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
"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

- package / distribute / configure / instrument / init / execute / monitor / stop / collect / analyze / archive / revise / repeat
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
my experiment needs...
- encapsulation
- automation
- instrumentation
- preservation

benefits
- verify previous results
- establish base for new research
- my own, or someone else’s

package / distribute / configure / instrument / init / execute / monitor / stop / collect / analyze / archive / revise / repeat

repeatable 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

emulab
Classic “Experiments”

- topology +
- SW (by reference) +
- events

expt. 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
Template

- template
- instance
- run
- activity
- record

- a “repository”

- definition role of the classic “experiment” notion
- resources by value
  - files, scripts, DB tables, disk images, …
- resources by reference
- prototype: implemented with Subversion (user-hidden)
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
Instantiating a Template

Template instances can also be created programmatically.

Parameters:
- ID: TemplateTest
- Idle-Swap: Swap out this experiment after 2 hours idle.
- Max. Duration: Swap out after 16 hours, even if not idle.
- State Saving: Save disk state on swapout

Formal Parameters:
- CLIENT_COUNT: 2
- DURATION: 60
- HWTYPE: pc850
- or XML file: 
- On Server: [/proj, /groups, /users]

Use this text area for an (optional) description:

Batch Mode Instantiation (See Tutorial for more information)

Linktest Option: Level 3 - (What is this?)

Instantiate
Template Instance

- template
- instance
- run
- activity
- record

- a container of testbed resources
- resource-owner role of classic “experiment” notion
- a transitory object
  - created and destroyed by users and activities
- nodes & network
  - files from the datastore
- database for user
Run & Activity

- template
- instance

- run
- activity

- record

- run: a container of a user-defined “unit of work”
  - defines a context
  - a “trial”
  - one / serial / parallel

- activity: a process, script, workflow, … within a run
  - events & data come from activities in a run

- 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

view, export, and replay

<table>
<thead>
<tr>
<th>Expand</th>
<th>EID</th>
<th>UID</th>
<th>Start Time</th>
<th>Stop Time</th>
<th>Show</th>
<th>Archive</th>
<th>Export</th>
<th>Replay</th>
<th>Load DBs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TT</td>
<td>stoller</td>
<td>2006-08-01 12:28:12</td>
<td>2006-08-01 14:04:01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TT</td>
<td>stoller</td>
<td>2006-08-07 14:09:03</td>
<td>2006-08-08 06:10:03</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TT</td>
<td>stoller</td>
<td>2006-08-08 07:09:38</td>
<td>2006-08-08 07:24:51</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Show</th>
<th>Archive</th>
<th>RunID</th>
<th>ID</th>
<th>Start Time</th>
<th>Stop Time</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>TT</td>
<td>1</td>
<td>2006-08-08 07:09:38</td>
<td>2006-08-08 07:20:14</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>TT-R2</td>
<td>2</td>
<td>2006-08-08 07:20:32</td>
<td>2006-08-08 07:23:13</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>TT-R3</td>
<td>3</td>
<td>2006-08-08 07:23:51</td>
<td>2006-08-08 07:24:51</td>
<td></td>
</tr>
</tbody>
</table>
Evaluation and Lessons Learned
Evaluation

- how to evaluate?
  - *new capabilities* \(\rightarrow\) *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
  - \(\rightarrow\) Subversion

- cognitive overload
  - \(\rightarrow\) careful UI design
Space and Time

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

<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>
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?