Running Complex Software on the Grid

Ian Zwiers

August 16, 2002

Outline

- Introduction to the Grid
  - Motivation
  - Ideals
  - Reality
- Software Environments
- Approaches
  - Basic Approach
  - Problem with BaBar
  - AFS Approach
- Current Status

The Grid: Motivation

- Ability to run jobs at many sites
- Even largest computing centers overloaded Example: SLAC
  - Over 1400 "public" computers
  - Totals over 2000 CPU, most with 0.5GB RAM each
  - Around 160 dataservers, 650TB stored
  - Swamped
- Collaboration members want means to contribute their compute resources
- Grid provides the tools

The Grid: Ideals

- Submission of jobs to foreign machines over the Internet
  - Entirely from your workstation
  - No need to login to each machine
  - Like application-independent SETI@Home
- Generic, commoditized computing
  - Reduce need for large centralized computing centers
  - Ability to easily (cheaply) meet sudden demand
- Submit and Forget
  - Load balancing to optimize performance
  - Resilient job monitoring

The Grid: Reality

- Globus Toolkit (*http://www.globus.org*)
  - Run programs on foreign machines (*rsh*, *ssh*)
  - Copy files between machines (*rcp*, *scp*)
  - Browse specifications of foreign machines
    - Number of CPUs, type, speed
    - Quantity RAM installed
    - Disk space available
  - Need only provide password once every 24 hours

- Limitations
  - No load balancing, must specify destination
  - Only transfers executable to foreign machine

Software Environments

- Everything an executable may use
- Typical Components
  - Environment Variables
  - Runtime Libraries
  - Configuration Files
  - Data Files
- Want ability to recreate same software environment on any computer
- Need way to transfer program's filetree
  1. Easy to use
  2. Efficient
  3. Consistent

Bundling Approach

- Only when configuration and data files easily defined

- General steps:

  1. Create archive: executable, runtime libraries, configuration files ...

  2. Copy archive to foreign machine

  3. Create script: extracts archive, sets environment, runs program

  4. Run script on foreign machine

- Observations:

  – Reasonably efficient

  – Mostly automated

  – Doesn't always work

BaBar Software

- Organized into "Packages"

- Serious design problems:
  - Packages assume can access development tree, even at **runtime**
  - Difficult to get list of packages used in particular program
  - Some packages use files from others' development trees

- Problems with bundling:
  - Rarely works automatically: guess'n'check
  - Can't take entire development tree: ~1GB in 5 million lines code, 750 packages

AFS Approach

- "Andrew's File System"
  - Filetree visible from any AFS client
  - Secure, need to login to use filetree
  - Can use globus to avoid typing password
- Place Entire Development Tree in AFS
  - Entire filetree available on all machines
  - Only transfer files as needed
- Limitations
  - Slower transfers than bundling
  - Need an AFS client on foreign machine

Current Status

- When entirely-contained on AFS:
  - Develop on workstation in AFS filetree
  - Compile, link, and run on Grid
- Successes:
  - BaBar Analysis (reconstruction) compilation, linking, execution
  - BaBar MonteCarlo (simulation) execution
  - Require little extra software on foreign machine
- Plans:
  - Run BaBar MonteCarlo for 24 hours around August 26, 14 CPU at 4 sites