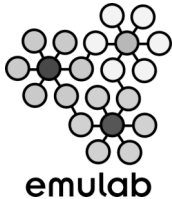


# An Experimentation Workbench for Replayable Networking Research



Eric Eide, Leigh Stoller,  
and Jay Lepreau

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

# Repeated Research



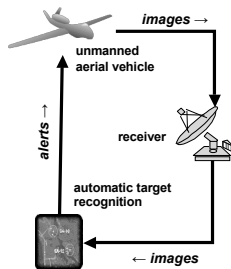
"A scientific community advances when its experiments are repeated..."

Translation: "I have trouble managing my own experiments."

2

# 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



3

# 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



4

# 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

5

# Need

- 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



6



## 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

7



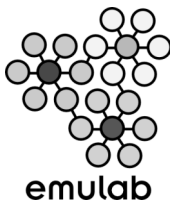
## Experimentation Workbench

- 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

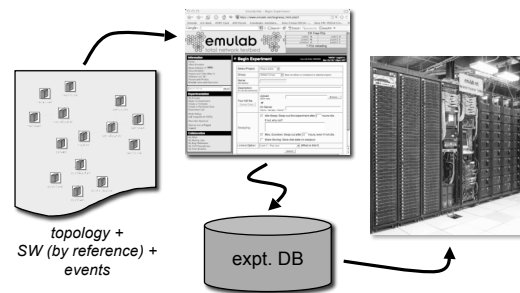


8

## Workbench



## Classic “Experiments”



10








## 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

11








## New Model

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

12

## Template

- *template*  • a "repository"
- *instance*  • definition role of the classic "experiment" notion
- *run*  • resources by value
- *activity*  • files, scripts, DB tables, disk images, ...
- *record*  • resources by reference

- prototype: implemented with Subversion (*user-hidden*)

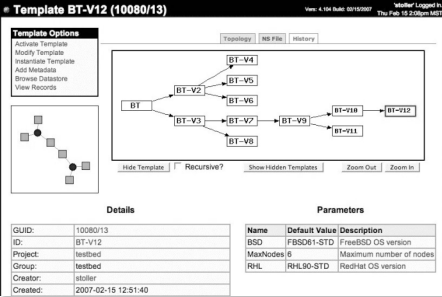
13

## Templates vs. Experiments

- a template is like a classic Emulab experiment, but a template has...
  - *datastore (file repository)*
  - *parameters*
  - *multiple instances*
  - *metadata*
  - *history*

14


## Template History



15

## Instantiating a Template






parameters



template instances can also be created programmatically

16






## Template Instance

- *template*  • a container of testbed resources
- *instance*  • resource-owner role of classic "experiment" notion
- *run*  • a transitory object
- *activity*  • created and destroyed by users and activities
- *record*  • nodes & network

- files from the datastore
- database for user

17

## Run & Activity

- *template*  • *run: a container of a user-defined "unit of work"*
- *instance*  • defines a context
- *run*  • a "trial"
- *activity*  • one / serial / parallel
- *record*  • 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

18

## 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

19

## Record Repository

view, export, and replay

20

## Evaluation and Lessons Learned

emulab

## Evaluation

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

22

## 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**

23

## 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

24



## Lessons: Storage

- initial philosophy: “store everything”
  - *templates + results + history + metadata + ...*
- space efficiency + group commits
  - → *Subversion*
- cognitive overload
  - → *careful UI design*

25



## Space and Time

	Record size (MB)	Stored in repo. (MB)	Elapsed time (m)
BitTorrent	31.0	18.9	7.0
GHETE	70.6	19.8	14.4

- **solution:** pipeline record-making with user activities
- **new problem:** isolation
- **new approach:** branching file systems

26



## 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

27



## Lessons: The Model

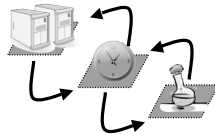
- initial philosophy: “divide and conquer”
  - *more kinds of entities*
  - *describe notions and relationships*
- experience:
  - *new model does map to users' abstractions*
  - *captures separations and connections...*
  - *...but not “life cycle” concerns*

28



## “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]*



29



## 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*

30



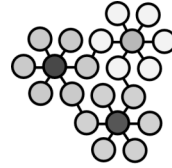
## Status

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



31

<http://www.emulab.net/>



emulab

**Thank you!**  
**Questions?**