Architecture & Software

Dr. Maximilian Ott  (max@winlab.rutgers.edu)
Overview

• Motivation
  – Why would researchers use ORBIT?
  – Why would they stop using it?

• Architecture
  – Infrastructure
  – Components

• Software
  – Inward facing
  – Customer facing

• Milestones
Why would researchers use ORBIT?

- Lots of resources someone else maintains
- More realistic than NS, but less messy than reality
- Benchmark scenarios allowing to compare apples with apples.
- Community support
- Convenient evolution out of NS
Why would they stop using it?

- Here are 600 nodes – good luck!
- How does it work? Look at the code.
- Whatever you want to do – first, install a OS
- Support? Ha, ha, … (picture favorite Dilbert cartoon)
ORBIT is a Service

• Simple experiments should be easy
• Difficult ones possible
• Layered abstractions for picking the right level of detail and complexity
  – Not all users want to change the NIC driver
  – Not everyone wants to log into each node to start apps
  – Ftp’ing hundreds of log files is not a fun thing
• We don’t know all the useful features
  – This is one of the reasons we proposed parallel EWPs
Experiment Life-cycle

• Simulation (Cluster)
  – Moderate compute facility to run simulations (NS)

• Emulation (Grid)
  – Move simulation to real hardware in controlled and repeatable environment (radio/mobility)

• Trial (Outdoor)
  – Move emulation to real-world scenarios
    • 3G basestation
    • Hotspots across trial area
    • Terminals on buses and fixed locations throughout
Testbed Architecture

[Diagram of testbed architecture with labels for components such as Global Internet, Firewall, High Speed Net, Mobility Server, and various other elements related to network testing and simulation.]
The Grid: Architecture
The Grid: Components

• User portal:
  – How a user will interact with TB
  – How to define, deploy, and interact with an experiment
  – How to collect, and process results of an experiment

• Hardware:
  – Node, physical construction, network, power

• Software
  – Control and manage components (Inward)
  – Services for experimenters (Outward)
Software: Control & manage components

• Fully automated
  – Testbed is open 24/7 (industrial, not academic - 24 weeks, 7 hours)
  – Should run lights-out

• Full access with safety net
  – User can get access to bare hardware, but we want to be able to get it back

• Isolation
  – No pollution from previous, or currently ongoing experiments
  – Isolated networks (challenge on the wireless side)
Software: Services for experimenters

App description → User A

Share

Mount Console → Agent

Spdy gamed

App

Measure. Lib

DB
Milestone 1: 12/03

• 2 x 2 Grid powered, wired, installed in ???
• Autonomously boots, and loads disk over network
• Develop simple experiment using libmac
• Deploy and operate experiment from single machine
• Install and evaluate Emulab cluster with 2 nodes
Milestone 2: 03/04

- 4 x 4 Grid
  - Using new platform
  - Remotely managed
  - Installed in ???
- Autonomously boot, and load disks in parallel (*frisbee*) over network
- Remotely control spectrum analyzer and wave-form generator
- Deploy and operate experiment from user portal
- Partial integration of Emulab user management component
- Recruiting one EWP as guinea pig user
Conclusion

• ORBIT is a tremendous opportunity to move the community past NS

• ORBIT is primarily a service:
  – Requires lots of engineering
  – Requires listening to the users and provide what they need (the customer is always right)
  – Requires solid documentation
  – Requires outreach, hand-holding, …
  – Not necessarily a hallmark of an academic projects

• Finally, we need ALL OF YOU to help us define what ORBIT should be
Architecture & Software

Contact: Max Ott (max@winlab.rutgers.edu)