

GIMI

Thierry Rakotoarivelo, Cong Wang,
Mike Zink

UMass
GEC 18

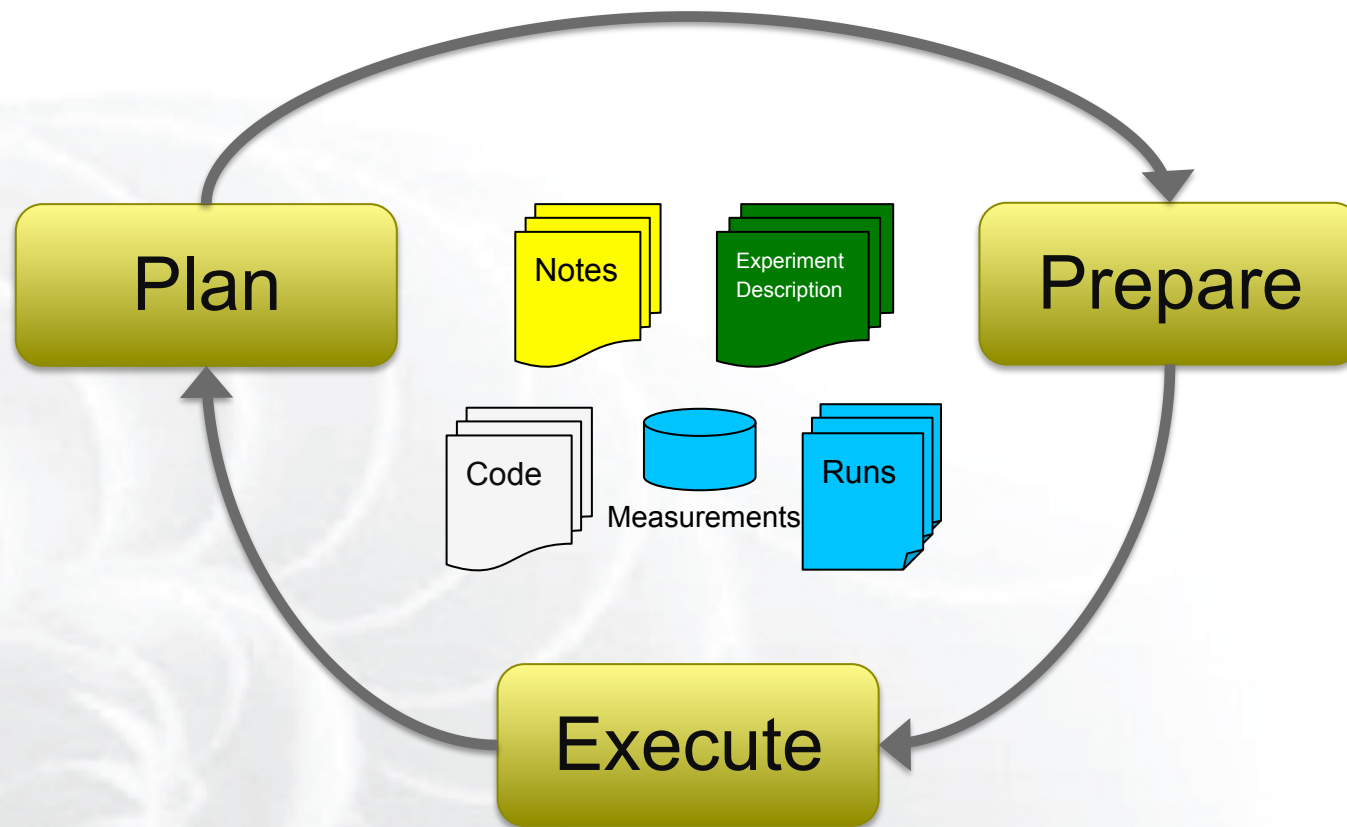




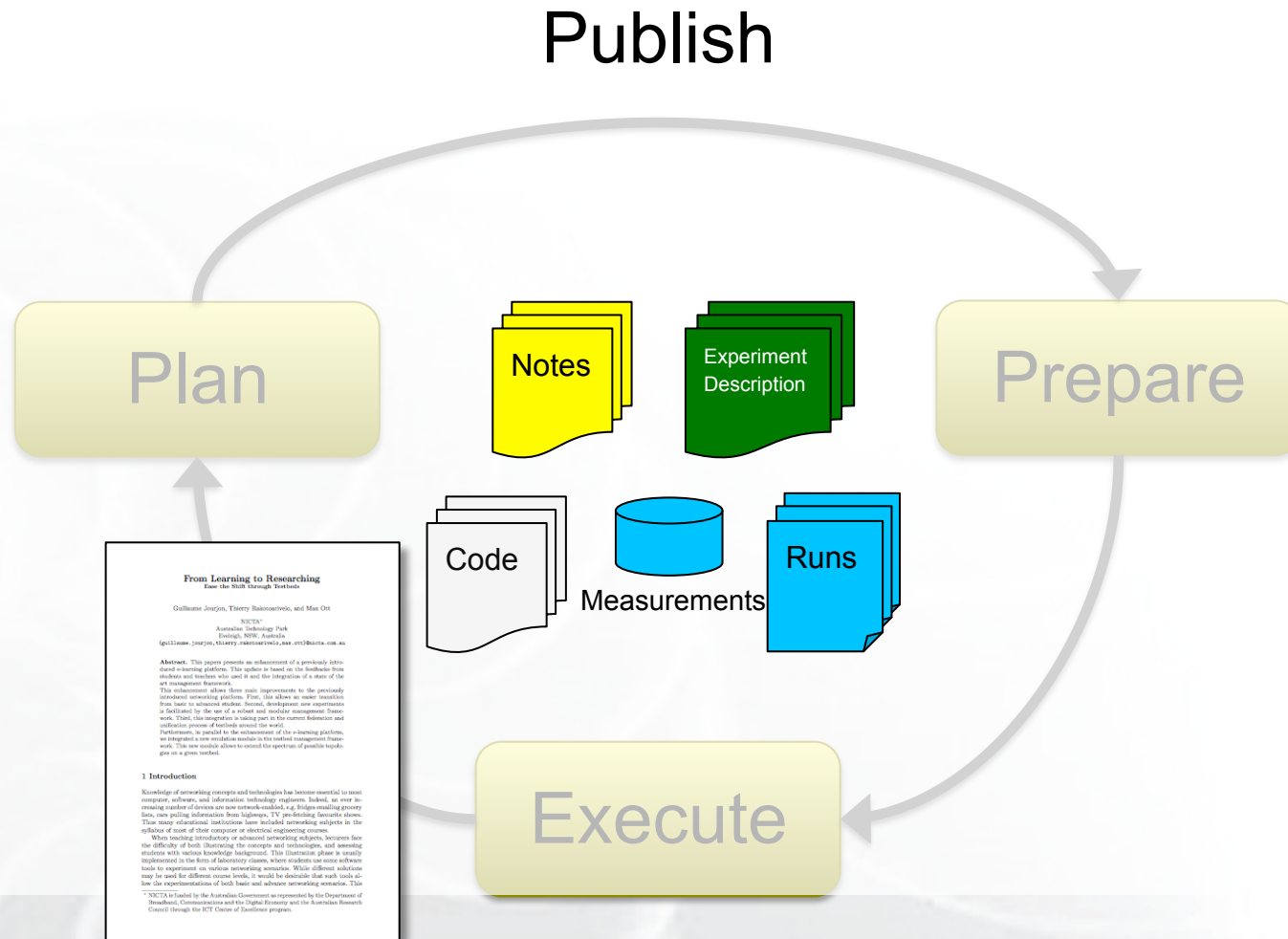
“Perform basic measurements on ExoGENI testbed”

- ✓ Simple topology
- ✓ Repeatable experiment
- ✓ Ability to monitor and document experiment

The “Experiment Cycle”



The “Successful Experiment Cycle”



The “Experiment Cycle” in a Tool: LabWiki

localhost:4000/labwiki
user1 Log out


Plan

Prepare

Execute

Tutorial: First Experiment

As mentioned before, we want to configure an experiment as shown below:



The first step is to describe the experiment in OEDL, the OMF Experiment Description Language. To see how this looks for this experiment, open the '1_hello.rb' file in the **Prepare** column.

Ignoring some of the details we can see the definition of two resource groups, **Sender** in line 6 and **Receiver** in line 21.

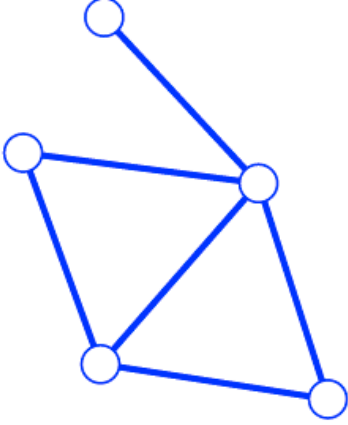
```
6: defGroup('Sender', ...
...
21: defGroup('Receiver', ...
```

There will be more on groups later, but in this

```
git:default:repo/oid/tutorial/1_hello.oedl

1 defProperty('res1', 'omf.nicta.node1', "
2 defProperty('res2', 'omf.nicta.node2', "
3 essid = (0...8).map{65.+(rand(25)).chr}.
4 channel = rand(11)+1
5
6 defGroup('Sender', property.res1) do |no
7   node.addApplication("test:app:otg2") d
8   app.setProperty('udp:local_host', '1
9   app.setProperty('udp:dst_host', '192
10  app.setProperty('udp:dst_port', 3000
11  #app.measure('udp_out', :interval =>
12  app.measure('udp_out', :samples => 1
13  end
14  node.net.w0.mode = "adhoc"
15  node.net.w0.type = 'g'
16  node.net.w0.channel = channel
17  node.net.w0.essid = essid
18  node.net.w0.ip = "192.168.0.2"
19 end
20
21 defGroup('Receiver', property.res2) do |
22  node.addApplication("test:app:otr2") d
23  app.setProperty('udp:local_host', '1
24  app.setProperty('udp:local_port', 30
25  #app.measure('udp_in', :interval =>
```

Slice 70256298659140





localhost:4000/labwiki



LabWiki

user1 Log out

Plan

Prepare

Execute



Search

Tutorial

As mentioned before, we want to configure an experiment as shown below:



Wiki

The first step in configuring an experiment in OEDL, the OEDL Description Language. For this experiment, open the '1_hello.rb' file in the Prepare column.

Ignoring some of the details we can see the definition of two resource groups, **Sender** in line 6 and **Receiver** in line 21.

```

6: defGroup('Sender', ...
...
21: defGroup('Receiver', ...
  
```

There will be more on groups later, but in this

git:default:repo/oidl/tutorial/1_hello.o

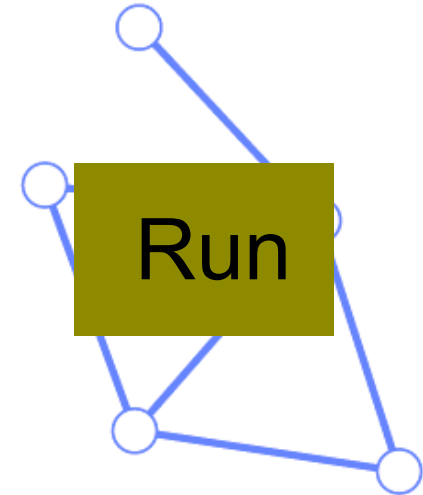


```

1 defProperty('res1', 'omf.nicta.node1', "
2 defProperty('res2', 'omf.nicta.node2', "
3 essid = (0...8).map(65.+(rand(25)).chr).
4 channel = rand(11)+1
5
6 defGroup('Sender', property.res1) do |no
7   node.addApplication("test:app:otg2") d
8   app.s
9   app.s
10  app.s
11  #app.
12  app.m
13 end
14 node.net.w0.mode = "adhoc"
15 node.net.w0.type = 'g'
16 node.net.w0.channel = channel
17 node.net.w0.essid = essid
18 node.net.w0.ip = "192.168.0.2"
19 end
20
21 defGroup('Receiver', property.res2) do |
22   node.addApplication("test:app:otr2") d
23   app.setProperty('udp:local_host', '1
24   app.setProperty('udp:local_port', 30
25   #app.measure('udp_in', :interval =>
  
```

Edit

Slice 7255298659140



Run

LabWiki Core

Plan

Plugin

Prepare

Your Plugin

Execute

GENI
CH/AM

OMF

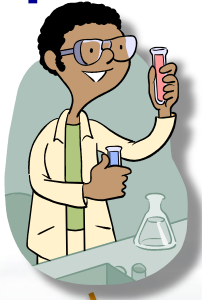
GIMI
Services

iRODS

Your
Service

- Status:
 - Open-source MIT License
 - Code:
 - <https://github.com/mytestbed/labwiki>
 - Bug reports & documentation (hahaha):
 - <http://omf.mytestbed.net/projects/labwiki>
 - Plugin example:
 - https://github.com/mytestbed/labwiki_topology_plugin

Experimenter



1. Instrument

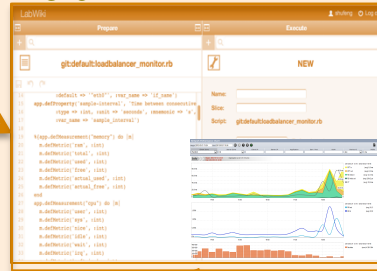
6. Obtain



Automated

5. Save

LabWiki

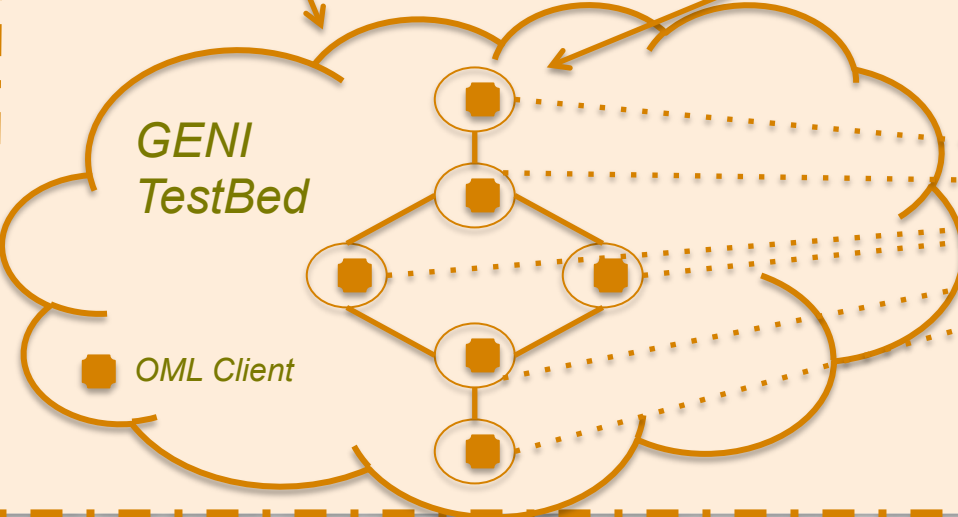


4. Plot

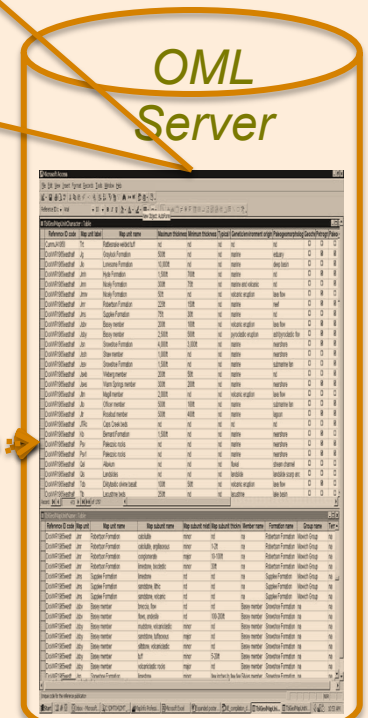
0. Reserve

2. Run

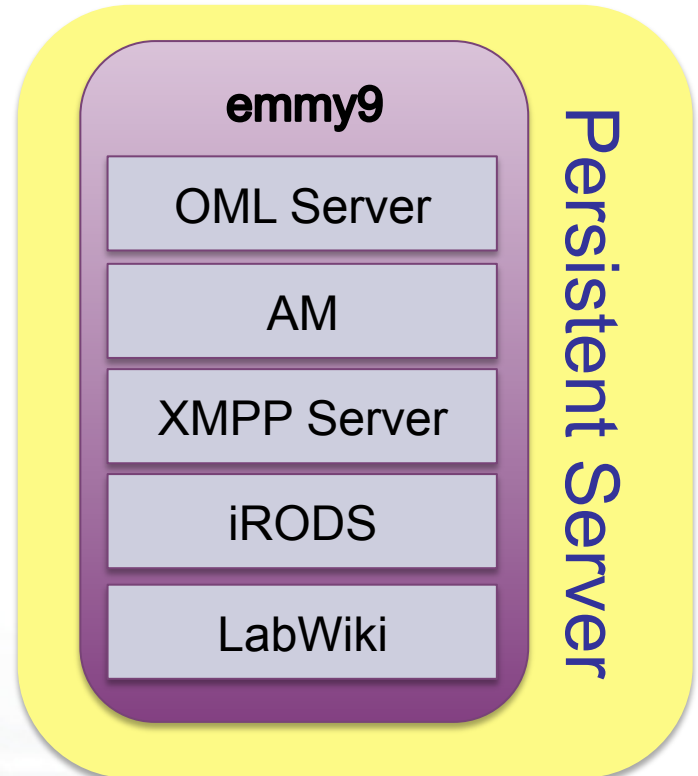
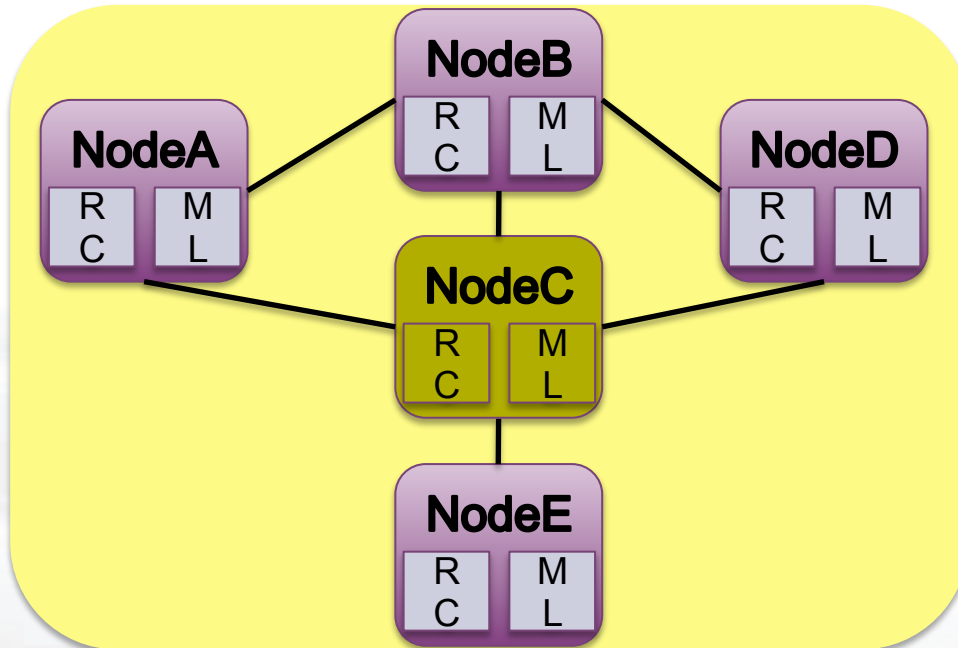
3. Collect



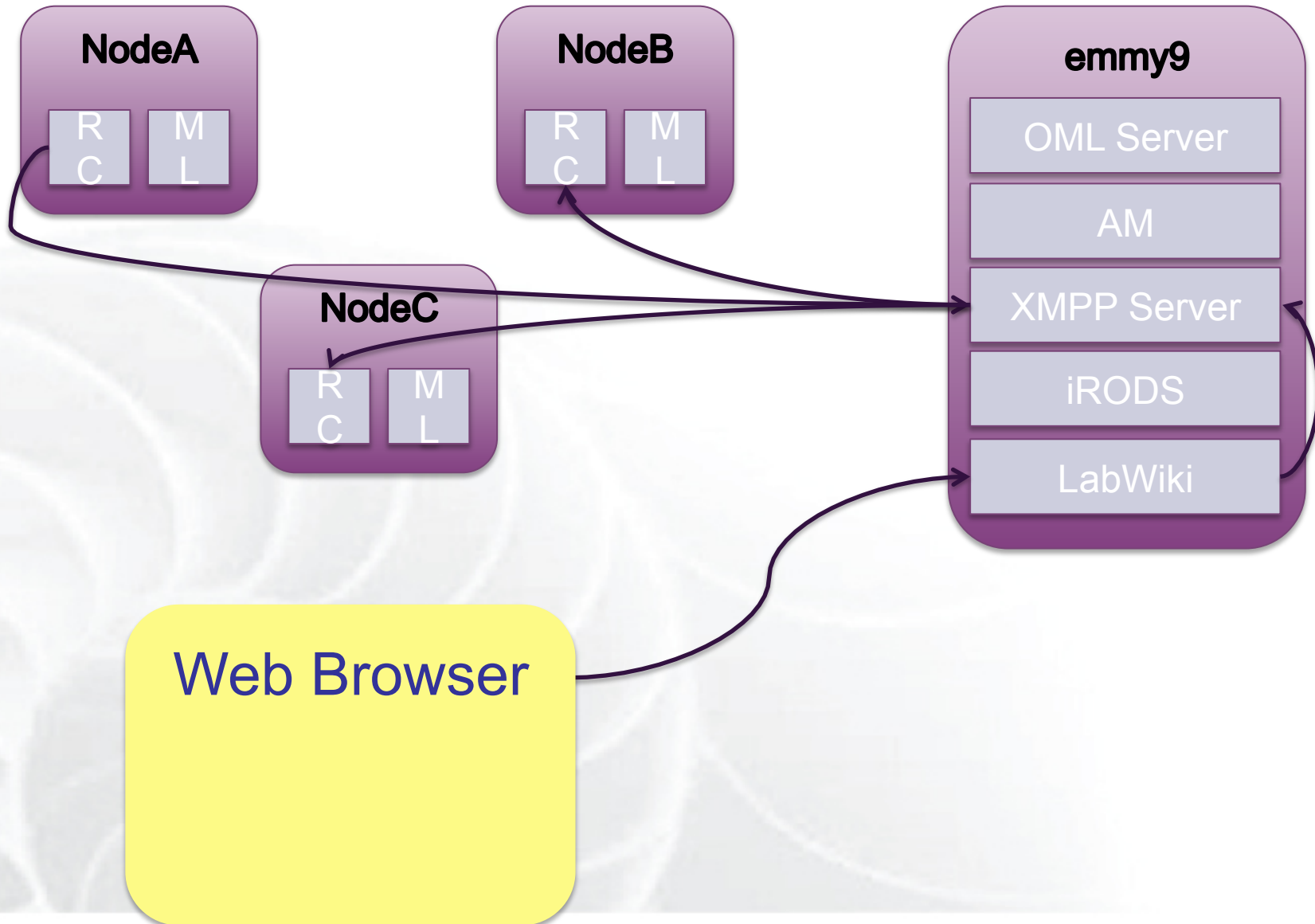
OML Server

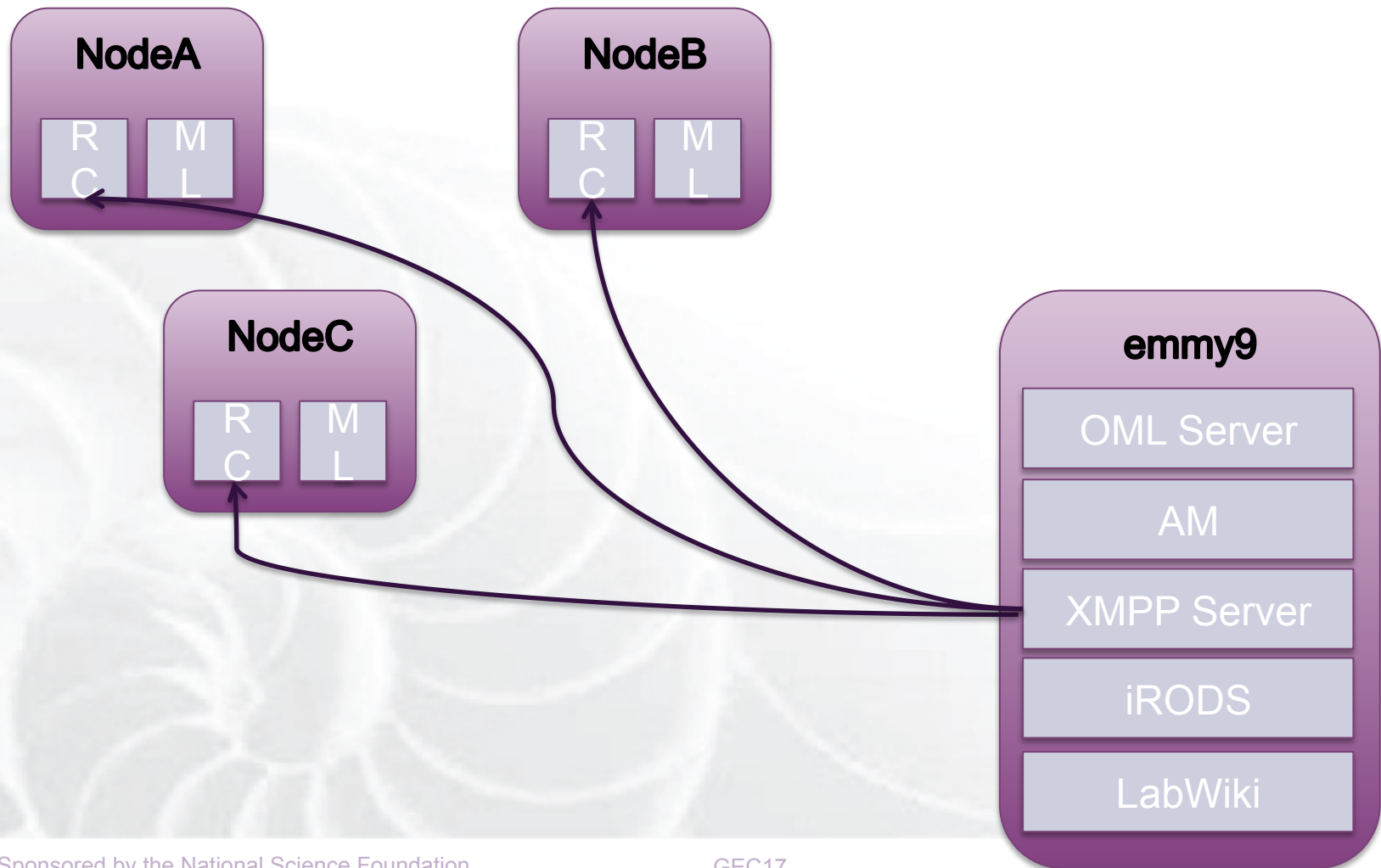


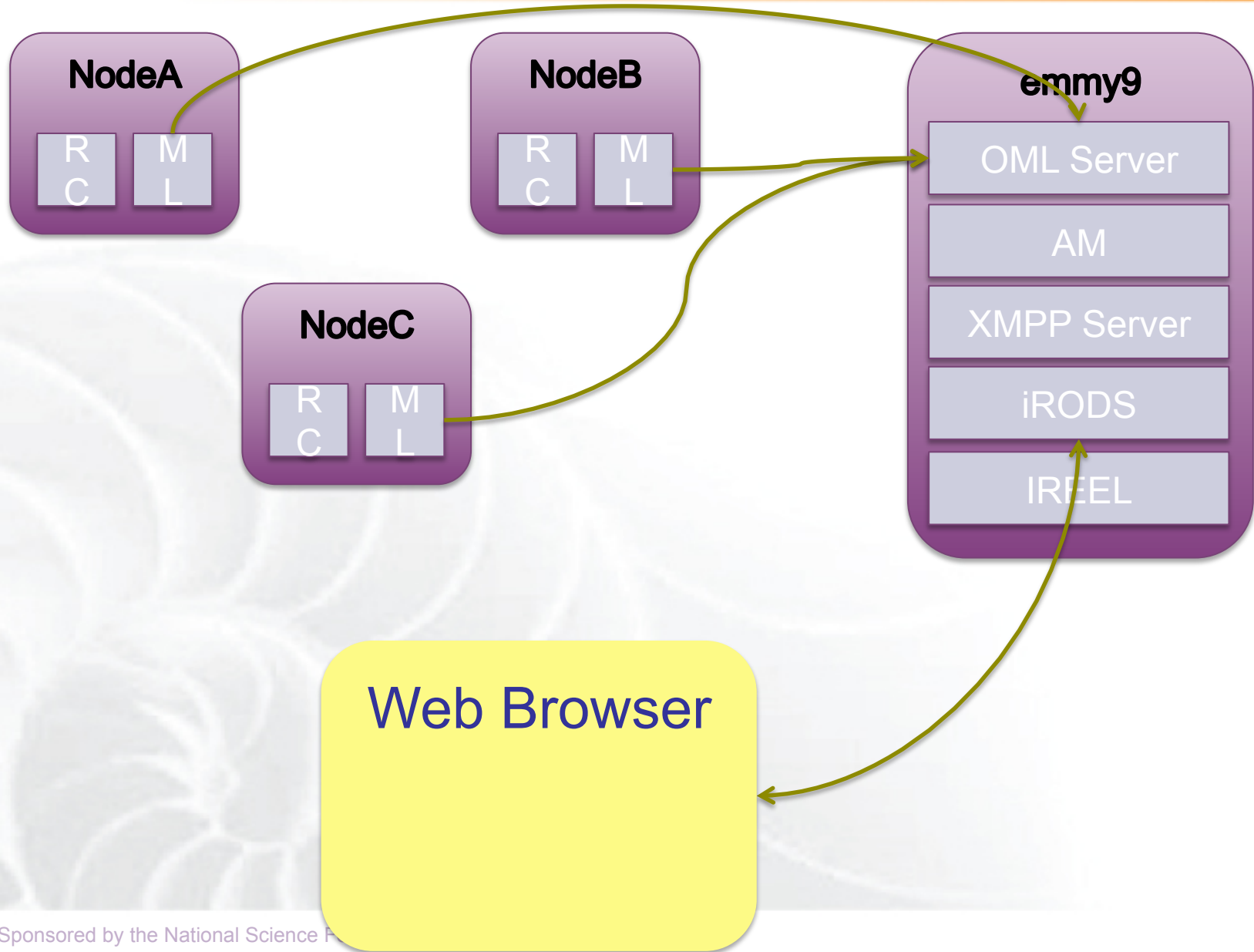
Host	IP	Port	State	...
Control	10.10.10.10	10000	OK	...
Control	10.10.10.11	10000	OK	...
Control	10.10.10.12	10000	OK	...
Control	10.10.10.13	10000	OK	...
Control	10.10.10.14	10000	OK	...
Control	10.10.10.15	10000	OK	...
Control	10.10.10.16	10000	OK	...
Control	10.10.10.17	10000	OK	...
Control	10.10.10.18	10000	OK	...
Control	10.10.10.19	10000	OK	...
Control	10.10.10.20	10000	OK	...
Control	10.10.10.21	10000	OK	...
Control	10.10.10.22	10000	OK	...
Control	10.10.10.23	10000	OK	...
Control	10.10.10.24	10000	OK	...
Control	10.10.10.25	10000	OK	...
Control	10.10.10.26	10000	OK	...
Control	10.10.10.27	10000	OK	...
Control	10.10.10.28	10000	OK	...
Control	10.10.10.29	10000	OK	...
Control	10.10.10.30	10000	OK	...
Control	10.10.10.31	10000	OK	...
Control	10.10.10.32	10000	OK	...
Control	10.10.10.33	10000	OK	...
Control	10.10.10.34	10000	OK	...
Control	10.10.10.35	10000	OK	...
Control	10.10.10.36	10000	OK	...
Control	10.10.10.37	10000	OK	...
Control	10.10.10.38	10000	OK	...
Control	10.10.10.39	10000	OK	...
Control	10.10.10.40	10000	OK	...
Control	10.10.10.41	10000	OK	...
Control	10.10.10.42	10000	OK	...
Control	10.10.10.43	10000	OK	...
Control	10.10.10.44	10000	OK	...
Control	10.10.10.45	10000	OK	...
Control	10.10.10.46	10000	OK	...
Control	10.10.10.47	10000	OK	...
Control	10.10.10.48	10000	OK	...
Control	10.10.10.49	10000	OK	...
Control	10.10.10.50	10000	OK	...



Web Browser







```
#!/bin/bash
cd /local
read -r slice</var/emulab/boot/nickname
slicename=$(echo $slice | cut -f2 -d.)
host=$(hostname)
host1=$(echo $host | cut -f1 -d.)
hostname $host1
curl http://emmy9.casa.umass.edu/pingWrap.rb -o /root/pingWrap.rb
chmod +x /root/pingWrap.rb
curl
http://emmy9.casa.umass.edu/omf-resctl.yaml -o /etc/omf-resctl-5.4/omf-
resctl.yaml
perl -i.bak -pe "s/\\:slice\\:\\:slice\\: $slicename/g" /etc/omf-resctl-5.4/omf-resctl.yaml
/etc/init.d/omf-resctl-5.4 restart"
VM-3.dbhatslice.emulab-net
```

- New GIMI functionality
- P2P-based streaming

- Client/server-based, adaptive streaming experiment

