



# **StarBED and SpringOS: Large-scale General Purpose Network Testbed and Supporting Software**

Toshiyuki MIYACHI, Ken-ichi Chinen and Yoichi Shinoda

toshi-m@jaist.ac.jp.

Japan Advanced Institute of Science and Technology

# Motivations and Goals

---

- Evaluating actual implementations on a realistic environment
  - In order to evaluate actual implementations, we should build a testbed based on actual nodes
  - Developers often build a testbed using 10-20 actual nodes
  - However, the gap between these small testbed and the live network is often quite huge regarding scale and complexity
- Therefore, we proposed and implemented StarBED and SpringOS
  - StarBED: a facility which has many actual nodes for experiments
  - SpringOS: supporting software for experiments on StarBED

# StarBED

- about 70 racks
- 680 actual PCs
- 7 switches for experiments



You can see more pictures at the StarBED Project web page

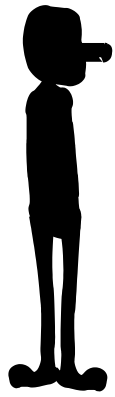
<http://www.starbed.org/>

# StarBED

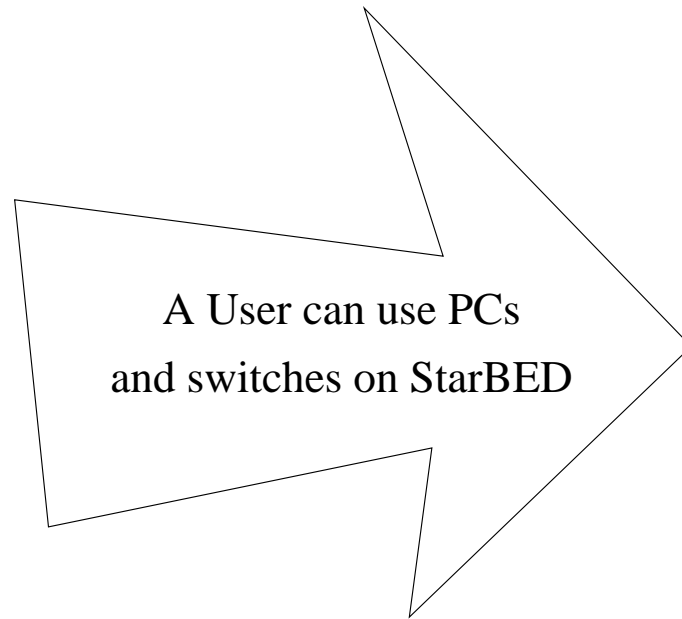
---

- Management network is separated from experiment network, in order to prevent management traffic influencing experiments
- Network Interfaces on experiment network are connected to intelligent switches and we make experiment topologies by configuring VLAN or ATM
- Users can change OS and install any applications to assigned nodes
- User plug-in lacks to connect user's nodes to StarBED
- Controlling each node's console from one terminal
- 2 \* 10Gbps links to the Internet(WIDE and JGN2)
- VPN accessibility

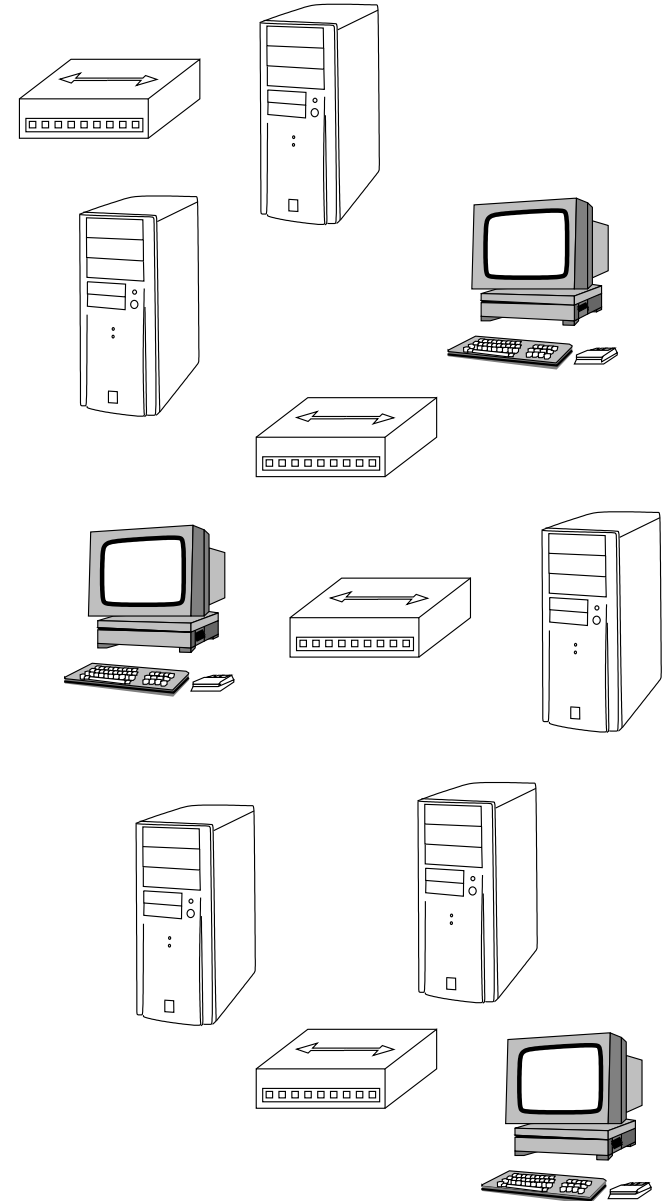
# Role of SpringOS



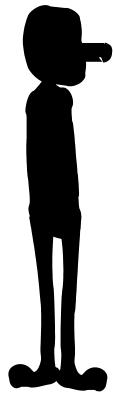
User



StarBED



# Role of SpringOS

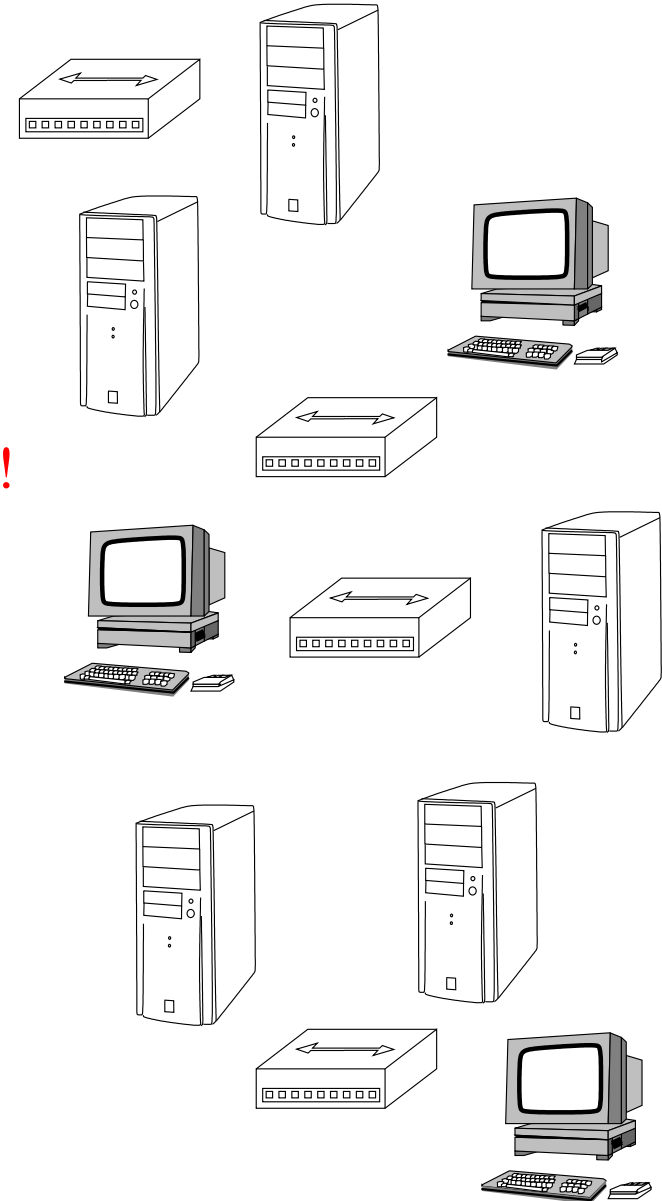


User

Controlling many nodes is difficult!

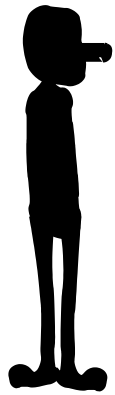
A User can use PCs  
and switches on StarBED

StarBED

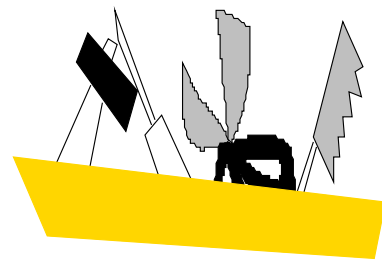


# Role of SpringOS

---

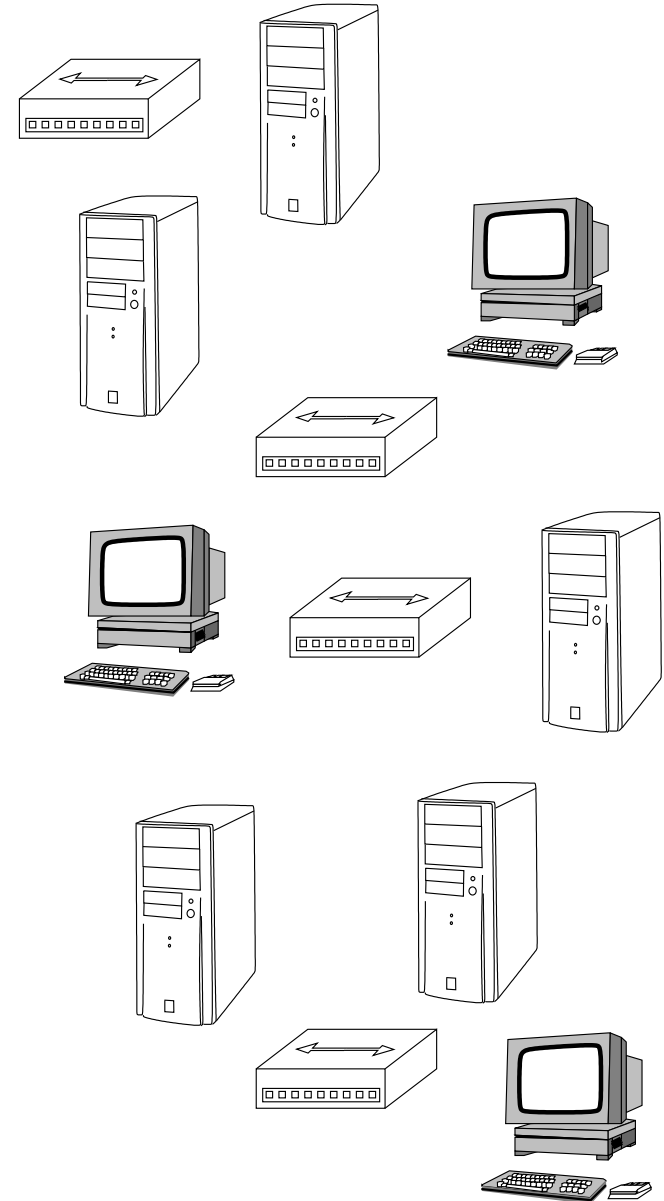


User

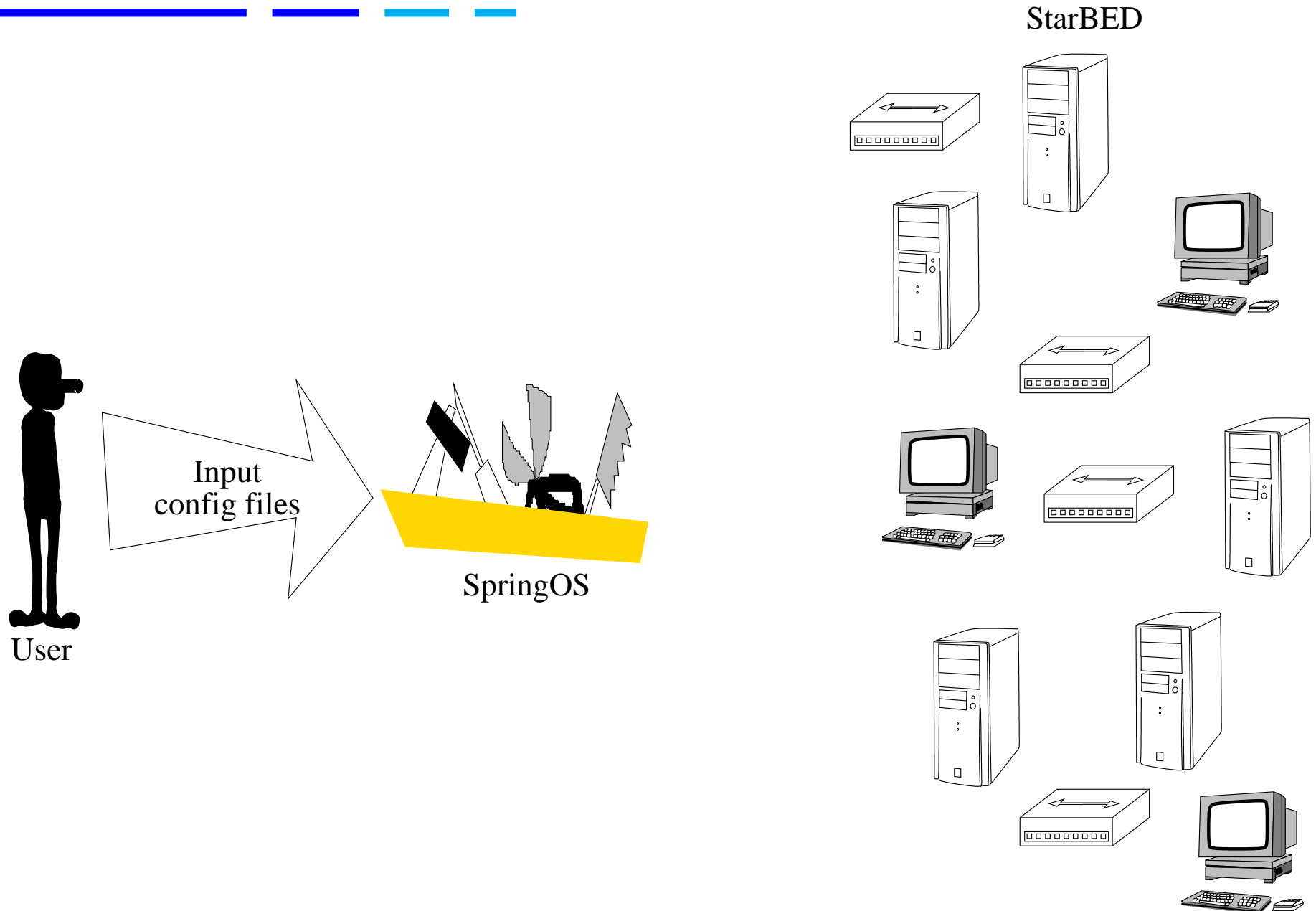


SpringOS

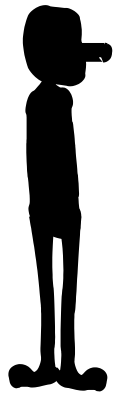
StarBED



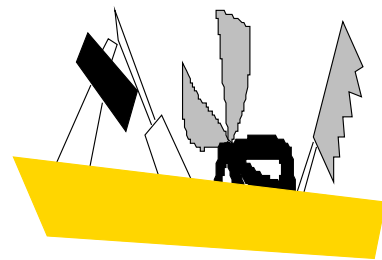
# Role of SpringOS



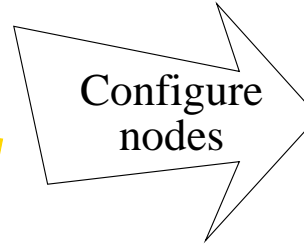
# Role of SpringOS



User

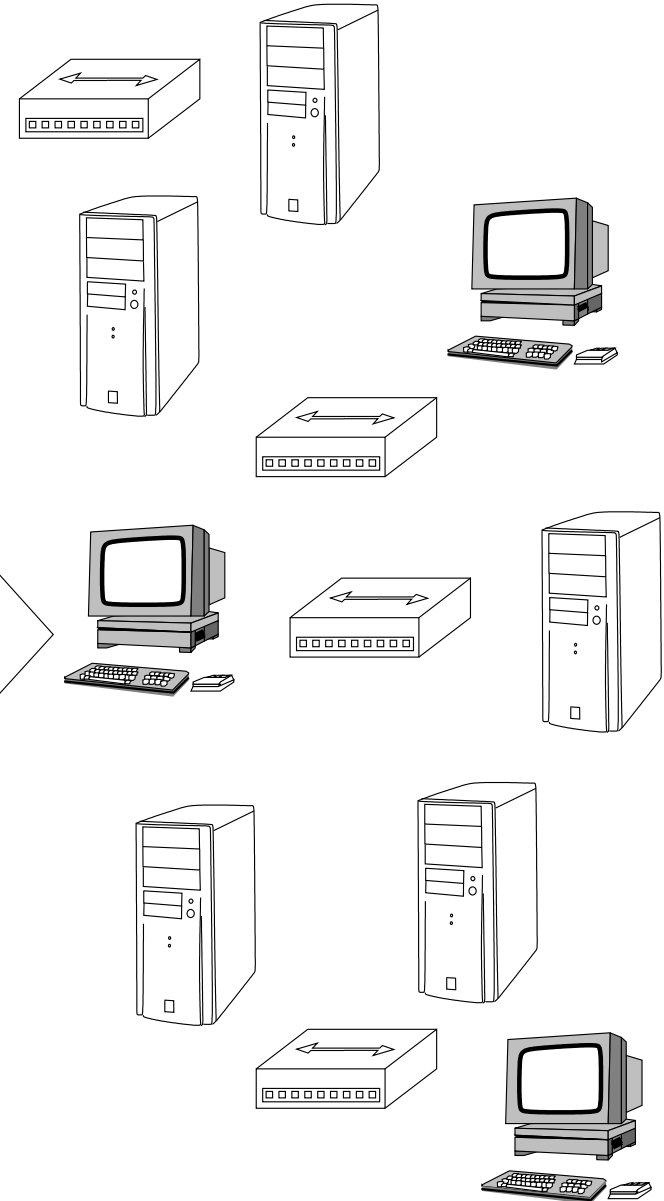


SpringOS



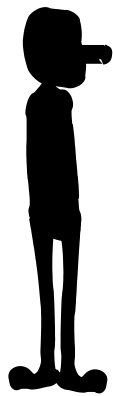
Configure nodes

StarBED

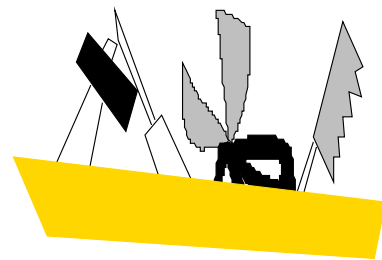




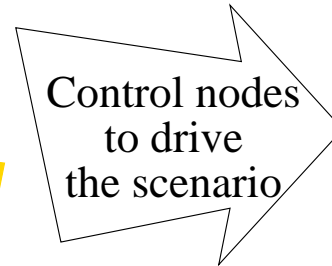
# Role of SpringOS



User

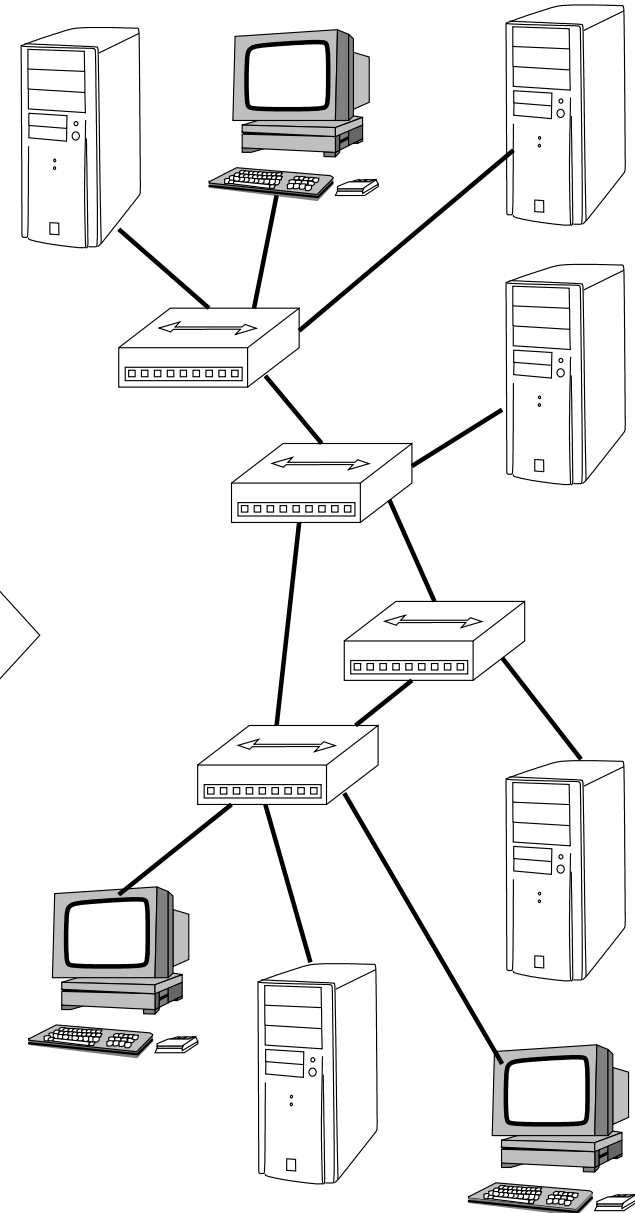


SpringOS



Control nodes  
to drive  
the scenario

StarBED



# Major Functions of SpringOS

---

- Resource management and assignment
  - User mediation
  - Find resources which satisfy user's request
- Node power management
- Software installation to nodes
- Building topology for experiments
- Driving Scenario
  - Execute commands on nodes
  - Node synchronization using message passing

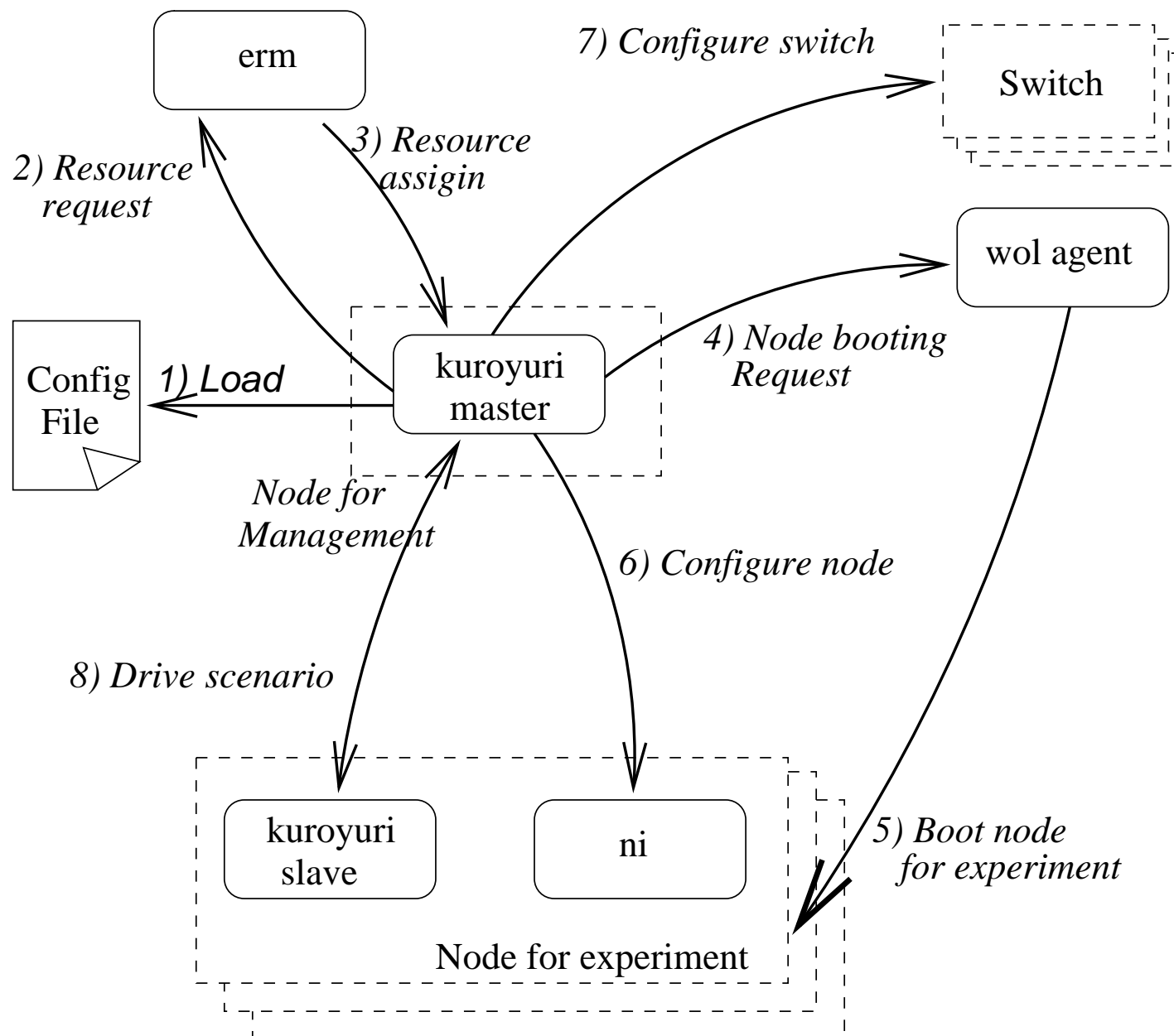
# Major Modules of SpringOS

---

SpringOS is not an application name, it is an application suite

- Kuroyuri master
  - Loads configuration files and then it conducts the other modules to make experiments
  - Manages node synchronization during driving scenarios
- Kuroyuri slave
  - Executes commands on nodes for experiments
- Experiment resource manager(erm)
- Node initiator(ni)
  - Introduces software for the experiment to nodes
- Wake on LAN Agent(wol agent)

# Experiment Control Steps by SpringOS



# Driving Scenario

---

- When kuroyuri slave boots up, kuroyuri master send the scenario for it
- kuroyuri slave begins to drive the scenarios when it receives the scenario
- If kuroyuri slave should synchronize with another node, it communicate another node via kuroyuri master
- kuroyuri master conducts kuroyuri slaves for synchronization by message passing

# Node Definition

---

- We adopt the class concept to define nodes and networks

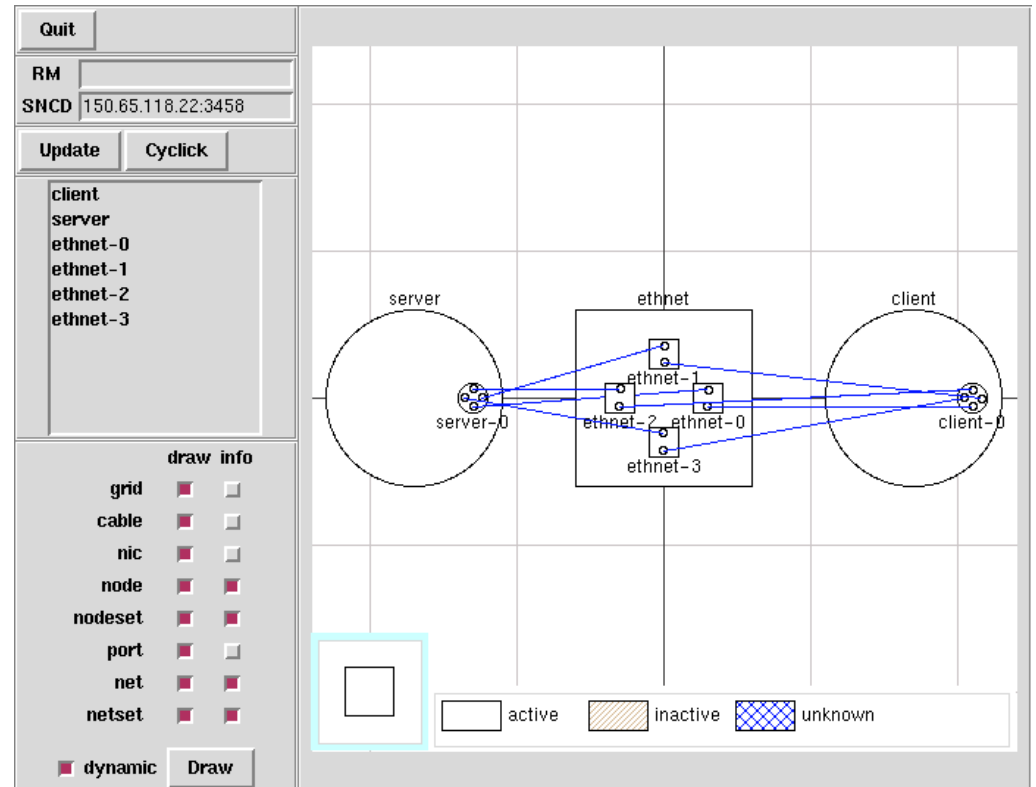
```
nodeclass svclass {
    partition 2
    method HDD
    ostype "FreeBSD"
    diskimage "ftp://anonymous:password@172.16.1.1/s_image.gz"
    netif media fastethernet num 4
    scenario {
        wake "/sim/netserver" "/sim/netserver"
        send "serverstarted"
        recv msg
        msgswitch msg {
            "quit" {
                wakewait "/usr/bin/killall" "killall" "netserver"
                exit
            }
        }
    }
}
```

# Topology Definition

```
nodeset client class clclass num 1
nodeset server class svclass num 1
netset ethnet class ethclass num 4

attach server.netif["lan0"] ethnet[0]
attach server.netif["lan1"] ethnet[1]
attach server.netif["lan2"] ethnet[2]
attach server.netif["lan3"] ethnet[3]

attach client.netif["lan0"] ethnet[0]
attach client.netif["lan1"] ethnet[1]
attach client.netif["lan2"] ethnet[2]
attach client.netif["lan3"] ethnet[3]
```



# Scenario Definition

```
nodeclass svclass {
    ...
    scenario {
        wake "/sim/netserver" "/sim/netserver"
        send "serverstarted"
        recv msg
        msgswitch msg {
            "quit" {
                wakewait "/usr/bin/killall" \\  
                    "killall" "netserver"
                exit
            }
        }
    }
    ...
    scenario {
        sync {
            msgmatch server[0] "serverstarted"
            msgmatch client[0] "clisetupdone"
        }
        send client[0] haddr(server[0].netif[0].ipaddr)
        sync {
            msgmatch client[0] "cdone"
        }
        send server[0] "quit"
        exit
    }
}
```

➤ Node scenario is driven by kuroyuri slave

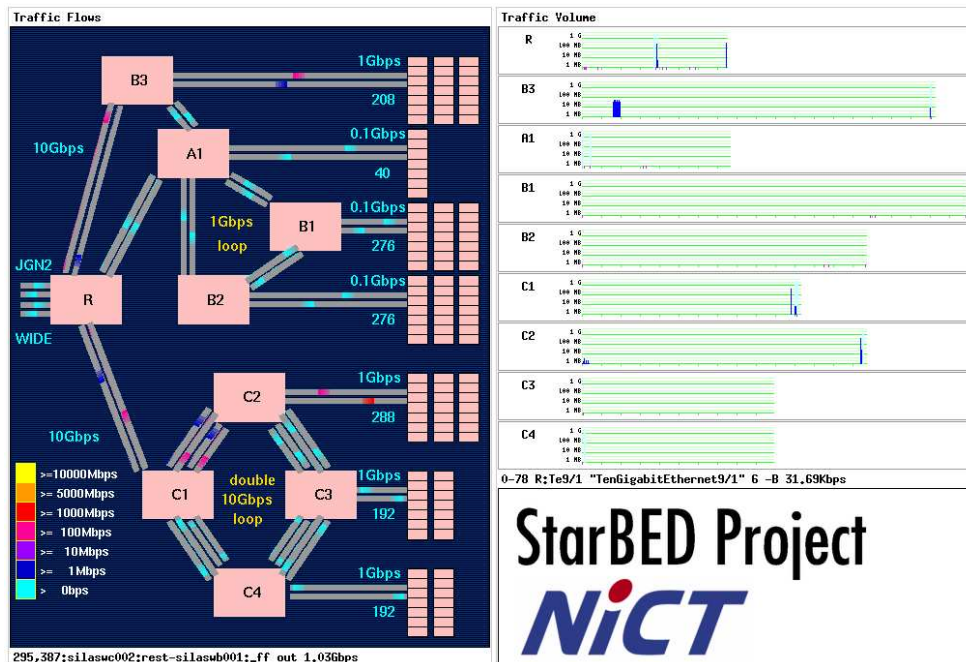
➤ In principle, node scenario is command list which should be executed on the nodes

➤ Global scenario is driven by kuroyuri master

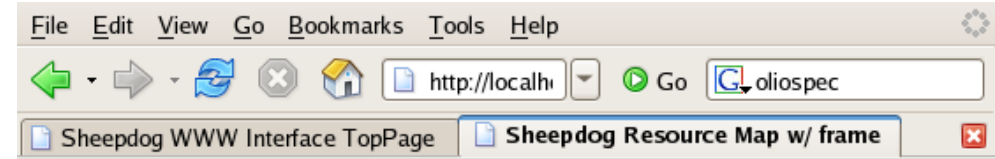
➤ Manage the message for synchronizing node scenario

# Other Modules of SpringOS

## Traffic Monitor

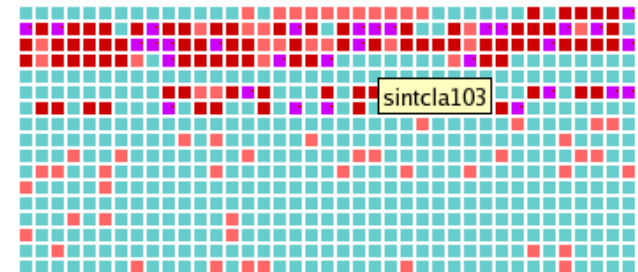


## Node State Monitor



## Sheepdog Resource Map [Top](#)

◆ Init, ■ Caos, ■ Fine, ■ MaybeFine, ■ MaybeDown, ■ Down, ■ Unknown.



Operation	OS	name
PowerON	AS IS	<input type="text"/>
<input type="button" value="SUBMIT"/>		

■ Fine sintclb052 172.16.1.52 ck 13345/8002311 IPMI 0 0(0) ICMP 706 12639(0)  
 SSH 705 12640(0) #  
 http://localhost:2468/usual/resmapnote.html?sintcla103

# Experiments on StarBED

---

- Evaluation of implementation of hierarchical IP traceback on Top-AS100 topology
- Name service performance comparison using P2P with using DNS
- L2 Switch benchmarks by multicast traffic using 200 nodes
- Investigation influence of new technology to ISP network using about 1050 nodes(using VMware)
  - The core nodes are actual nodes and surrounding nodes are virtual nodes
  - It is our record regarding scale of topology
- Evaluation of integrated middleware for overlay networking
- Evaluation of implementation of control method for network game using P2P
- etc.

# Conclusion and Future works

---

- We implemented StarBED and SpringOS
  - Users can evaluate their implementations on large-scale, realistic and flexible environment
- Now we are designing StarBED2: Large-scale, Realistic and Real-time Testbed for Ubiquitous Networks
  - Enhance StarBED for ubiquitous and sensor networks
  - The goal is emulation of town network
  - This project is just beginning phase

# Thank you

---

- StarBED Project
  - <http://www.starbed.org/>
  - [info@starbed.org](mailto:info@starbed.org)