Supporting Experiment Workflows in GENI

GENI Experiment Workflow and Services Working Group

Jeannie Albrecht
Williams College

Revisiting Experiment Workflow

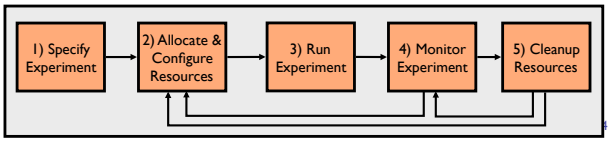
- One of the key purposes of this working group is to focus on "Experiment Workflow"
- How can we make it "easy" for a researcher or user to run an experiment on GENI?
- Solutions must address tasks associated with slice management, resource discovery and configuration, and experiment control (some potential overlap with other WGs)

PlanetLab Experience

- PlanetLab taught us many things about experiment control and maintenance
  - For novice researchers, it's often not "easy"
- Several challenges must be overcome in wide-area distributed environments:
  - Acquiring and configuring "optimal" resources
  - Managing a long-running experiment or service
  - Detecting and recovering from failures
  - Monitoring experiments in real-time
- Many researchers resort to brittle, custom-built, application-specific scripts to accomplish these tasks

Things to Consider

- Experiment control requirements:
  - Extensible experiment specifications
  - Resource discovery and slice configuration
  - Support for potentially multi-phased execution
  - Support for experiment composition
  - Experiment maintenance and monitoring



General Design Goals

- Extensibility
  - Define a general set of APIs for controlling experiments
  - Novice vs. experienced researchers
  - Long-running services vs. Short-lived experiments
  - Support evolution
- Robustness
  - Resources will fail
  - Researchers will run buggy/broken code
- Scalability
  - Experiments may run on thousands of resources (or more)
- Usability
  - We need to make GENI "easy to use"

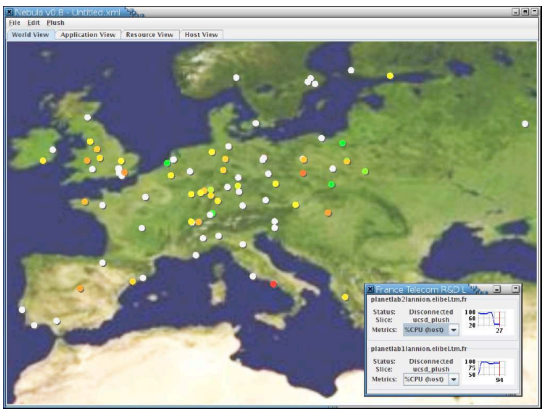
My Goals

- I teach at Williams College
  - Top ranked liberal arts college
  - CS students are bright, ambitious, and like to tinker
- Distributed systems courses are often not taught to undergrads (why?)
- Goal 1: Use GENI as a learning laboratory
  - Help my students appreciate, understand, and experiment with real distributed systems
  - Bring tech-richness of big universities to a small college
- Goal 2: Lower the entry barrier for research in distributed systems
  - Involve undergrads!
  - Skilled undergrads → Smarter grad students and software developers

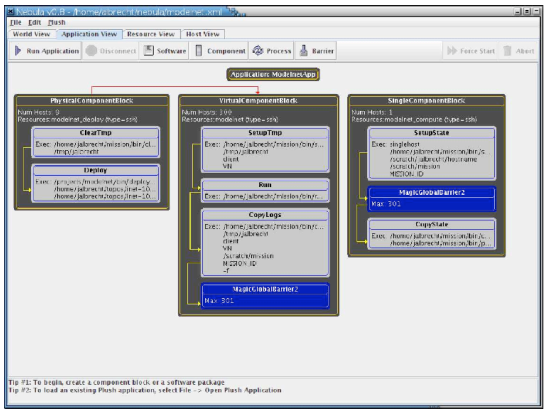
One Potential Solution: Gush

- Gush (GENI User Shell)
  - Scalable experiment control framework for deploying and maintaining GENI experiments
  - Extension of Plush, which was initially designed for PlanetLab application control
  - Uses XML for describing experiments and resources
  - Exports APIs for interacting with resource managers and measurement services
  - Supports three UIs: graphical, command-line, programmatic (currently XML-RPC)

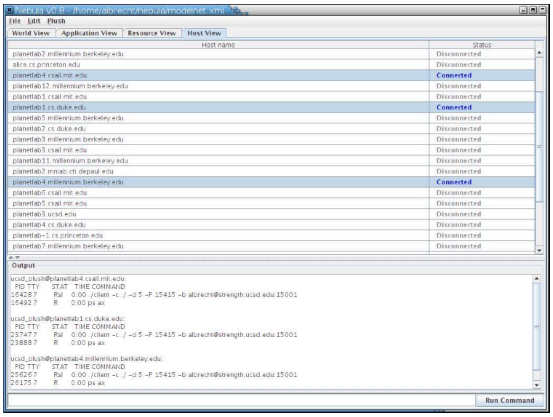
Visualizing Experiments & Resources



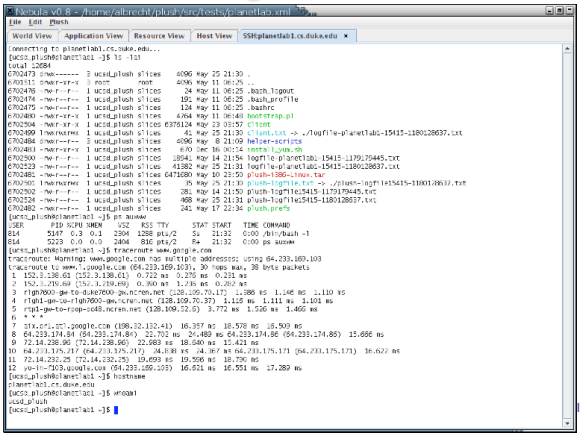
Describing Experiments



Issuing Commands to Resources



Connecting to Resources



Future Plans

- Get feedback from my students regarding usability
- Involve undergrads in development of Gush
  - Currently working with a freshman
  - Two students will work on Gush this summer (hopefully)
- Future goals:
  - Develop API for interacting with Clearinghouses to find resources on behalf of researchers
  - Need to interact with other WG services (perhaps describe resources using RSpecs)
  - Plug in external debugging, measurements, monitoring tools

Questions?