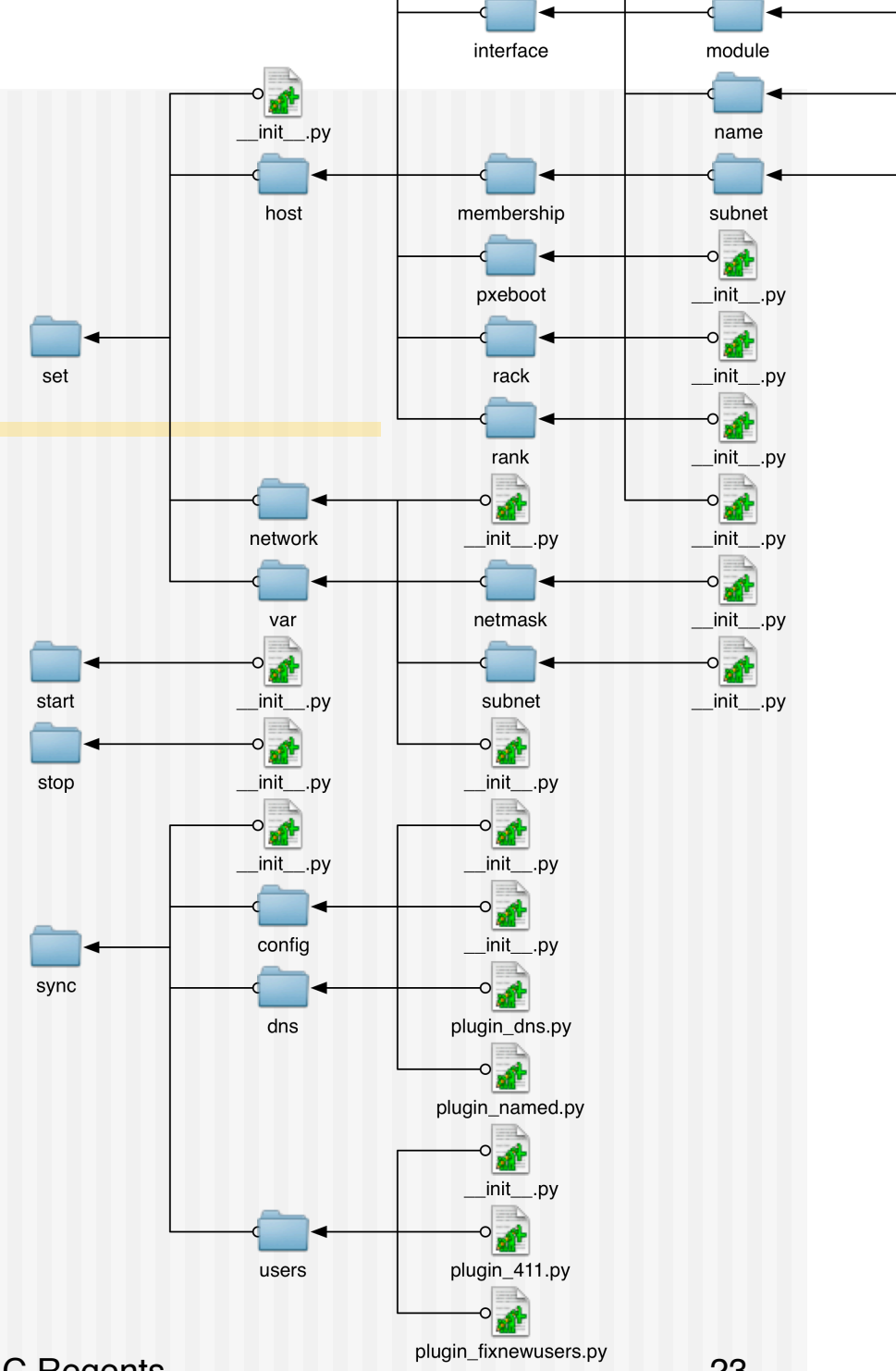
- Start and stop something
- NULL commands
- Reserve the verbs for use on other Rolls
- Think “abstract base class”



© 2008 UC Regents

sync

- Synchronizes the database state to software configuration files
- Similar to the old “insert-ethers –update”
- Not complete yet, see Rocks VI (5.1)



© 2008 UC Regents

rocks sync config

```
1: import os
2: import sys
3: import string
4: import rocks.file
5: import rocks.commands
6:
7:
8: class Command(rocks.commands.sync.command):
9:         """
10:        For each system configuration file controlled by Rocks, first
11:        rebuild the configuration file by extracting data from the
12:        database, then restart the relevant services.
13:
14:        <example cmd='sync config'>
15:        Rebuild all configuration files and restart relevant services.
16:        </example>
17:        """
18:
19:        def run(self, params, args):
20:                cmd = '/opt/rocks/sbin/insert-ethers --update'
21:                for line in os.popen(cmd).readlines():
22:                        self.owner.addText(line)
23:
```

© 2008 UC Regents

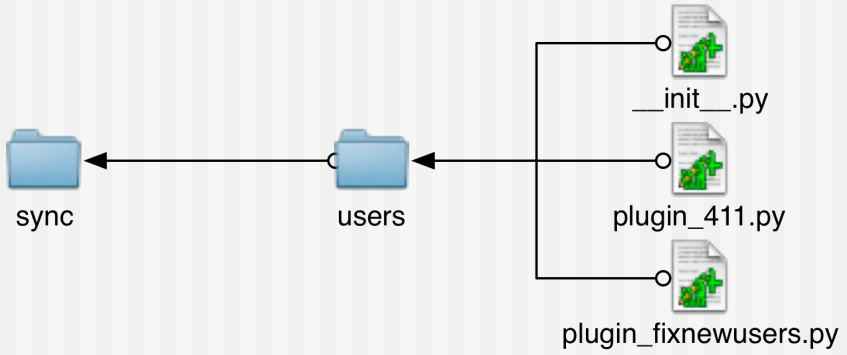
Extensibility

- New commands
  - Add directories
  - Add __init__.py code
- Existing commands
  - Some commands can be extended
  - Plugins

© 2008 UC Regents

rocks sync users

- Run after useradd
  - Populate auto.home
  - Cleanup password file
  - Send 411 files
- Two plugins
  - Fixnewusers
  - 411
- Partial Ordering
- Other Rolls can add more plugins to this command
- Command must be design for plugins (not default)



© 2008 UC Regents

__init__.py

```
1:  import rocks.commands
2:
3:
4:  class Command(rocks.commands.sync.command):
5:          """
6:          Update all user-related files (e.g., /etc/passwd, /etc/shadow, etc.)
7:          on all known hosts. Also, restart autofs on all known hosts.
8:
9:          <example cmd='sync users'>
10:         Send all user info to all known hosts.
11:         </example>
12:         """
13:
14:         def run(self, params, args):
15:                 self.runPlugins()
16:
```

© 2008 UC Regents

411 plugin

```
1: import os
2: import rocks.commands
3:
4: class Plugin(rocks.commands.Plugin):
5:         """Force a 411 update and re-load autofs on all nodes"""
6:
7:         def provides(self):
8:                 return '411'
9:
10:        def requires(self):
11:                return ['fixnewusers']
12:
13:        def run(self, args):
14:                #
15:                # force the rebuild of all files under 411's control
16:                #
17:                for line in os.popen('make -C /var/411 force').readlines():
18:                        self.owner.addText(line)
19:
20:                #
21:                # restart autofs on all known hosts
22:                #
23:                cmd = '/opt/rocks/bin/tentakel "service autofs reload"'
24:                for line in os.popen(cmd).readlines():
25:                        self.owner.addText(line)
26:
27:
```

© 2008 UC Regents

auto.home / passwd plugin

```
import os
import string
import rocks.commands

class Plugin(rocks.commands.Plugin):
        """Relocates home directories to /export and fixes autofs.home"""

        def provides(self):
                return 'fixnewusers'

        def run(self, args):
                # scan the password file for any '/export/home' entries
                # this is the default entry as setup by useradd
                new_users = []
                default_dir = '/export/home/'

                file = open('/etc/passwd', 'r')

                for line in file.readlines():
                        l = string.split(line[:-1], ':')

                        if len(l) < 6:
                                continue

                        username = l[0]
                        homedir = l[5]

                        if homedir[:len(default_dir)] == default_dir:
                                new_users.append(username)
                file.close()

                hostname = '%s.%s' % \
                        (self.db.getGlobalVar('Kickstart', 'PrivateHostname'),
                        self.db.getGlobalVar('Kickstart', 'PrivateDNSDomain'))

                for user in new_users:

                        # for each new user, change their default directory to
                        # /home/<username>

                        cmd = '/usr/sbin/usermod -d %s %s' % \
                                (os.path.join('/home', user), user)
                        for line in os.popen(cmd).readlines():
                                self.owner.addText(line)

                        # then update the auto.home file
```

© 2008 UC Regents

Argument Processing

- rocks <verb> <object…> <subject> <params…>
- Subject is typed by first object
  - host -> one or more hostname
  - roll -> one or more roll names
- Params are in key=value form
- Same as –flag=value but easier to read

© 2008 UC Regents

Helper classes and functions

- ArgumentProcessors
  - Class to parse the subject in a standard way
  - Exists for hosts, rolls, appliances, …
- Parameters Parsing
  - fillPositionalArgs
  - fillParams

© 2008 UC Regents

HostArgumentProcessor

- Command must derive from `rocks.commands.HostArgumentProcessor`
- `self.getHostnames(args)`
  - Return a list of hostname as they appear in the cluster database
  - If `args = None` all the host in the cluster are returned
  - `args` can also be a group
    - Rack0, rack1
  - Or an appliance type
    - Compute, Tile, …

© 2008 UC Regents


```
1:  import rocks.commands
2:
3:  class command(rocks.commands.HostArgumentProcessor,
4:          rocks.commands.list.command):
5:          pass
6:
7:  class Command(command):
8:          """
9:          List the membership, CPU count, physical position info and comment for
10:         a list of hosts.
11:
12:         <arg optional='1' type='string' name='host' repeat='1'>
13:         Zero, one or more host names. If no host names are supplied, info about
14:         all the known hosts is listed.
15:         </arg>
16:
17:         <example cmd='list host compute-0-0'>
18:         List info for compute-0-0.
19:         </example>
20:
21:         <example cmd='list host'>
22:         List info for all known hosts.
23:         </example>
24:         """
25:
26:         def run(self, params, args):
27:                 self.beginOutput()
28:
29:                 for host in self.getHostnames(args):
30:                         self.db.execute("""select m.name, n.cpus,
31:                                 n.rack, n.rank, n.comment from
32:                                 nodes n, memberships m where
33:                                 n.membership=m.id and n.name='%s'""" % host)
34:                         self.addOutput(host, self.db.fetchone())
35:
36:                 self.endOutput(header=['host', 'membership',
37:                         'cpus', 'rack', 'rank', 'comment'])
38:
```

© 2008 UC Regents

args = None

```
# rocks list host
HOST       MEMBERSHIP CPUS RACK RANK COMMENT
vizagra:   Frontend   1    0    0    -------
tile-0-1: Tile        2    0    1    -------
tile-0-0: Tile        2    0    0    -------
tile-0-2: Tile        2    0    2    -------
tile-0-3: Tile        2    0    3    -------
tile-1-3: Tile        2    1    3    -------
tile-1-2: Tile        2    1    2    -------
tile-1-1: Tile        2    1    1    -------
tile-1-0: Tile        2    1    0    -------
tile-2-0: Tile        2    2    0    -------
tile-2-1: Tile        2    2    1    -------
tile-2-2: Tile        2    2    2    -------
tile-2-3: Tile        2    2    3    -------
```

© 2008 UC Regents

args = list of hosts

```
# rocks list host tile-0-0 10.255.255.253 tile-3-0.local
HOST      MEMBERSHIP CPUS RACK RANK COMMENT
tile-0-0: Tile       2    0    0    -------
tile-0-1: Tile       2    0    1    -------
tile-3-0: Tile       2    3    0    -------
```

© 2008 UC Regents

args = rack

```
# rocks list host rack2
HOST       MEMBERSHIP CPUS RACK RANK COMMENT
tile-2-0: Tile        2    2    0    -------
tile-2-1: Tile        2    2    1    -------
tile-2-2: Tile        2    2    2    -------
tile-2-3: Tile        2    2    3    -------
```

© 2008 UC Regents

args = appliance type

```
# rocks list host tile
HOST       MEMBERSHIP CPUS RACK RANK COMMENT
tile-0-0: Tile        2    0    0    -------
tile-0-1: Tile        2    0    1    -------
tile-0-2: Tile        2    0    2    -------
tile-0-3: Tile        2    0    3    -------
tile-1-0: Tile        2    1    0    -------
tile-1-1: Tile        2    1    1    -------
tile-1-2: Tile        2    1    2    -------
tile-1-3: Tile        2    1    3    -------
tile-2-0: Tile        2    2    0    -------
tile-2-1: Tile        2    2    1    -------
```

© 2008 UC Regents

Any combination is fine

```
# rocks list host tile-2-0 rack1 frontend
HOST       MEMBERSHIP CPUS RACK RANK COMMENT
tile-1-0: Tile        2    1    0    -------
tile-1-1: Tile        2    1    1    -------
tile-1-2: Tile        2    1    2    -------
tile-1-3: Tile        2    1    3    -------
tile-2-0: Tile        2    2    0    -------
vizagra:  Frontend    1    0    0    -------
```

© 2008 UC Regents

ArgumentProcessors

Class Name	Helper Function
ApplianceArgumentProcessor	getApplianceNames
DistributionArgumentProcessor	getDistributionNames
HostArgumentProcessors	getHostnames
MembershipArgumentProcessor	getMembershipNames
NetworkArgumentProcessor	getNetworkNames
RollArgumentProcessor	getRollNames

© 2008 UC Regents

RollArgumentProcessor

```
 1: import os
 2: import stat
 3: import time
 4: import sys
 5: import string
 6: import rocks.commands
 7:
 8:
 9: class Command(rocks.commands.RollArgumentProcessor,
10:         rocks.commands.list.command):
11:         """
12:         List the status of available rolls.
13:
14:         <arg optional='1' type='string' name='roll' repeat='1'>
15:         List of rolls. This should be the roll base name (e.g., base, hpc,
16:         kernel). If no rolls are listed, then status for all the rolls are
17:         listed.
18:         </arg>
19:
20:         <example cmd='list roll kernel'>
21:         List the status of the kernel roll
22:         </example>
23:
24:         <example cmd='list roll'>
25:         List the status of all the available rolls
26:         </example>
27:         """
28:
29:         def run(self, params, args):
30:
31:                 self.beginOutput()
32:                 for (roll, version) in self.getRollNames(args, params):
33:                         self.db.execute("""select version, arch, enabled from
34:                                 rolls where name='%s' and version='%s'""" %
35:                                 (roll, version))
36:                         for row in self.db.fetchall():
37:                                 self.addOutput(roll, row)
38:
39:                 self.endOutput(header=['name', 'version', 'arch', 'enabled'],
40:                         trimOwner=0)
41:
```

© 2008 UC Regents

No Parameter

```
# rocks list roll
NAME         VERSION ARCH ENABLED
viz:         5.0     i386 yes
sge:         5.0     i386 yes
kernel:      5.0     i386 yes
updates:     5.1     i386 yes
java:        4.3.2   i386 yes
xen:         5.0     i386 yes
CentOS:      5.1     i386 yes
ganglia:     5.0     i386 yes
web-server:  5.0     i386 yes
base:        5.0     i386 yes
```

© 2008 UC Regents

Version Parameter

```
# rocks list roll version=4.3.2
NAME  VERSION ARCH ENABLED
java: 4.3.2   i386 yes
```

© 2008 UC Regents

Summary

- ArgumentProcessors standardize the handling of command line subjects
- Calling the helper function with an empty list returns all subject in the database
- HostArgumentProcessor knows about more than just host names
- RollArgumentProcessor can filter on versions

© 2008 UC Regents

fillParams

- Create local variables based on command parameters (key=value)
- Argument a list of (key, default) tuples
- If the parameter is not found on the command line the default value is used

© 2008 UC Regents

```
33:         </example>
34:         """
35:
36:         def maketorrent(self, filename, data):
37:                 info = {}
38:                 info['length'] =  os.stat(filename)[stat.ST_SIZE]
39:                 info['name'] =  os.path.basename(filename)
40:
41:                 data['info'] = info
42:
43:                 encoded = BitTorrent.bencode.bencode(data)
44:
45:                 file = open('%s.torrent' % (filename), 'w')
46:                 file.write(encoded)
47:                 file.close()
48:
49:
50:         def run(self, params, args):
51:
52:                 if len(args) != 1:
53:                         self.abort('must supply one file')
54:                 filename = args[0]
55:
56:                 (timestamp, ) = self.fillParams([('timestamp', time.time())])
57:                 try:
58:                         creation_date = int(timestamp)
59:                 except:
60:                         creation_date = int(time.time())
61:
62:                 data = {}
63:
64:                 #
65:                 # announce string
66:                 #
67:                 localhost = self.db.getGlobalVar('Kickstart', 'PrivateAddress')
68:                 data['announce'] = 'http://%s:7625/announce' % (localhost)
69:
70:                 data['creation date'] = creation_date
71:
72:                 #
```

rocks create torrent

© 2008 UC Regents

rocks add host

```
73:                     basename, rack, rank = host.split('-')
74:                     self.db.execute("""select m.name from
75:                             appliances a, memberships m where
76:                             a.name="%s" and m.appliance=a.id""" % basename)
77:                     membership, = self.db.fetchone()
78:                     rack = int(rack)
79:                     rank = int(rank)
80:             except:
81:                     membership = None
82:                     rack = None
83:                     rank = None
84:
85:             # fillParams with the above default values
86:
87:             (membership, numCPUs, rack, rank) = self.fillParams(
88:                     [('membership', membership),
89:                     ('cpus', 1),
90:                     ('rack', rack),
91:                     ('rank', rank)])
92:
93:             if not membership:
94:                     self.abort('membership not specified')
95:             if rack == None:
96:                     self.abort('rack not specified')
97:             if rank == None:
98:                     self.abort('rank not specified')
99:
100:            self.db.execute("""insert into nodes
101:                    (site, name, membership, cpus, rack, rank)
102:                    values
103:                    (0,
104:                    '%s',
105:                    (select id from memberships where name='%s'),
106:                    '%d',
107:                    '%d',
108:                    '%d')""" %
109:                    (host, membership, int(numCPUs), int(rack), int(rank)))
110:
111:
```

© 2008 UC Regents

fillPositionalArgs

- Allows for parameters to have implied keys (just values on command line)
- This is an optimization for ease of use, not ease of software
- Argument is a list of keys
  - No default value processing, if a key is specified it is required
  - Use this only when a parameter is required
- Example:

```
# rocks set network netmask optiputer netmask=255.255.255.0
# rocks set network netmask optiputer 255.255.0.0
```

© 2008 UC Regents

rocks set network netmask

```
11:         </arg>
12:
13:         <arg type='string' name='netmask'>
14:         Netmask that named networks should have.
15:         </arg>
16:
17:         <param type='string' name='netmask'>
18:         Can be used in place of netmask argument.
19:         </param>
20:
21:         <example cmd='set network netmask optiputer 255.255.255.0'>
22:         Sets the netmask for the "optiputer" network to a class-c address
23:         space.
24:         </example>
25:
26:         <example cmd='set network netmask optiputer netmask=255.255.255.0'>
27:         Same as above.
28:         </example>
29:
30:         <example cmd='set network netmask optiputer cavewave 255.255.0.0'>
31:         Sets the netmask for the "optiputer" and "cavewave" networks to
32:         a class-b address space.
33:         </example>
34:
35:         <related>add network</related>
36:         <related>set network subnet</related>
37:         """
38:
39:         def run(self, params, args):
40:                 (args, netmask) = self.fillPositionalArgs(('netmask',))
41:
42:                 if not len(args):
43:                         self.abort('must supply network')
44:                 if not netmask:
45:                         self.abort('must supply netmask')
46:
47:                 for network in self.getNetworkNames(args):
48:                         self.db.execute("""update subnets set netmask='%s' where
49:                                 subnets.name='%s'""" % (netmask, network))
50:
```

© 2008 UC Regents

```
32:         Sets the MAC Address for the eth1 device on host compute-0-0.
33:         </example>
34:
35:         <example cmd='set host interface mac compute-0-0 iface=eth1 mac=00:0e:0c:a7:5d:ff'>
36:         Same as above.
37:         </example>
38:
39:         <example cmd='set host interface mac compute-0-0 iface=eth1 mac=NULL'>
40:         clears the mac address from the database
41:         </example>
42:
43:         <!-- cross refs do not exist yet
44:         <related>set host interface iface</related>
45:         <related>set host interface ip</related>
46:         <related>set host interface gateway</related>
47:         <related>set host interface module</related>
48:         -->
49:         <related>add host</related>
50:         """
51:
52:         def run(self, params, args):
53:
54:                 (args, iface, mac) = self.fillPositionalArgs(('iface', 'mac'))
55:
56:                 hosts = self.getHostnames(args)
57:
58:                 if len(hosts) != 1:
59:                         self.abort('must supply one host')
60:                 if not iface:
61:                         self.abort('must supply iface')
62:                 if not mac:
63:                         self.abort('must supply mac')
64:
65:                 for host in hosts:
66:                         self.db.execute("""update networks, nodes set
67:                                 networks.mac=NULLIF('%s','NULL') where
68:                                 nodes.name='%s' and networks.node=nodes.id and
69:                                 (networks.device='%s' or networks.mac='%s')""" %
70:                                 (mac, host, iface, iface))
71:
```

rocks set host interface

© 2008 UC Regents

Help and Docstrings

- The command line is the documentation
  - No more out of date man pages
  - Still needs a cookbook document, but reference is part of the code
- We've been looking at this all session
- Class docstring `"""text"""`
- Command line has an XML format

© 2008 UC Regents


```
# rocks list roll help
rocks list roll [roll]...

Description:

    List the status of available rolls.


Arguments:

    [roll]

    List of rolls. This should be the roll base name (e.g., base, hpc,
    kernel). If no rolls are listed, then status for all the rolls are
    listed.


Examples:

    $ rocks list roll kernel

    List the status of the kernel roll

    $ rocks list roll

    List the status of all the available rolls
```

© 2008 UC Regents

rocks list roll help

```
import os
import stat
import time
import sys
import string
import rocks.commands

class Command(rocks.commands.RollArgumentProcessor,
        rocks.commands.list.command):
        """
        List the status of available rolls.

        <arg optional='1' type='string' name='roll' repeat='1'>
        List of rolls. This should be the roll base name (e.g., base, hpc,
        kernel). If no rolls are listed, then status for all the rolls are
        listed.
        </arg>

        <example cmd='list roll kernel'>
        List the status of the kernel roll
        </example>

        <example cmd='list roll'>
        List the status of all the available rolls
        </example>
        """

        def run(self, params, args):

                self.beginOutput()
                for (roll, version) in self.getRollNames(args, params):
                        self.db.execute("""select version, arch, enabled from
                                rolls where name='%s' and version='%s'""" %
                                (roll, version))
                        for row in self.db.fetchall():
                                self.addOutput(roll, row)
                self.endOutput(header=['name', 'version', 'arch', 'enabled'],
                        trimOwner=0)

```

© 2008 UC Regents

<arg>

- Attributes
  - name (required)
  - optional (default = “0”)
  - type (default = “string”)
  - repeat (default = “0”)
- Example:

```
<arg type='string' name='network' repeat='1'>
  One or more named networks that should have the
  defined netmask.
</arg>
```

© 2008 UC Regents

<param>

- Attributes
  - name (required)
  - optional (default = “1”)
  - type (default = “string”)
  - repeat (default = “0”)
- Example:

```
<param type='string' name='iface'>
  Can be used in place of the iface argument.
</param>
```

© 2008 UC Regents

<example>

- Attributes
  - cmd(required)
- Example:

```
<example cmd='set host interface mac compute-0-0
  eth1 00:0e:0c:a7:5d:ff'>
  Sets the MAC Address for the eth1 device on host
  compute-0-0.
</example>
```

© 2008 UC Regents

<related>

- Example

```
<related>set host interface iface</related>
<related>set host interface ip</related>
<related>set host interface gateway</related>
<related>set host interface module</related>
```

© 2008 UC Regents

Help

- rocks <verb> <object…> <subject> help
  - Loads the command module
  - Parses the XML docstring
  - Format and output help as 80 column text
- Debug syntax with `format=` parameter

© 2008 UC Regents

help format=raw

```
# rocks list roll help format=raw
1:
2: List the status of available rolls.
3:
4: <arg optional='1' type='string' name='roll' repeat='1'>
5: List of rolls. This should be the roll base name (e.g., base, hpc,
6: kernel). If no rolls are listed, then status for all the rolls are
7: listed.
8: </arg>
9:
10: <example cmd='list roll kernel'>
11: List the status of the kernel roll
12: </example>
13:
14: <example cmd='list roll'>
15: List the status of all the available rolls
16: </example>
```

© 2008 UC Regents

Help format=parsed

```
# rocks list roll help format=parsed
{'related': [], 'example': [(u'list roll kernel', u'\t\t\n\tList the
    status of the kernel roll\n\t'), (u'list roll', u'\n\tList the status
    of all the available rolls\n\t')], 'description': u'\n\tList the status
    of available rolls.\n\t\n\t', 'param': [], 'arg': [((u'roll',
    u'string', 1, 1), u'\n\tList of rolls. This should be the roll base
    name (e.g., base, hpc,\n\tkernel). If no rolls are listed, then status
    for all the rolls are\n\tlisted.\n\t')]}
```

© 2008 UC Regents

Docbook

- Roll Usersguide Command Reference is generated automatically

```
# rocks list roll help format=docbook
<section id="rocks-list-roll" xreflabel="list roll">
<title>list roll</title>
<cmdsynopsis>
   <command>rocks list roll</command>
   <arg rep="repeat" choice="opt">roll</arg>
</cmdsynopsis>
<para>

   List the status of available rolls.


</para>
<variablelist><title>arguments</title>
   <varlistentry>
```

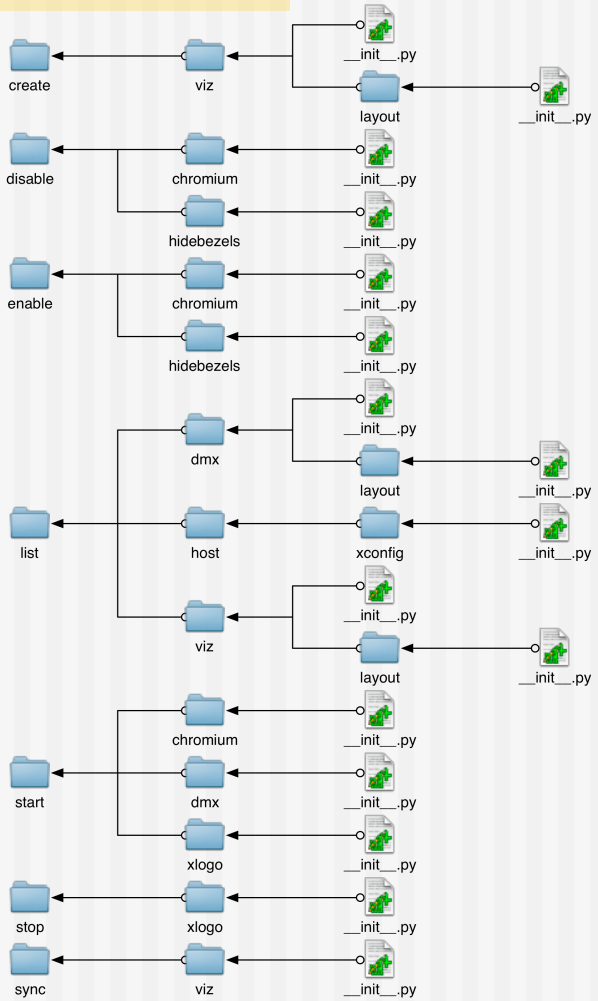
© 2008 UC Regents

Viz Roll

© 2008 UC Regents

Dozen+ Command

```
# rocks list roll command viz
COMMAND
create viz layout
disable chromium
disable hidebezels
enable chromium
enable hidebezels
list dmx layout
list host xconfig
list viz layout
start chromium
start dmx
start xlogo
stop xlogo
sync viz
```



© 2008 UC Regents

LCD Bezels

© 2008 UC Regents

rocks enable hidebezels

- Draws pixels behind the bezels (mullions) of the LCD monitors
- Calculated offset for TwinView and normal modes
- Reset the X11 configuration on all nodes
- Great mode for moving graphics

© 2008 UC Regents

```
import rocks.commands.enable
import os

class Command(rocks.commands.enable.command):
        """
        Enable Bezel Hiding mode.

        <example cmd="enable hidebezels">
        </example>
        """

        MustBeRoot = 0

        def run(self, params, args):

                os.system('touch ~/.hidebezels')

                # If the database videowall layout has two (or more) entries
                # for the same host and card we know we are in twinview
                # mode.  In this case we need to reconfigure and restart
                # X11 for this host.

                self.db.execute("""select n.name, v.cardid
                        from nodes n, videowall v where v.node=n.id""")
                dict     = {}
                for key in self.db.fetchall():
                        if dict.has_key(key):
                                dict[key] = 1   # TwinView host
                        else:
                                dict[key] = 0   # NonTwinView host (so far)

                for (host, card) in dict.keys():
                        if dict[(host, card)]:
                                os.system('ssh -f '
                                        '%s /usr/X11R6/bin/xrandr -d :0 -s 1'
                                        % host)
```

© 2008 UC Regents

rocks disable hidebezels

- All pixels are drawn and the bezels break apart the image
- Removes any offset from previous mode
- Resets the X11 configuration on all nodes
- Great for static images and text

© 2008 UC Regents

```
import rocks.commands.enable
import os

class Command(rocks.commands.disable.command):
        """
        Disable Bezel Hiding mode.

        <example cmd="disable hidebezels">
        </example>
        """

        MustBeRoot = 0

        def run(self, params, args):

                os.system('/bin/rm ~/.hidebezels')

                # If the database videowall layout has two (or more) entries
                # for the same host and card we know we are in twinview
                # mode.  In this case we need to reconfigure and restart
                # X11 for this host.

                self.db.execute("""select n.name, v.cardid
                        from nodes n, videowall v where v.node=n.id""")
                dict     = {}
                for key in self.db.fetchall():
                        if dict.has_key(key):
                                dict[key] = 1   # TwinView host
                        else:
                                dict[key] = 0   # NonTwinView host (so far)

                for (host, card) in dict.keys():
                        if dict[(host, card)]:
                                os.system('ssh -f '
                                        '%s /usr/X11R6/bin/xrandr -d :0 -s 0'
                                        % host)

```

© 2008 UC Regents

More Commands …

- Starting and stopping
  - Chromium
  - DMX
  - Xlogo image
- Re-writing X11 config files
- What do you want to see?
- HINT: This is interactive right now

© 2008 UC Regents

Roll Screen Development

Debugging assistance for building Rocks Rolls with screens

OSGC, May 2008

Nadya Williams nadya@oci.uzh.ch
University of Zurich

SDSC

Very Brief Rolls Overview

- Rolls provide for a cluster customization
- Rolls reliably install and configure a software on a cluster frontend
  - Extend/modify stock OS
  - Add third party packages
- Interaction during install only via screens
- Fully tested before release

Building Roll's Screen

Screen's Functions

- Collect user input before roll installation
- Verify user input and forward it to the rocks installer
- Depends on: what a roll developer puts in!

Screen's Pros and Cons

◆ Laconic form	⟷	◆ Limited space
◆ Provides help	⟷	• Don't assume user can type
◆ Verify input correctness	⟷	• Can't foresee all site details
◆ Easy and fast to use	⟷	• Screens are available only during cluster install

Prepare for the roll build

- Check out CVS Rocks distribution

  # cvs -d:pserver:anonymous@cvs.rocksclusters.org:/home/cvs/CVSROOT login

  # cvs -d:pserver:anonymous@cvs.rocksclusters.org:/home/cvs/CVSROOT checkout -r ROCKS_$VERSION
- Change to the top-level rolls' directory:

  # cd rocks/src/roll
- Create new roll directory

  # bin/make-roll-dir.py -n YourRollName -v YourRollVersion

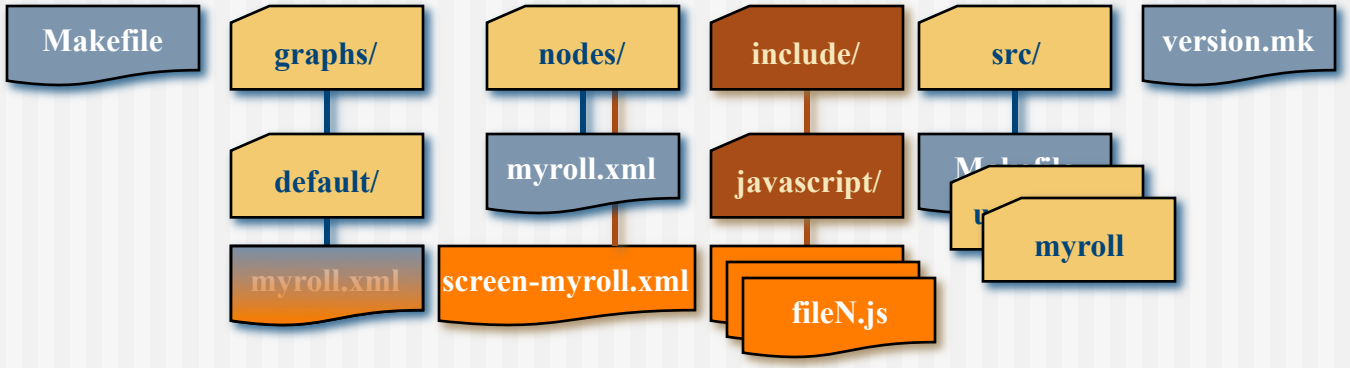
  # ls YourRollName

  graphs/ Makefile nodes/ src/ version.mk
- Create your roll

Add your Screen

- Prerequisites from rocks cvs repository
  - roll/bin/
  - roll/base/src/screens
  - Your basic roll structure is ready
    - Skeleton xml files, makefiles …

1. Add screen xml file to nodes/
2. Add javascript
3. Add screen to the graph



screen-myroll.xml file

```
<kickstart>
<screen>
        <title>Your Roll Title </title>
        <code>
           <include file="javascript/yourRoll.js">
        </code>
        <variable>
                <label>My field label</label>
                <name>Info_NameForAppsGlobals</name>
                <type>string</type>
                <size>20</size>
                <default>ReasonableDeafaultValue</default>
                <value><var name="Info_NameForAppsGlobals"/></value>
                <help>Help text for the variable field.</help>
                <validate>check_function1</validate>
        </variable>
        <variable>
        …
        </variable>
</screen>
</kickstart>
```

Screen XML Language

Screen contains

- Title
  - Your title description
- code
  - Specify javascript file
- variable
  - Simple syntax. Supports multiple variables.
  - Defines info about variable “appearance” on the screen
  - Defines info that goes to *app_globals* table

Screen <variable> attributes

- name
  - Format: *Service_Component*
  - *Service* and *Component* are columns names in the *app_globals* database table.
  - Use “Info” for Service:

```
<name>Info_GfarmMetaServer</name>
```

- value
  - Format same as “name”

```
<value>
<var name="Info_GfarmMetaServer"/>
</value>
```

- default
  - Sets the value of this variable

```
<default>pine.forrest.edu</default>
```

- label
  - Sets the form label
- type
  - String, menu, ipv4-address
- size
  - Sets the size of the screen field
- help
  - Sets the explanation for the variable
- validate
  - Sets the javascript validation function
  - Use the same name as in myroll.js javascript file

```
<validate>check_GFmetaserver</validate>
```

Add screen to the graph

In graphs/default/myroll.xml:

- add ordering

```
<order head="screen-timezone">
        <tail>screen-myroll</tail>
</order>
<order head="screen-myroll">
        <tail>screen-partitioning</tail>
</order>
```

- add edges

```
<edge from="server">
        <to>screen-myroll</to>
        <to>myroll-server</to>
        <to>myroll-client</to>
</edge>
```

File myroll.js

- Contains screen variable verification
- One function per variable
- Can use existing code from other rolls
  - Copy desired file from *otherroll/include/javascript/* to *myroll/include/javascript/*
- For examples see rolls:
  - base
  - gama

Debugging Roll's Screen

Screen html files

- Create screen html files:

```
# export PATH=$PATH:`pwd`/rocks/src/roll/bin
# cd myroll/
# make-screen-val.py   -x  screen-myroll.xml    myroll
```

Make-screen-val.py actions:

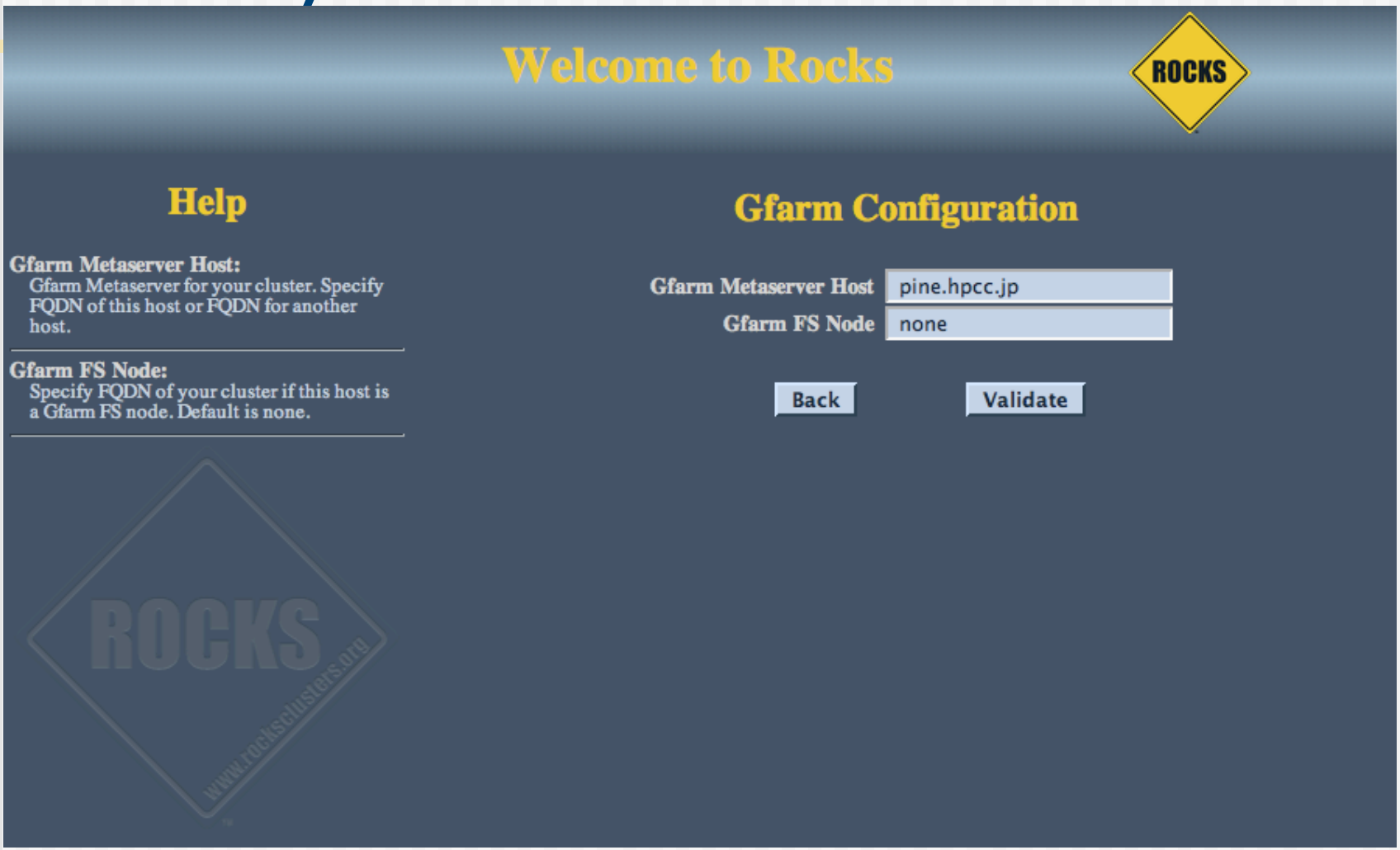
- Uses your javascript
- Creates myroll/screenval/ and all files under it

- View html files:

```
# cd screenval/
# firefox  file:///<path>/myroll/screenval/rocks.html
```

Note: will not work with Safari

View your Screen



Try your Screen

What can you test

- Test input fields one by one
  - alter the default value and press “validate”
- Test your javascript
  - If don’t see expected behavior, check your javascript
- Test, test, test…

```
while ( errors ) {
    recreate screenval/ with make-screen-val.py
    reopen rocks.html
    fix another error in your javascript
}
```

Lets see this in action

When Things Go Wrong

- Problem with javascript syntax
- One of javascript files is absent
- Names mismatch
- Function is not returning value

Thank you!

Questions ?