An Experimentation Workbench for Replayable Networking Research

Eric Eide, Leigh Stoller, and Jay Lepreau

University of Utah, School of Computing
NSDI 2007 / April 12, 2007

Repeated Research

“A scientific community advances when its experiments are repeated…”

Translation: “I have trouble managing my own experiments.”

Example From My Past

- a distributed, real-time application
- evaluate improvements to real-time middleware
  - vs. CPU load
  - vs. network load
- 4 research groups
- x 19 experiments
- x 56 metrics
- use Emulab

A Laboratory Is Not Enough

- testbeds give you lots of resources…
- …but offer little help in using those resources

What’s Missing: Workflow

- current network testbeds
  - …manage the “laboratory”
  - …not the experimentation process
  - i.e., scientific workflow
- a big problem for large-scale activities

Need

- my experiment needs…
  - encapsulation
  - automation
  - instrumentation
  - preservation
- benefits
  - verify previous results
  - establish base for new research
  - my own, or someone else’s

repeable research
Opportunity

- get the lab manager to help us out!
- **integrated** support for experimental procedures
- resources + encapsulation + automation
- framework: rapid start & common basis
- manage scientific workflow, but also manage lab

Experimentation Workbench

- an environment for “replayable research”
- experiment management + experiment execution
  (but really: help me manage my work)
- all Emulab-managed devices, incl. PlanetLab slivers, ...

- initial design, implementation, and evaluation
- new model of testbed-based experiments
- prototype implementation
- case studies
- lessons learned

Workbench

[Diagram of Emulab]

Classic “Experiments”

- topology + SW (by reference) + events
- expl. DB

Problems

- definition versus instance
- related experiments
- multiple trials per session
- data management
  - instrumentation, collection, archiving, analyzing
- ecosystem
  - topology, software, config, input data, …
- evolution over time

New Model

- template
- instance
- run
- activity
- record
- divide and conquer
- separate the roles that an experiment plays
- evolve the new abstractions
- build on what testbed users already know and do
Templates vs. Experiments

- A template is like a classic Emulab experiment, but a template has...
  - Datastore (file repository)
  - Parameters
  - Multiple instances
  - Metadata
  - History

Template History

- Template instances can also be created programmatically.

Instantiating a Template

- Template instances can also be created programmatically.

Run & Activity

- Runs and activities can be scripted or interactive.
- Prototype: implemented via agents & events.
Record
- template
- instance
- run
- activity
- record

- the "flight recorder" of a run
- parameter values
- input & output files, DBs
- raw data & derived data
- template’s by-reference resources
- dynamically recorded events

Record Repository

evaluation, export, and replay

Evaluation
- how to evaluate?
  - new capabilities ➔ user studies
- goal: early feedback about design & impl.
- approach: three case studies
- outcome: specific & general lessons learned

Study 1: Flexlab Development
- replace ad hoc experiment management
- originally:
  - a configurable ns file
  - start/stop trial scripts
  - "scaffold" in CVS
  - manual archiving
  - destructive modification
- now:
  - templates & params
  - runs, start/stop hooks
  - scaffold & results in WB
  - automatic archiving
  - preserved history

Conclusion: the new model "fits" developers’ model

Study 2: Flexlab Use
- study BitTorrent on Flexlab and PlanetLab
- outcome:
  - parameterization
  - utilized per-run database
  - team communication
  - results for publication
- stress point: latency
- stress point: node failure
Lessons: Storage

- Initial philosophy: "store everything"
  - templates + results + history + metadata + ...
- Space efficiency + group commits
  - Subversion
- Cognitive overload
  - careful UI design

Space and Time

<table>
<thead>
<tr>
<th></th>
<th>Record size (MB)</th>
<th>Stored in repo. (MB)</th>
<th>Elapsed time (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BitTorrent</td>
<td>31.0</td>
<td>18.9</td>
<td>7.0</td>
</tr>
<tr>
<td>GHETE</td>
<td>70.6</td>
<td>19.8</td>
<td>14.4</td>
</tr>
</tbody>
</table>

- Solution: pipeline record-making with user activities
- New problem: isolation
- New approach: branching file systems

What Users Want

- Deletion!
  - not a space concern
  - Cognitive clutter — "junk"
  - Privacy — "mistakes"
- A range of options is required
- "True deletion" is a new requirement

Lessons: The Model

- Initial philosophy: "divide and conquer"
  - More kinds of entities
  - Describe notions and relationships
- Experience:
  - New model does map to users' abstractions
  - Captures separations and connections...
  - "...but not "life cycle" concerns"

“Life Cycle” Concerns

- Multiple levels of abstraction
  - Instance: "the lab"
  - Run & activity: "the work"
- Intertwined & concurrent
  - Workbench must manage experiments and the lab
  - A key difference with "ordinary" scientific workflow systems
- Approach: further refine and enhance our model
  - E.g., adopt features of Plush [Albrecht et al., OSR 40(1)] or SmartFrog [Sabharwal, ICNS '06]

Summary

- Goal: better research ➔ better process tools
- Experiment management + experiment execution
- Prototype builds on existing testbed infrastructure
  - Model maps pretty well to user notions
- Experience: strong integration is required
  - …for overlapping activities safely
  - …for lab management + experiment management
  - …for making it user-friendly in practice
Status

- "alpha test" stage
- internal users
- select external users...
  - mail to testbed-ops@emulab.net

http://www.emulab.net/

Thank you!
Questions?