

GENI Experiment Control Using Gush

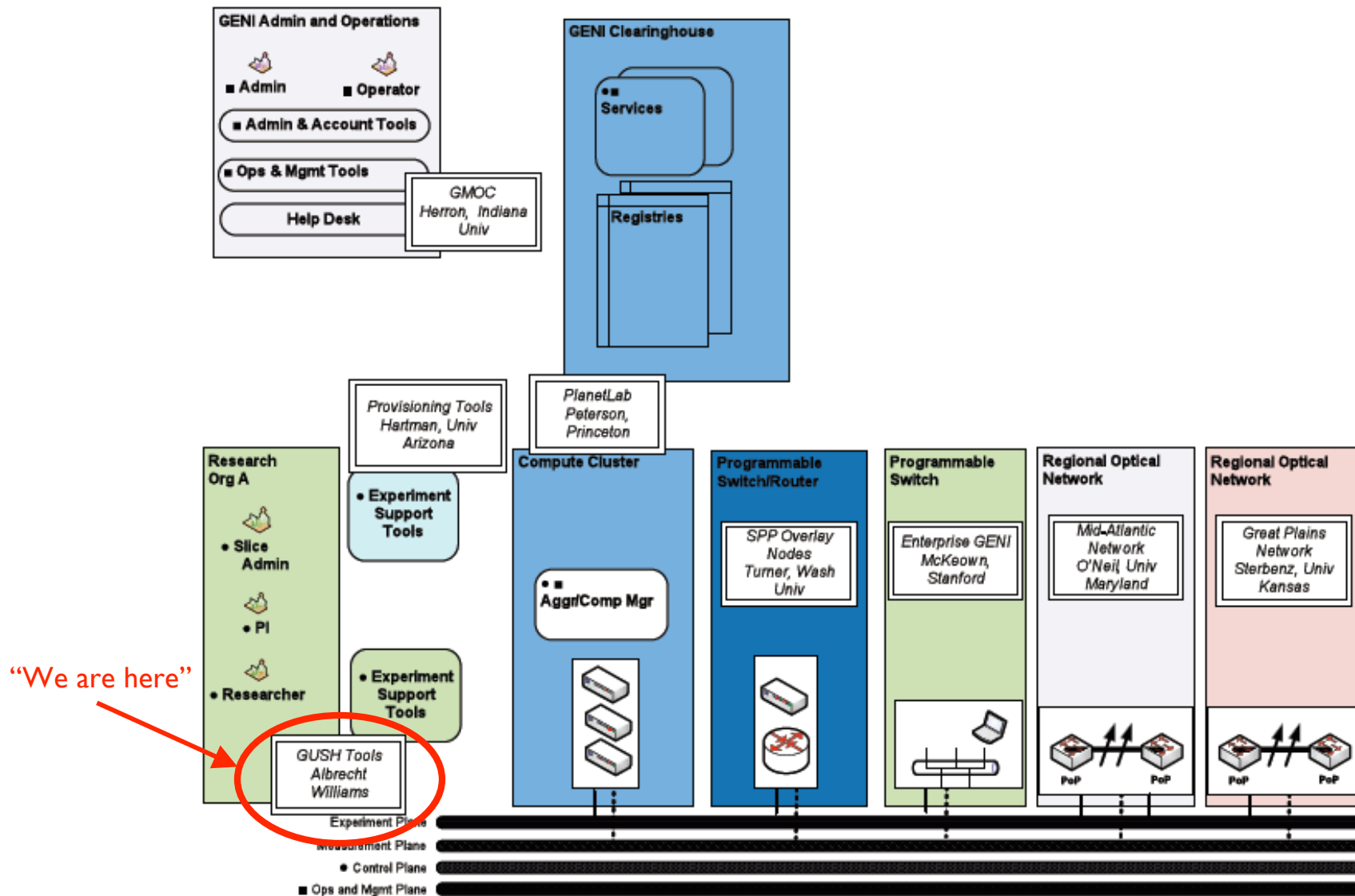
Jeannie Albrecht and Amin Vahdat
Williams College and UC San Diego



Overview

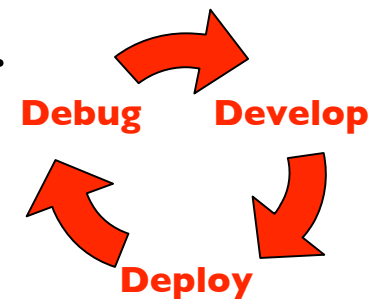
- How do we actually use GENI?
- Goal: Develop abstractions and tools for addressing the challenges of managing distributed applications
 - Make it easy for a range of users to run a variety of experiments on GENI
- Talk outline
 - Summarize existing work and design of **Gush**
 - Describe implementation/integration plan and timeline
 - Stimulate a broader discussion about the general requirements of an “experiment controller” in GENI

Cluster B: PlanetLab CF



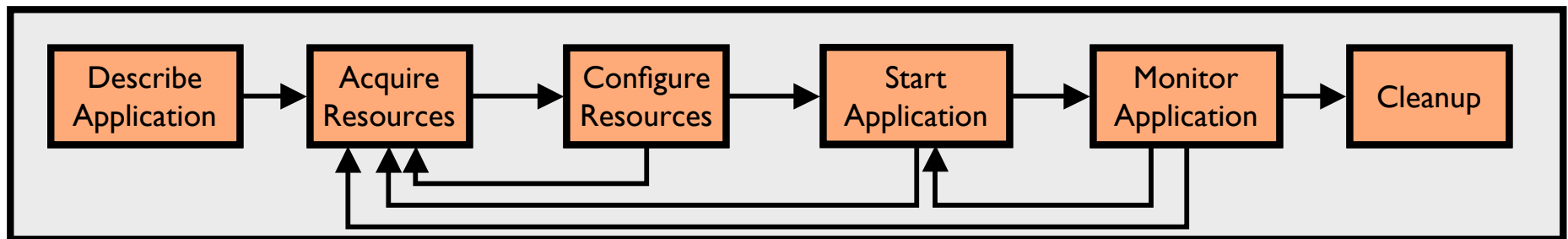
Running an Experiment

- Suppose we have written our software and are ready to deploy on PlanetLab for the first time
- We could...
 1. Connect to each of the 800+ PlanetLab machines
 2. Download software (no common file system)
 3. Install software
 4. Run application and analyze performance
 5. Check for errors on each machine
 6. When we find an error, we start all over...
- Or we could use **Gush**



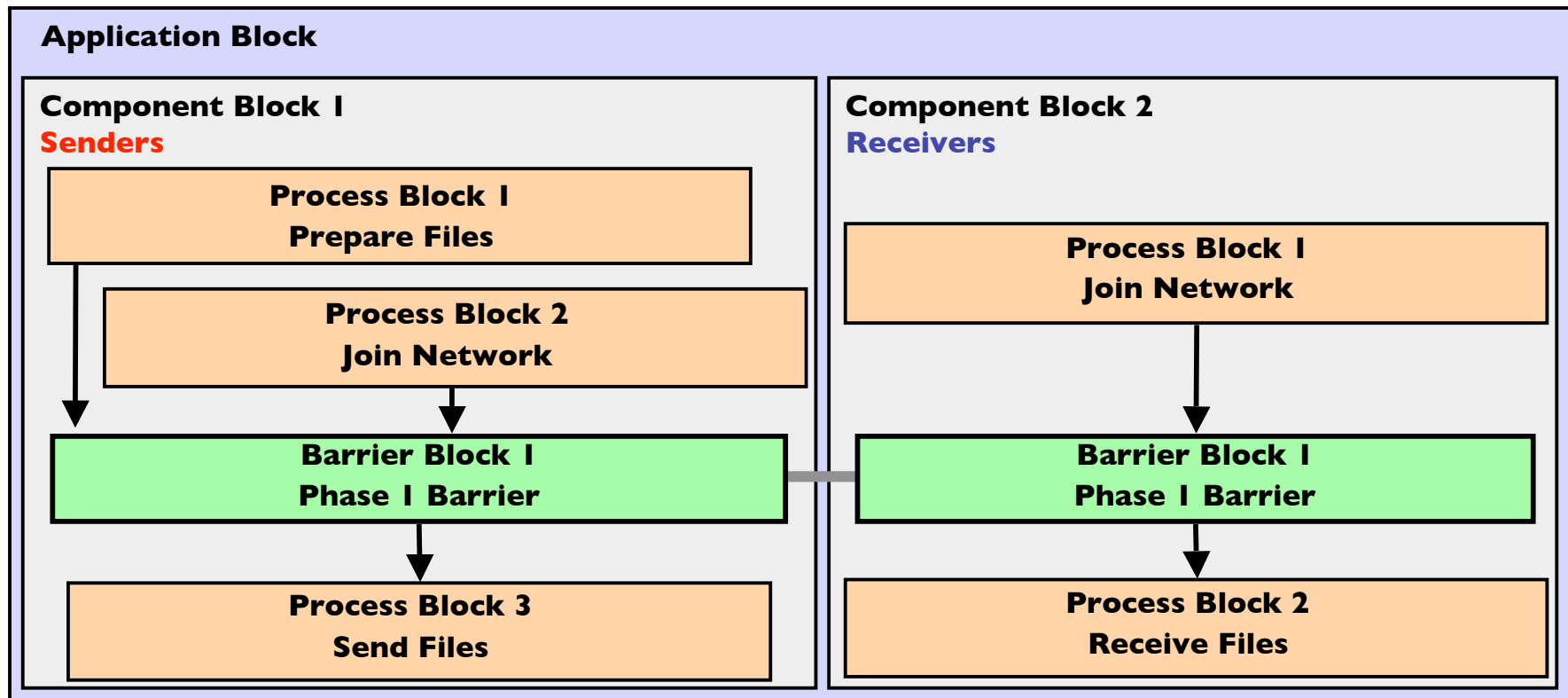
Gush

- A distributed application management infrastructure
 - Extends functionality of **Plush** to support experiment control on GENI
 - Designed to simplify deployment of distributed applications
 - Provides abstractions for configuration and management
 - Allows users to “remotely control” computers running distributed applications worldwide

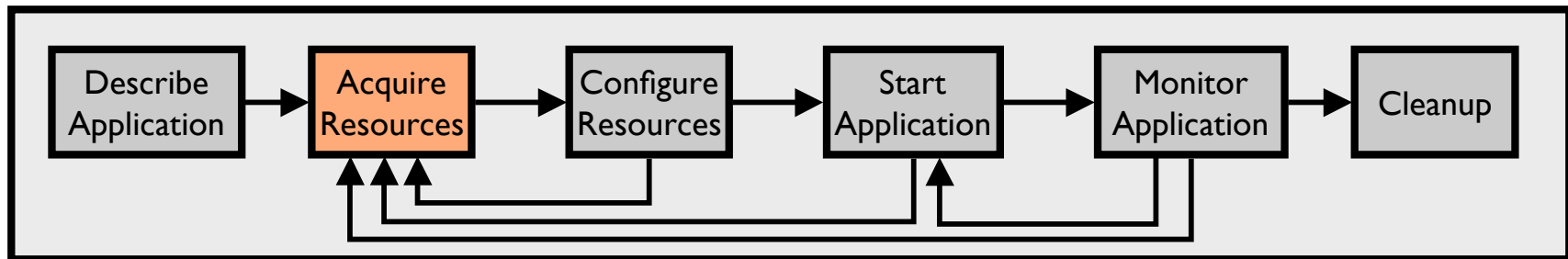


Step I: Describe Application

- Describe experiment using application “building blocks”
- Create customized control flow for distributed applications
- **Application specification** blocks are described using XML

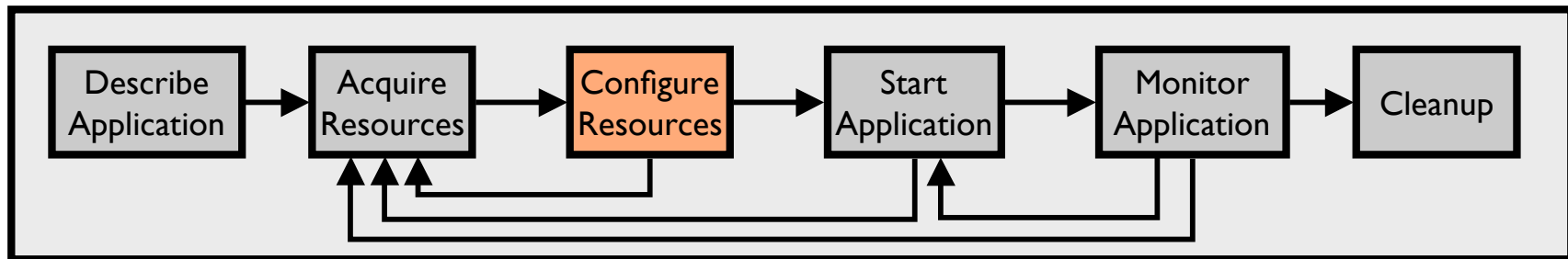


Step 2: Acquire Resources

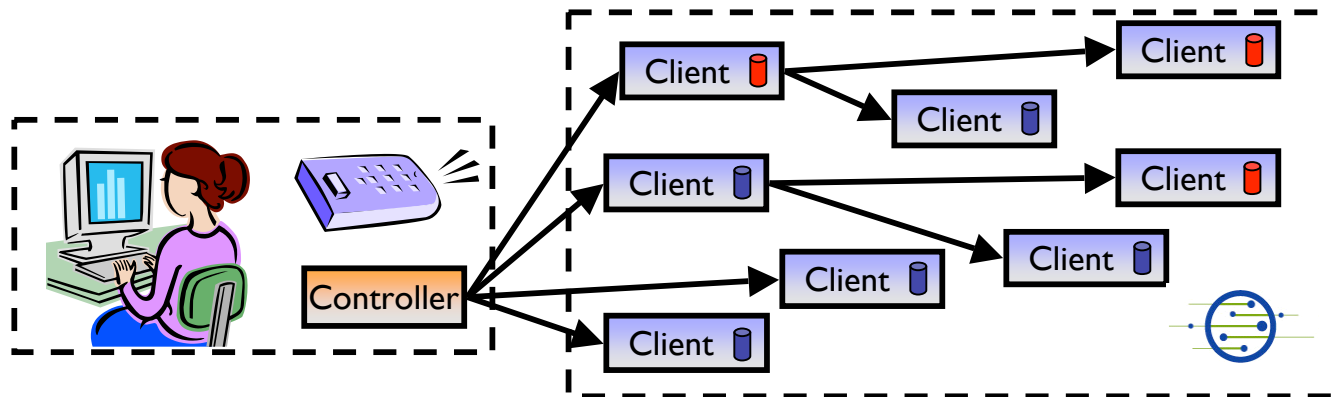


- How can we find “good” machines?
 - We want machines with specific characteristics
 - PlanetLab services perform *resource discovery* to find machines that satisfy our requirements
- Gush interfaces directly with these services
 - Eventually Gush will talk directly to GENI Clearinghouses to find resources

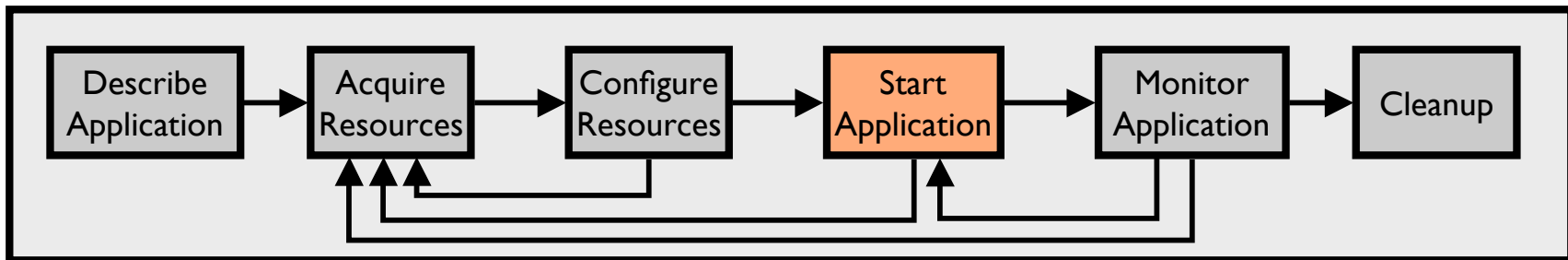
Step 3: Configure Resources



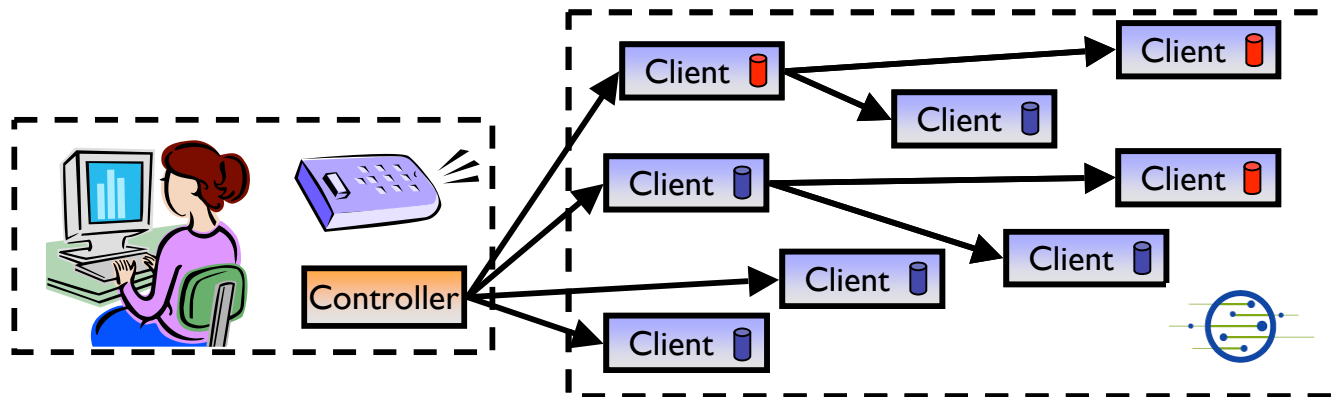
- Connect to and configure selected resources
 - Optionally create a tree for achieving scalability in communication
 - **Controller** “remotely controls” the **clients** on our behalf
 - Install software on clients (some are **senders**, some are **receivers**)



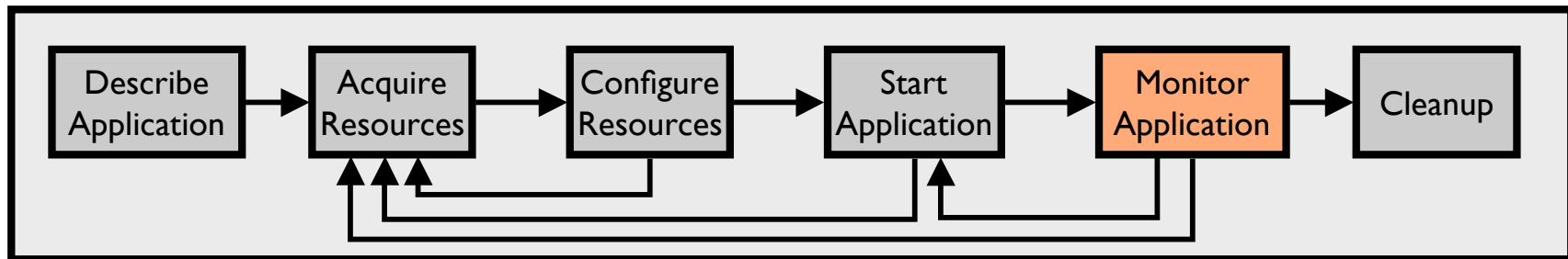
Step 4: Start Application



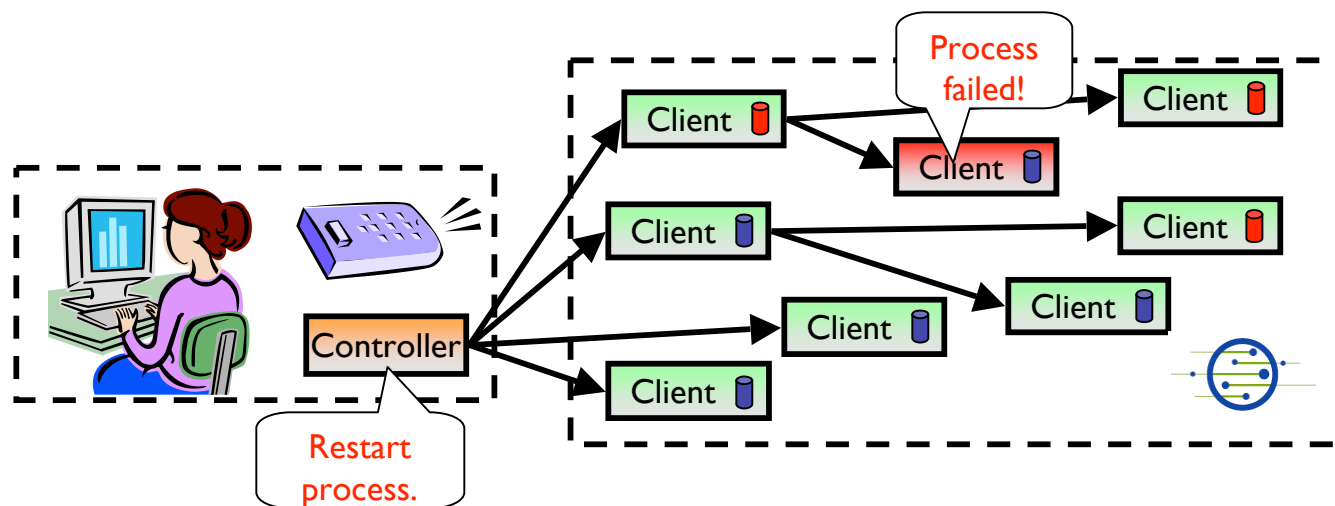
- Controller issues commands to clients telling them to start running our application
 - **Senders** begin running sender processes
 - **Receivers** begin running receiver processes



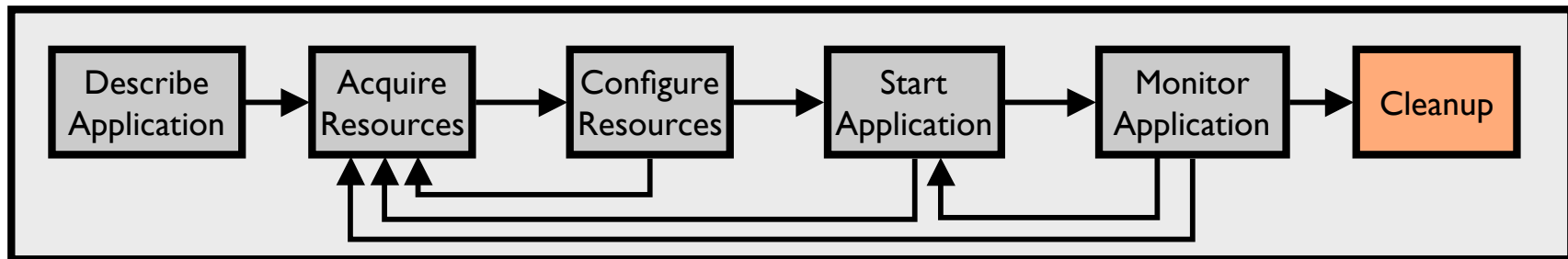
Step 5: Monitor Application



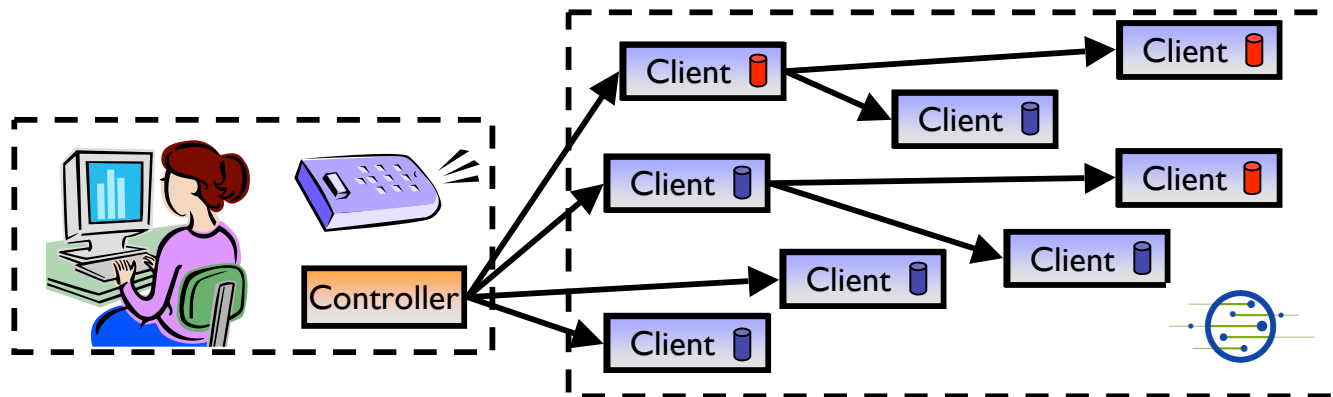
- We want to make sure the processes keep running
- Gush clients monitor experiment processes for failures
 - If a failure is detected, client notifies controller
 - Controller decides to tell client to restart failed program or process



Step 6: Cleanup

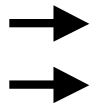


- Gush clients make sure all programs exited cleanly
- Remove logs and software from remote machines
- Disconnect clients from controller



Gush User Interfaces

- Command-line interface used to interact with applications
 - Provides single point of control for remotely controlling resources
- Nebula (GUI) allows users to describe, run, monitor, & visualize applications
- XML-RPC interface for managing applications programmatically



The screenshot shows the Nebula v0.8 GUI with a terminal window titled 'SSH:planetlab1.cs.duke.edu'. The terminal displays the following content:

```
logfile-planetlab1-15415-1178479282.txt logfile-planetlab1-15415-1178664027.txt logfile-planetlab1-15417-1178514401.txt
logfile-planetlab1-15415-1178484137.txt logfile-planetlab1-15415-1178664362.txt
logfile-planetlab1-15415-1178484906.txt logfile-planetlab1-15415-1178664430.txt
[ucsd_plush@planetlab1 ~]$ less logfile-planetlab1-1541
[ucsd_plush@planetlab1 ~]$ ls -ltr
total 6732
-rwxr--r-- 1 ucsd_plush slices 241 Apr 24 17:54 plush.prefs
drwxr--r-- 3 ucsd_plush slices 4096 May 6 03:17 helper-scripts
-rwxr--r-- 1 ucsd_plush slices 6458700 May 6 19:09 client
-rw-r--r-- 1 ucsd_plush slices 293 May 6 19:21 plush-logfile15415-1178479282.txt
-rw-r--r-- 1 ucsd_plush slices 27361 May 6 19:28 logfile-planetlab1-15415-1178479282.txt
-rwxr--r-- 1 ucsd_plush slices 4764 May 6 20:41 bootstrap.pl
-rw-r--r-- 1 ucsd_plush slices 291 May 6 20:42 plush-logfile15415-1178484137.txt
-rw-r--r-- 1 ucsd_plush slices 39787 May 6 20:43 logfile-planetlab1-15415-1178484137.txt
-rw-r--r-- 1 ucsd_plush slices 293 May 6 20:55 plush-logfile15415-1178484906.txt
-rw-r--r-- 1 ucsd_plush slices 37634 May 6 20:57 logfile-planetlab1-15415-1178484906.txt
-rw-r--r-- 1 ucsd_plush slices 280 May 7 05:06 plush-logfile15417-1178514401.txt
-rw-r--r-- 1 ucsd_plush slices 18694 May 7 05:08 logfile-planetlab1-15417-1178514401.txt
-rw-r--r-- 1 ucsd_plush slices 311 May 8 22:40 plush-logfile15415-1178664027.txt
-rw-r--r-- 1 ucsd_plush slices 32749 May 8 22:44 logfile-planetlab1-15415-1178664027.txt
-rw-r--r-- 1 ucsd_plush slices 313 May 8 22:46 plush-logfile15415-1178664362.txt
-rw-r--r-- 1 ucsd_plush slices 32923 May 8 22:46 logfile-planetlab1-15415-1178664362.txt
lrwxrwxrwx 1 ucsd_plush slices 35 May 8 22:47 plush-logfile.txt -> ./plush-logfile15415-1178664430.txt
lrwxrwxrwx 1 ucsd_plush slices 41 May 8 22:47 client.txt -> ./logfile-planetlab1-15415-1178664430.txt
-rw-r--r-- 1 ucsd_plush slices 313 May 8 22:47 plush-logfile15415-1178664430.txt
-rw-r--r-- 1 ucsd_plush slices 168123 May 8 22:48 logfile-planetlab1-15415-1178664430.txt
[ucsd_plush@planetlab1 ~]$ traceroute www.google.com
traceroute: Warning: www.google.com has multiple addresses; using 72.14.205.99
traceroute to www.l.google.com (72.14.205.99), 30 hops max, 38 byte packets
 1 152.3.138.61 (152.3.138.61) 0.330 ms 0.275 ms 0.229 ms
 2 152.3.219.69 (152.3.219.69) 0.353 ms 0.300 ms 0.230 ms
 3 tellsp-roti.netcom.duke.edu (152.3.219.54) 0.281 ms 0.333 ms 0.245 ms
 4 te2-1--581.tr01-asbnva01.transitrail.net (137.164.131.173) 7.633 ms 7.663 ms 8.402 ms
 5 tel-2.tr01-sttlwa01.transittrail.net (137.164.129.37) 76.141 ms 84.463 ms 76.121 ms
 6 te4-1--160.tr01-plalca01.transittrail.net (137.164.129.34) 93.630 ms 93.511 ms 93.597 ms
 7 calren-trcust.plalca01.transittrail.net (137.164.131.254) 99.644 ms 97.167 ms 93.723 ms
 8 * * *
 9 209.85.130.4 (209.85.130.4) 95.293 ms 97.987 ms 94.702 ms
10 64.233.174.81 (64.233.174.81) 86.525 ms 86.340 ms 86.495 ms
   MPLS Label=684000 CoS=0 TTL=1 S=1
11 72.14.236.20 (72.14.236.20) 93.077 ms 110.785 ms 93.037 ms
12 72.14.232.113 (72.14.232.113) 100.908 ms 96.452 ms 98.807 ms
13 72.14.232.62 (72.14.232.62) 99.173 ms 72.14.236.142 (72.14.236.142) 95.319 ms 72.14.232.66 (72.14.232.66) 100.434 ms
14 qb-in-f99.google.com (72.14.205.99) 95.983 ms 93.976 ms 107.922 ms
[ucsd_plush@planetlab1 ~]$
```

Current Status

- Gush works with current PlanetLab API
 - Will continue to modify Gush as GENI APIs are implemented and released
- Preliminary user study completed
- Student involvement
 - Undergraduate research projects at Williams College
 - Gush web interface (GoogleMaps)
 - Batch scheduler that uses Gush XML-RPC interface for “automated” experiment control on PlanetLab
 - Gush in the classroom
 - “Bringing Big Systems to Small Schools: Distributed Systems for Undergraduates,” To appear in SIGCSE 2009.

Summary

- Gush provides abstractions for managing a range of distributed applications on GENI
 - Provides three different user interfaces to meet the needs of a variety of researchers with varying levels of expertise
- Next steps (6-12 months)
 - Continue obtaining user feedback to enhance usability and provide additional functionality
 - Develop user documentation
 - Integrate with other Cluster B GENI services (e.g., Stork/Raven)
 - Demo different user interfaces at future GECs
- Long term goals
 - Build on prior Plush/Orca integration to support experiments on sensor/mobile networks (e.g., DieselNet)

Thanks!

For more info, visit

<http://gush.cs.williams.edu>

Email:

jeannie@cs.williams.edu

Discussion Questions

- What experiment control functionality do researchers want/need on GENI?
 - Support for both novice and advanced users
- How much control should be exposed?
- How do I specify an experiment?
- What is the API between experiment controllers and the underlying control frameworks?
- What is the API between the experiment controller and GENI Clearinghouses?