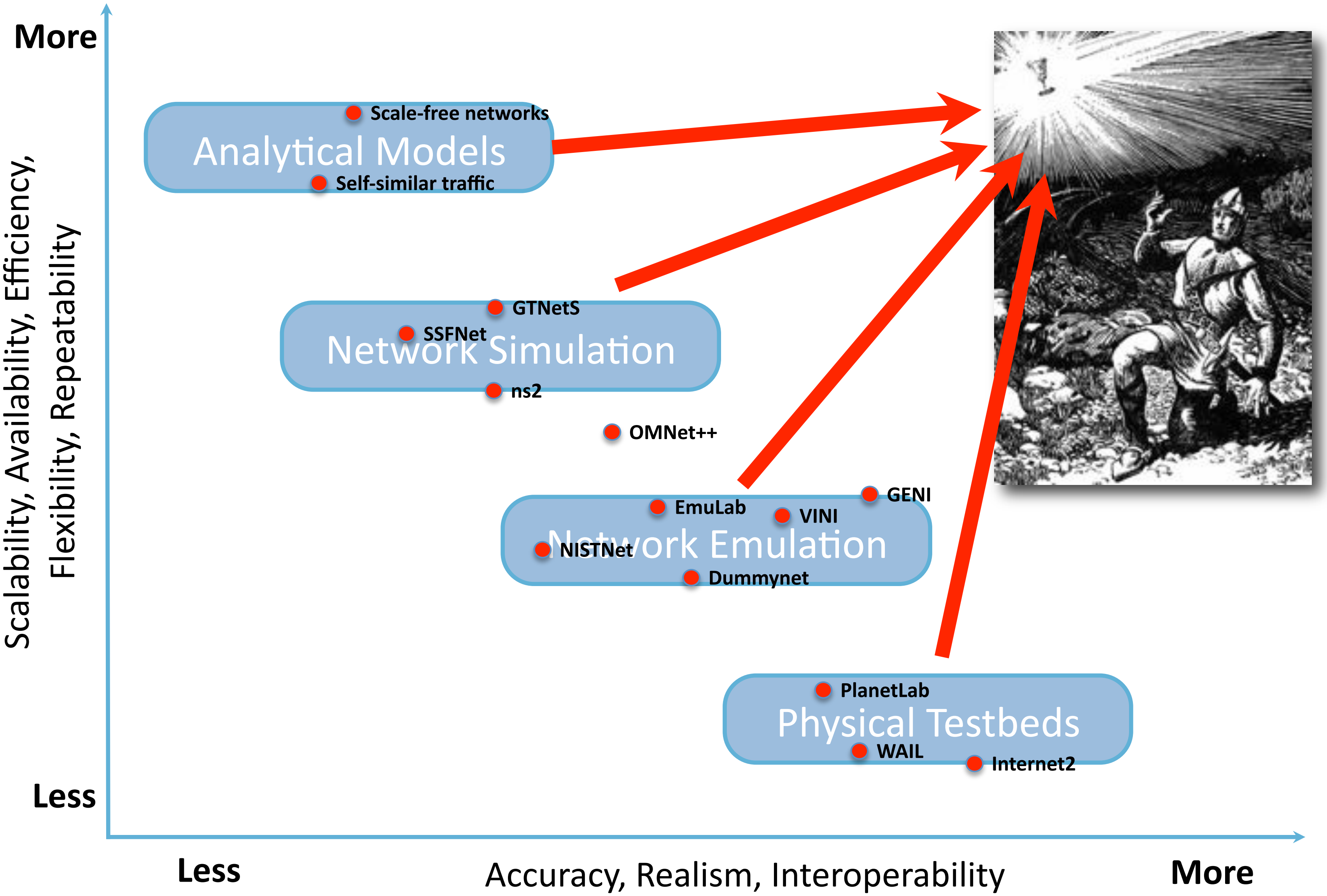
PrimoGENI Current State and Demonstration

Jason Liu, PI
Julio Ibarra, Heidi Alvarez, co-PI
Miguel Erazo, Nathanael Van Vorst, grad students
Eduardo Pena, Eduardo Tibau, undergrad students
Florida International University

GEC 9 ProtoGENI Cluster Meeting
November 2, 2010, Washington DC

Outline

- PrimoGENI intro
- New PRIME
- Slingshot
- Demonstration



More

Less

Scalability, Availability, Efficiency, Flexibility, Repeatability

Analytical Models

Scale-free networks

Self-similar traffic

Network Simulation

GTNetS

SSFNet

ns2

OMNet++

Network Emulation

EmuLab

VINI

GENI

NISTNet

Dummynet

Physical Testbeds

PlanetLab

WAIL

Internet2

Less

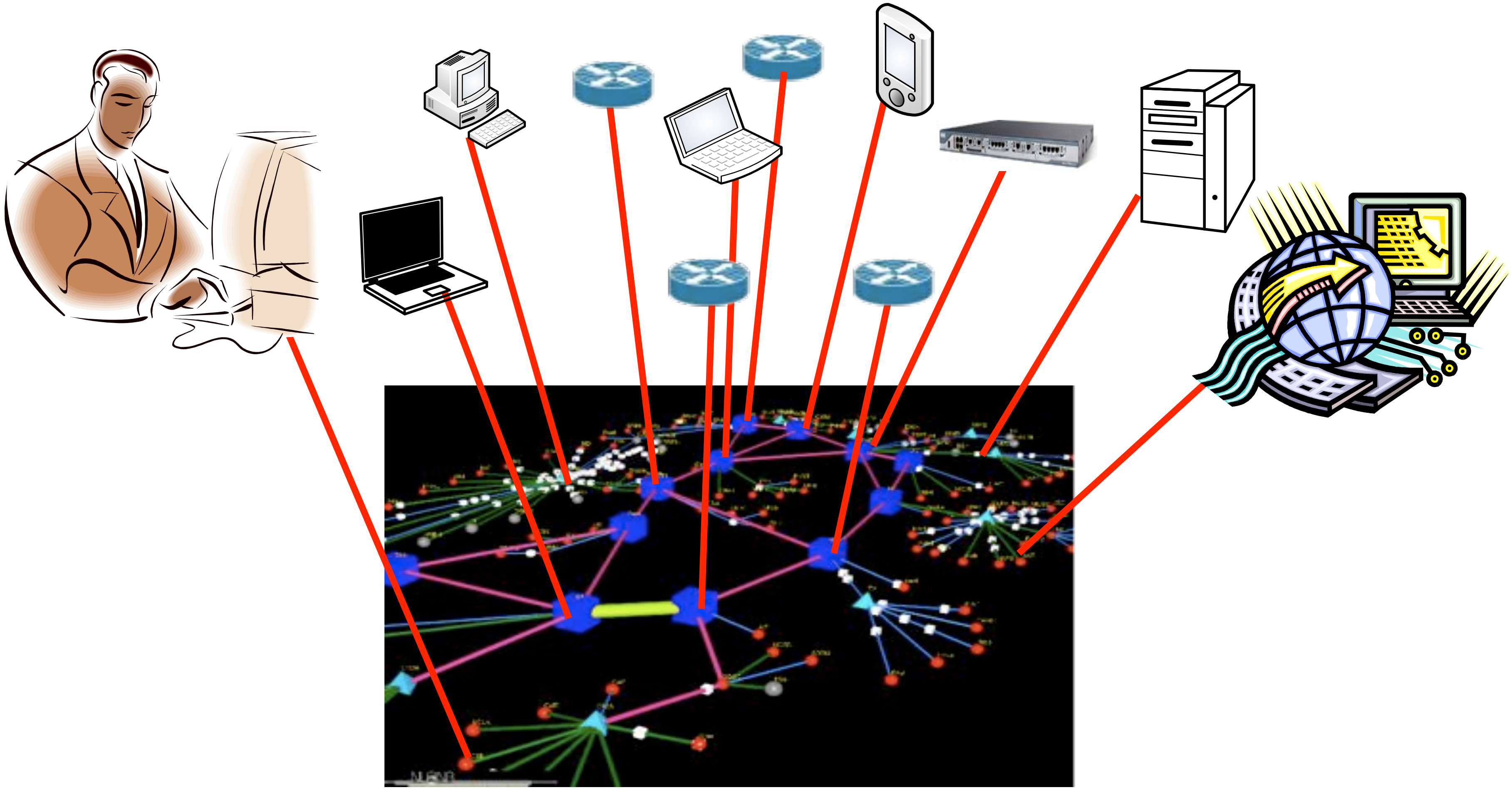
Accuracy, Realism, Interoperability

More

Disclaimer: The tools shown on this slide should not be interpreted as positioned exactly in terms of capabilities.

PrimoGENI Project

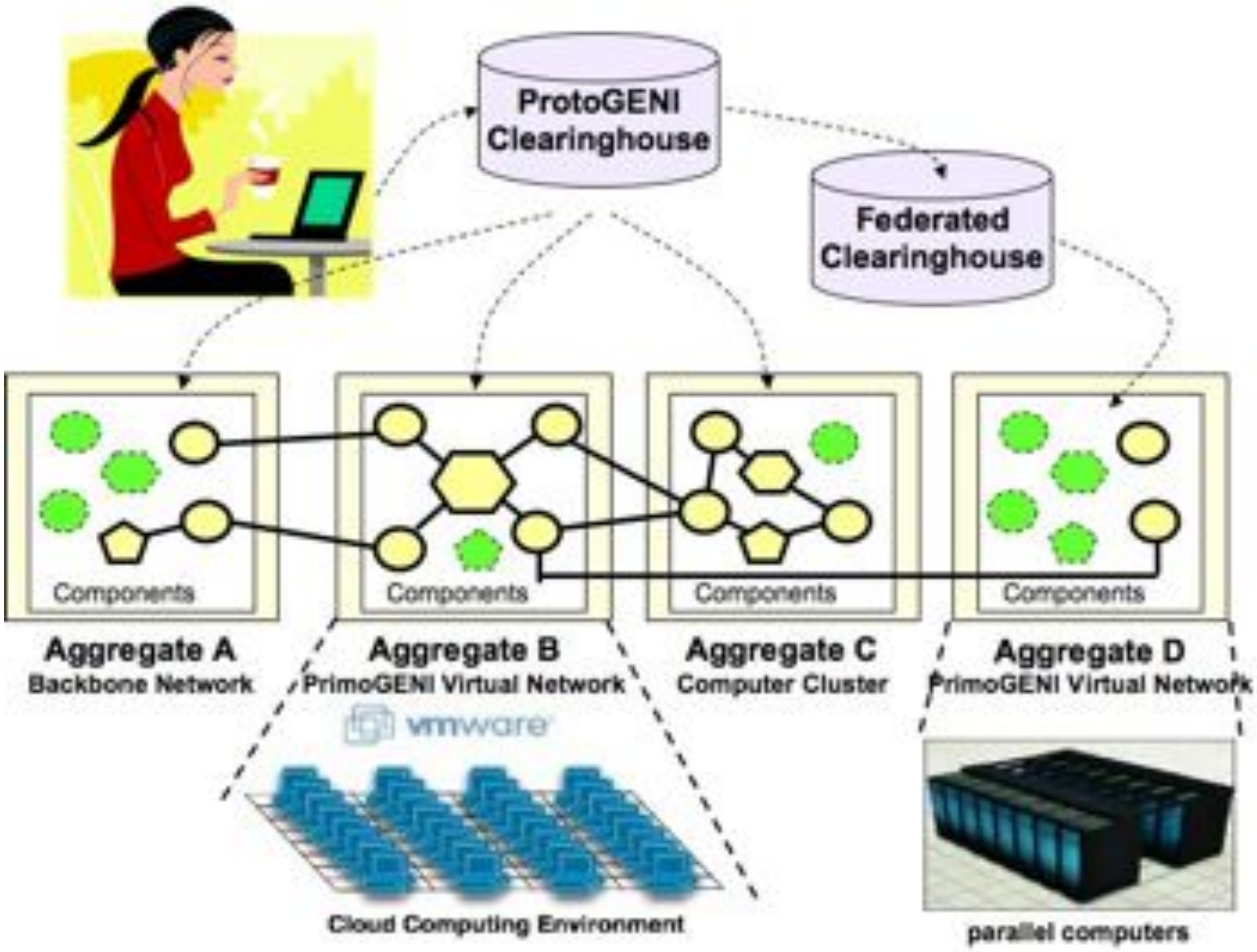
- The goal is to enable hybrid network experiments on GENI
  - Physical, simulated and emulated network components



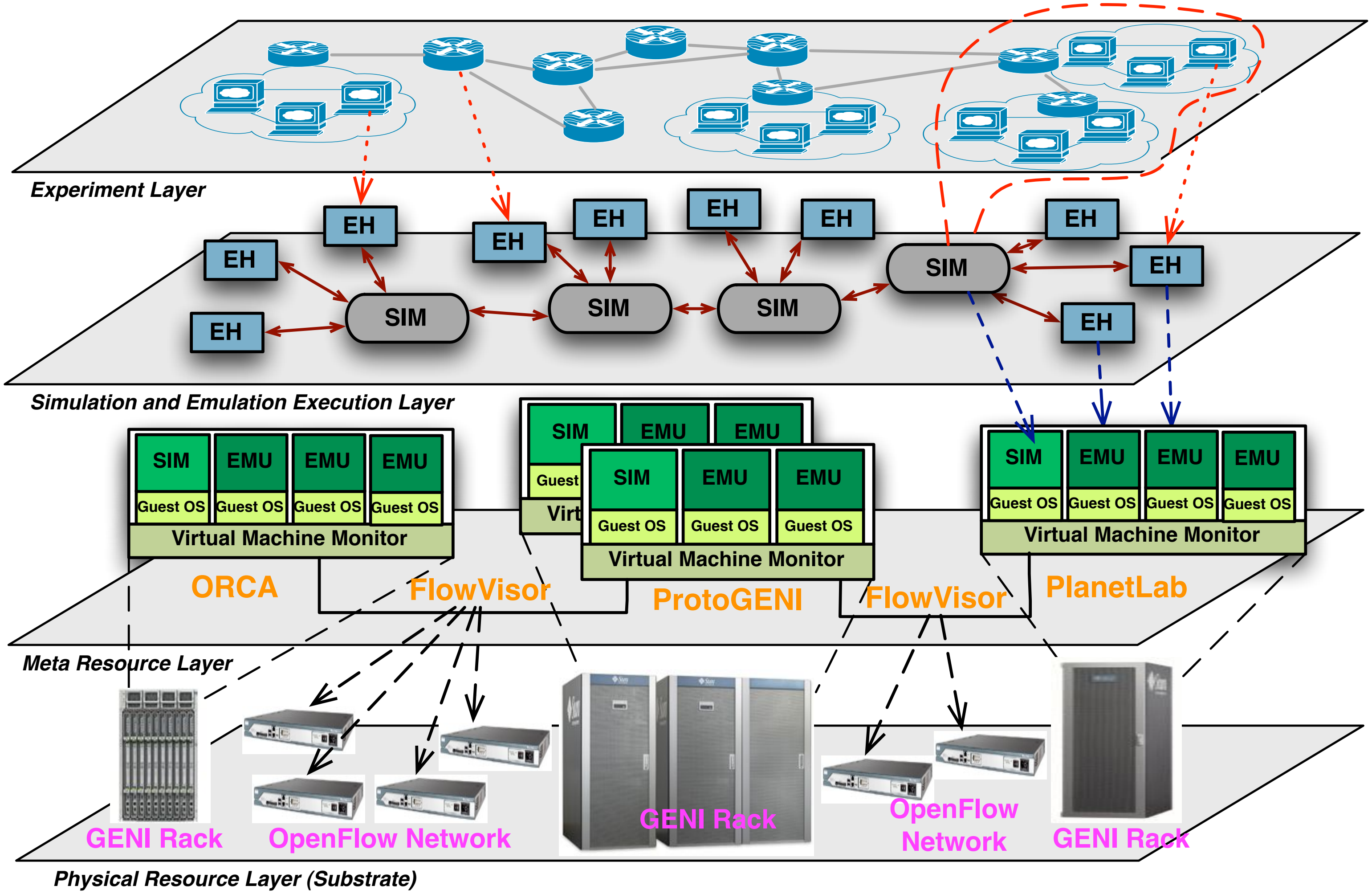
Simulation, Emulation, and Real-time Simulation

- Simulation:
  - Millions of simulated network entities (routers, hosts, links)
  - Common network protocols (TCP, UDP, BGP, …)
  - Large network models, and background network traffic
- Emulation:
  - Thousands of emulated elements (on virtual machines)
  - Unmodified network applications, protocols, services
- Real-time simulation:
  - Abstracted network models
  - Parallel and distributed simulation
  - High-speed conduit for emulated traffic

PrimoGENI – A GENI Aggregate for Large-scale Network Simulation/Emulation Experiments



PrimoGENI Experiments



New PRIME

Existing Network Simulators

- ns-2 [Breslau et al., 2000]
  - Widely used
  - Many protocols and applications
  - Many users
  - Ongoing ns-3 code revamping
- Large-scale simulators: SSFNet, PDNS, GTNets, …
  - Based on parallel and distributed simulation
  - Constraints: realism, protocols, support
  - Difficult to use!
- So why YET another simulator?

What are the Problems?

- Large-scale simulation
  - Memory consumption grows *super-linearly* with model size
  - For example, routing …
- Large-scale emulation
  - "Emulation" here means running large number of unmodified applications in a virtual/simulated network
  - High traffic volume, low latency
- Large-scale Interaction
  - Monitor and steer network experiments
  - Real time

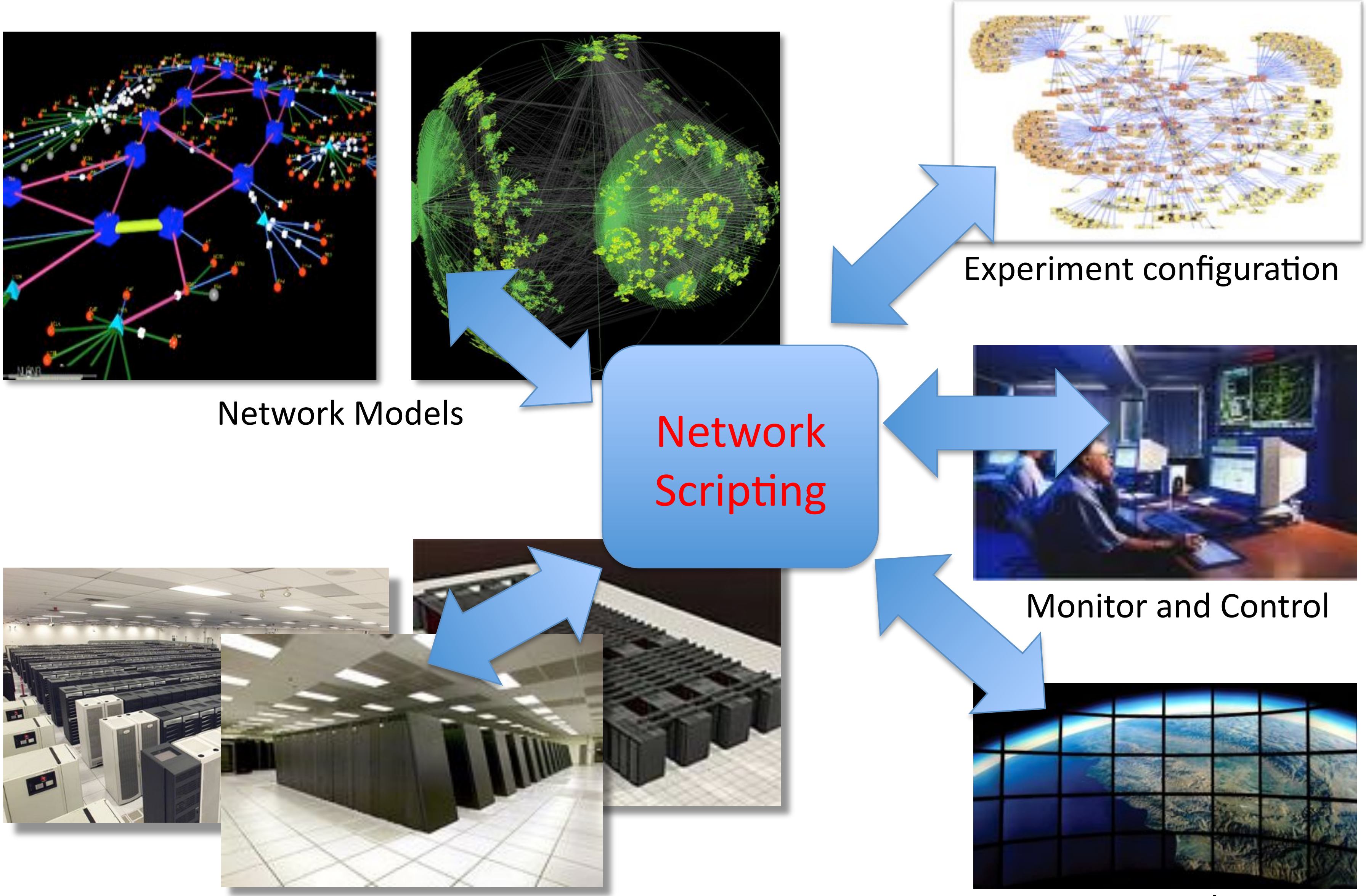
What Problems?

- Large-scale simulation
  - Memory consumption grows *super-linearly* with model size
  - For example, routing …
- Large-scale emulation
  - “Emulation” here means running large number of unmodified applications in a virtual/simulated network
  - High traffic volume, low latency
- Large-scale Interaction
  - Monitor and steer network experiments
  - Real time

The Problem

We need to manage the entire ``life cycle'' of a large-scale network experiment, including model configuration, resource specification, simulation deployment, execution, online monitoring and steering, inspection, visualization, data collection and analysis.

Network Scripting



Network Scripting

Network Models

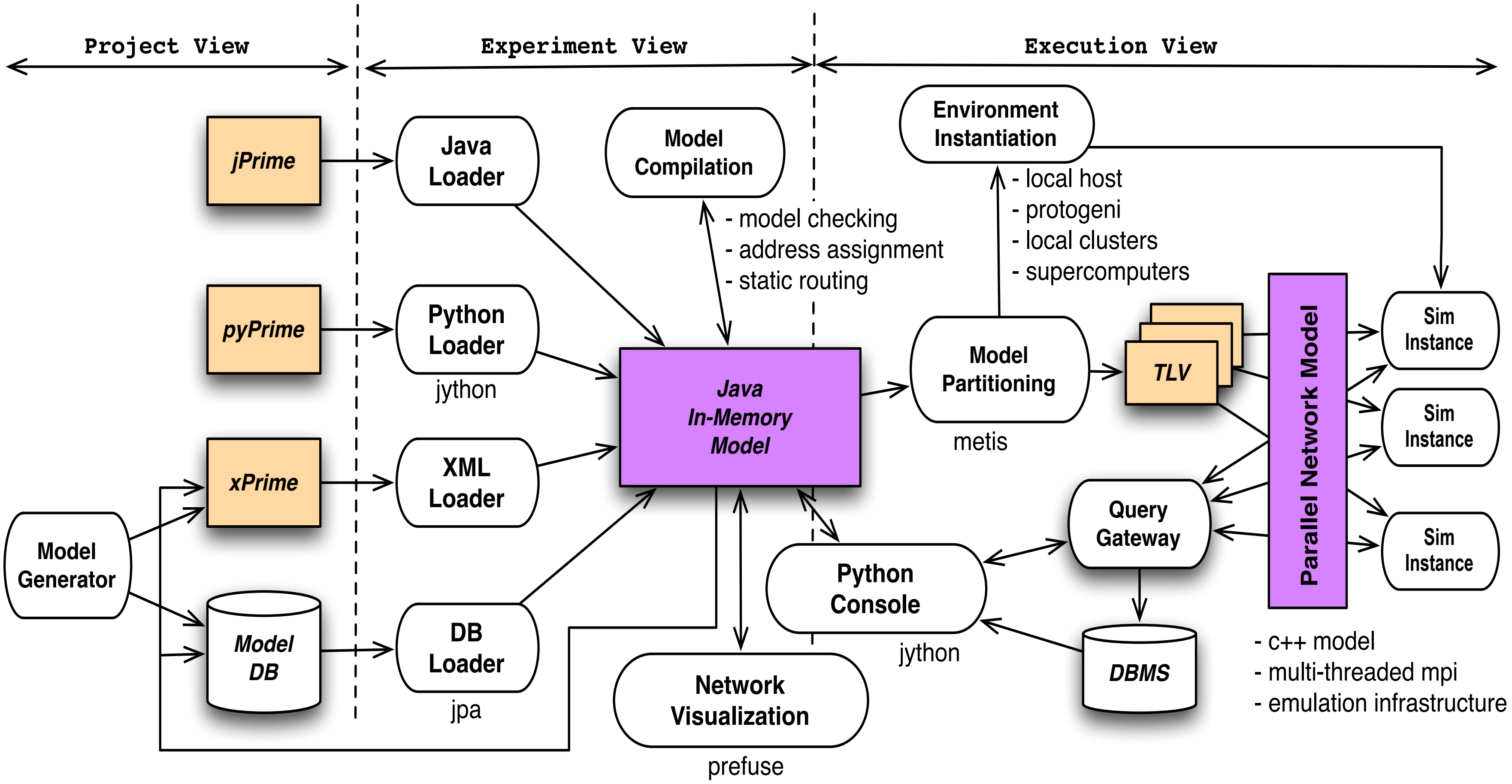
Experiment configuration

Monitor and Control

Data Visualization

Simulation & Emulation Execution Environment

A Tale of Two Models



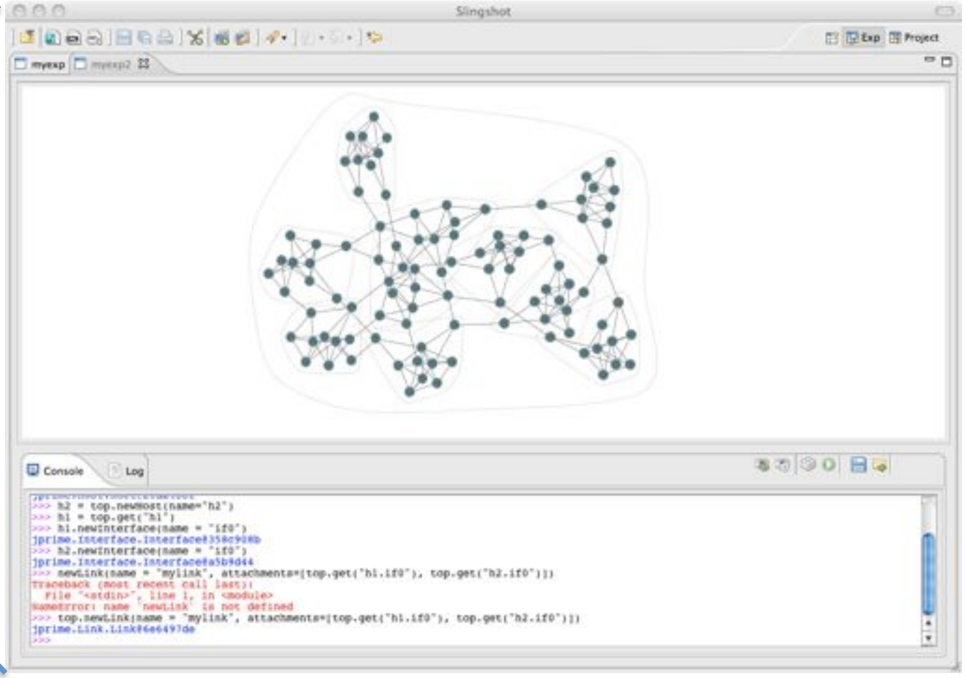
An integrated client-side GUI for building network models, managing resources, and monitoring and steering experiments

Step 1

User installs Slingshot on client machine

User obtains Emulab account and installs PrimoGENI images

Step 2



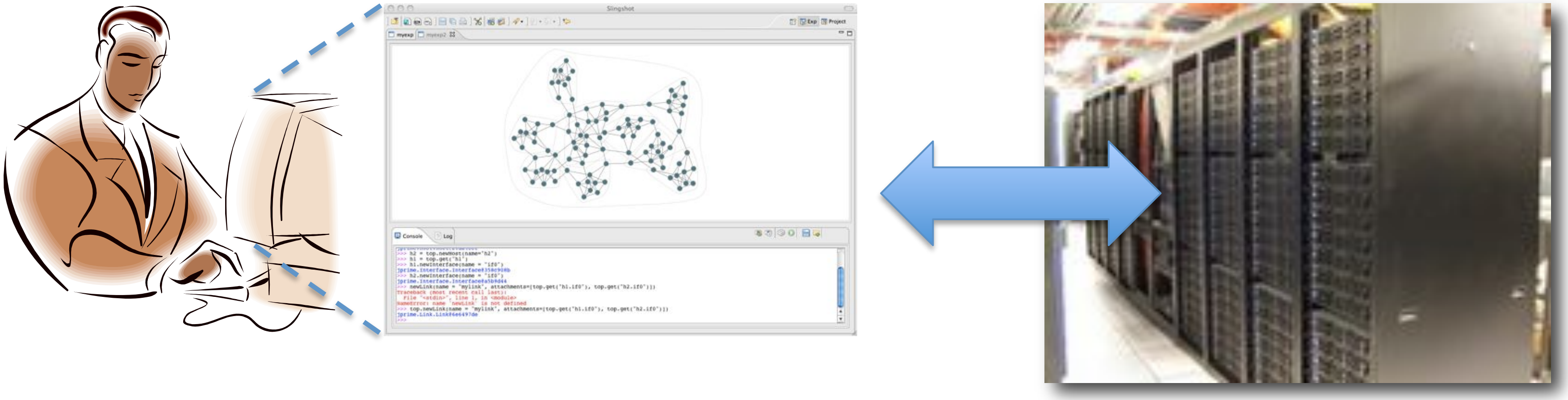
```
>>> h2 = top.newHost(name="h2")
>>> h1 = top.get("h1")
>>> h1.newInterface(name = "if0")
jprime.interface.Interface@358c928b
>>> h2.newInterface(name = "if0")
jprime.interface.Interface@a5b9d44
>>> newLink(name = "mylink", attachments=[top.get("h1.if0"), top.get("h2.if0")])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'newLink' is not defined
>>> top.newLink(name = "mylink", attachments=[top.get("h1.if0"), top.get("h2.if0")])
jprime.Link.Link@6e6497de
>>>
```

User constructs network model using Slingshot (in Python, Java, or XML, using model generators, or from database)

Step 3

User specifies resources, uses Slingshot to allocate the resources (through ProtoGENI), and initiates the experiment (TO BE EXPLAINED)

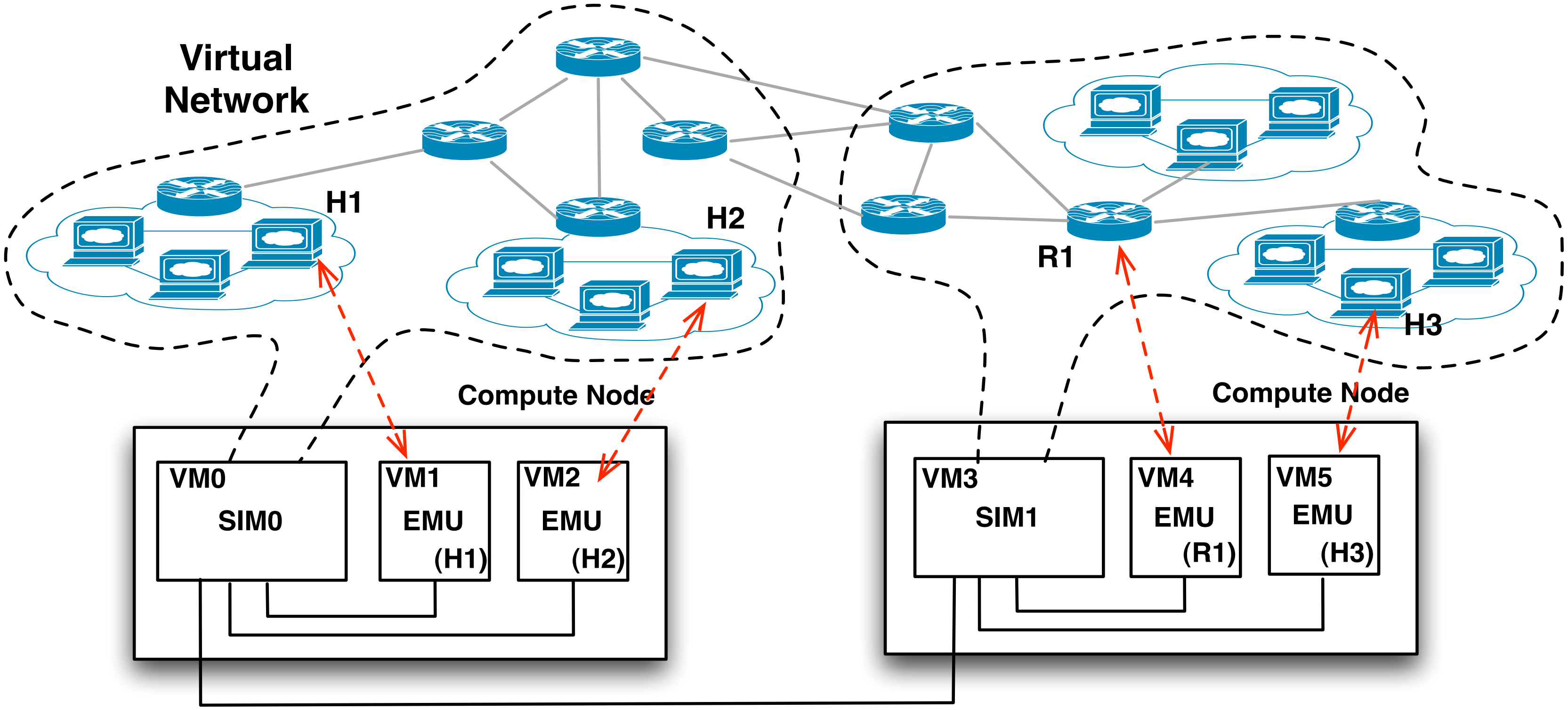
Step 4



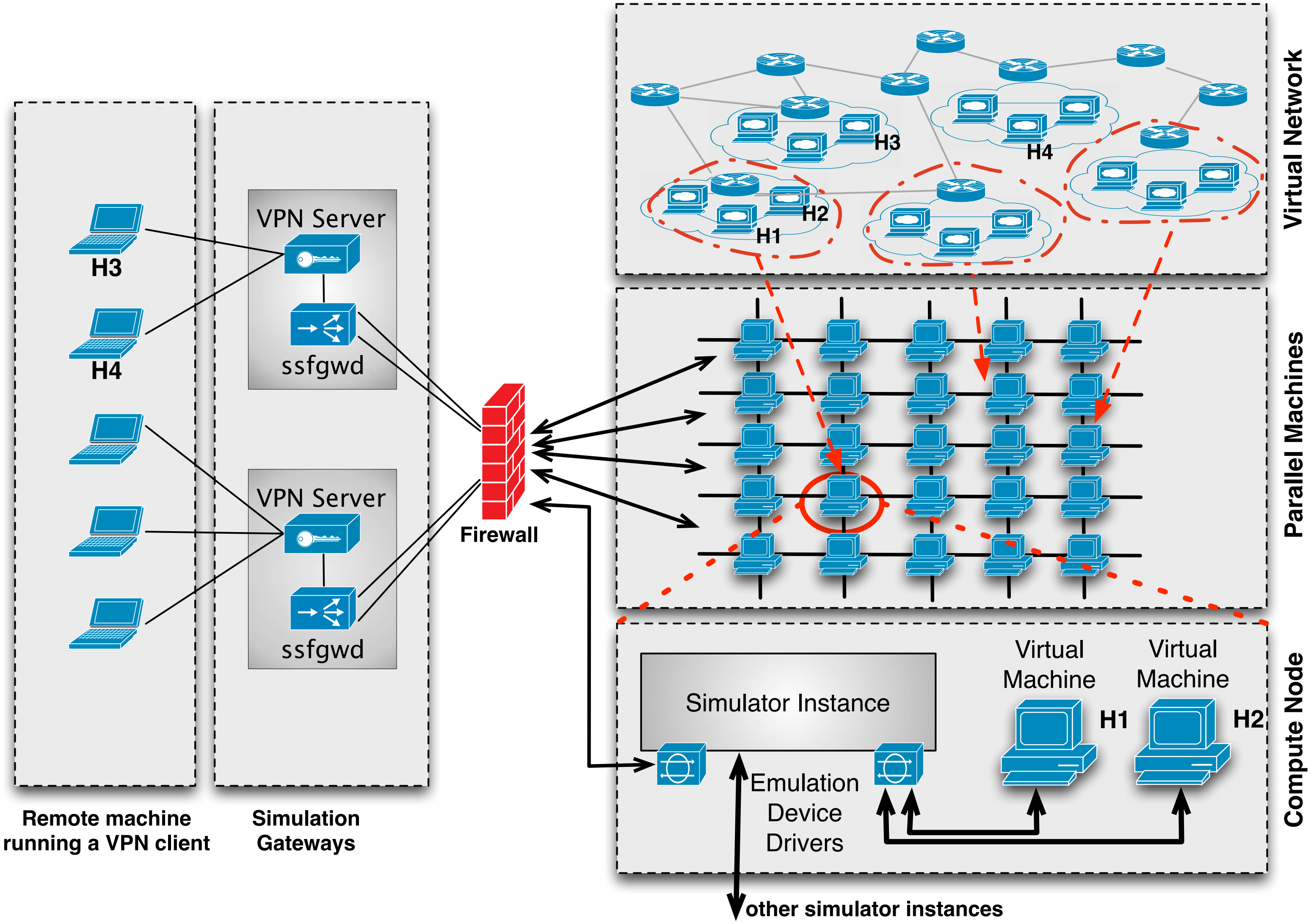
```
>>> h2 = top.newHost(name="h2")
>>> h1 = top.get("h1")
>>> h1.newInterface(name = "if0")
jprime.interface.Interface@358c908b
>>> h2.newInterface(name = "if0")
jprime.interface.Interface@a5b9d44
>>> newLink(name = "mylink", attachments=[top.get("h1.if0"), top.get("h2.if0")])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'newLink' is not defined
>>> top.newLink(name = "mylink", attachments=[top.get("h1.if0"), top.get("h2.if0")])
jprime.Link.Link@6e6497de
>>>
```

User uses Slingshot to monitor and steer the experiment in real time, and happily live after

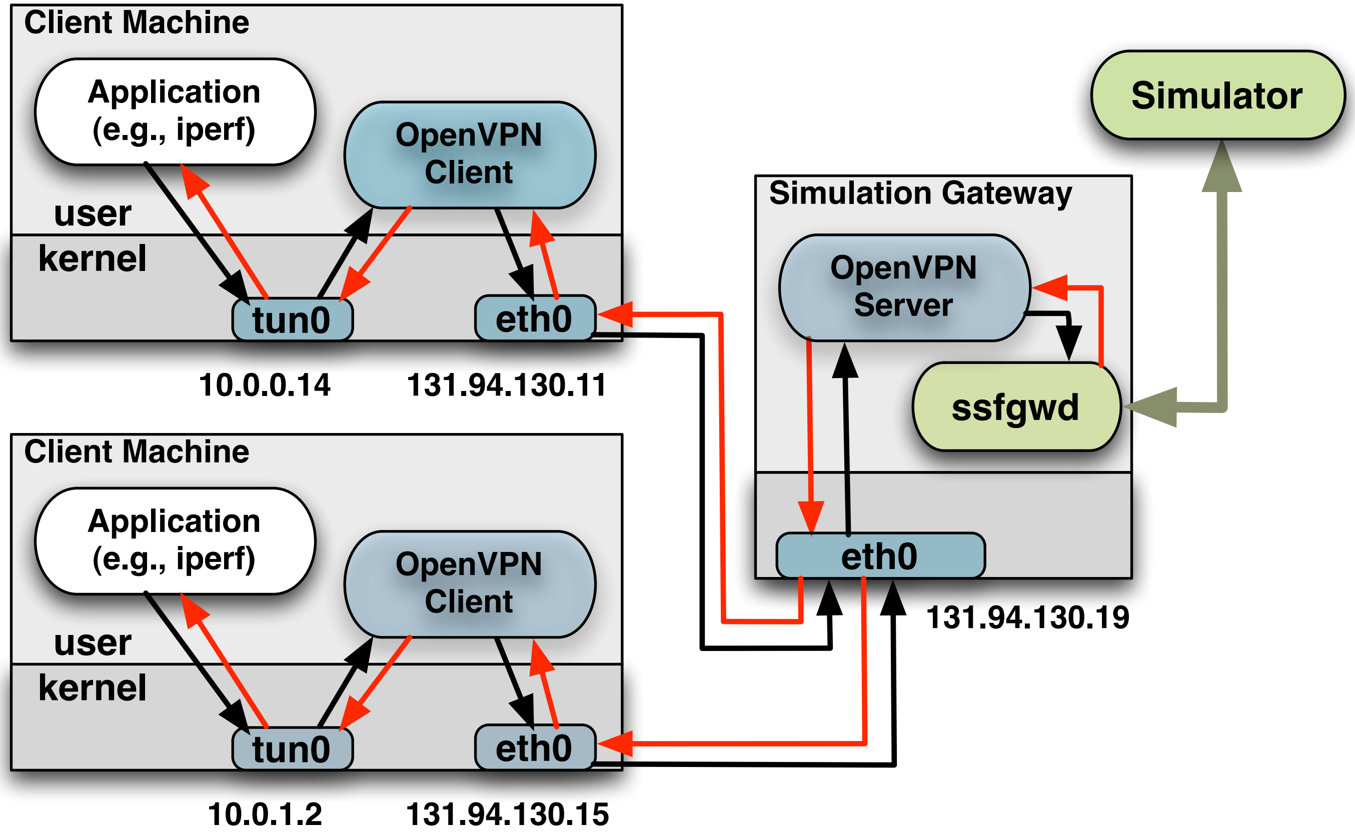
Back to Step 3



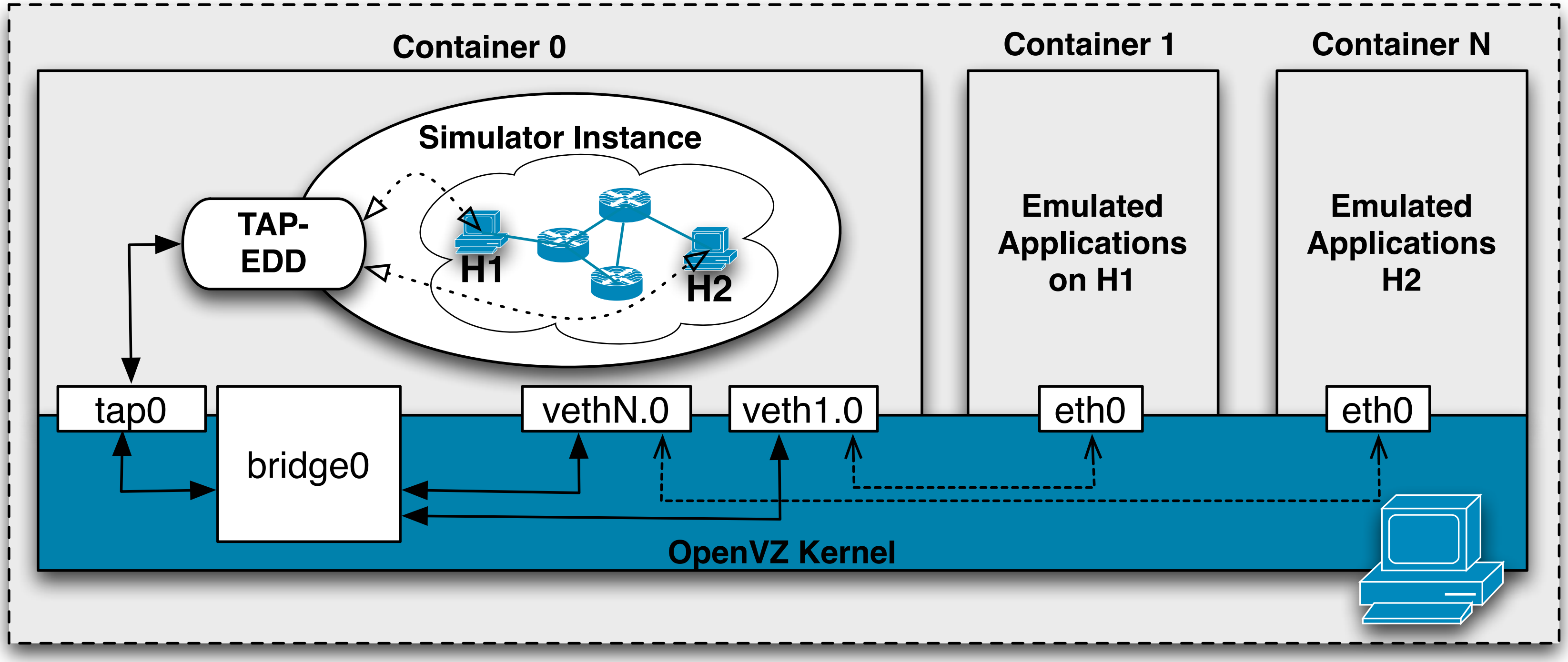
The Emulation Infrastructure



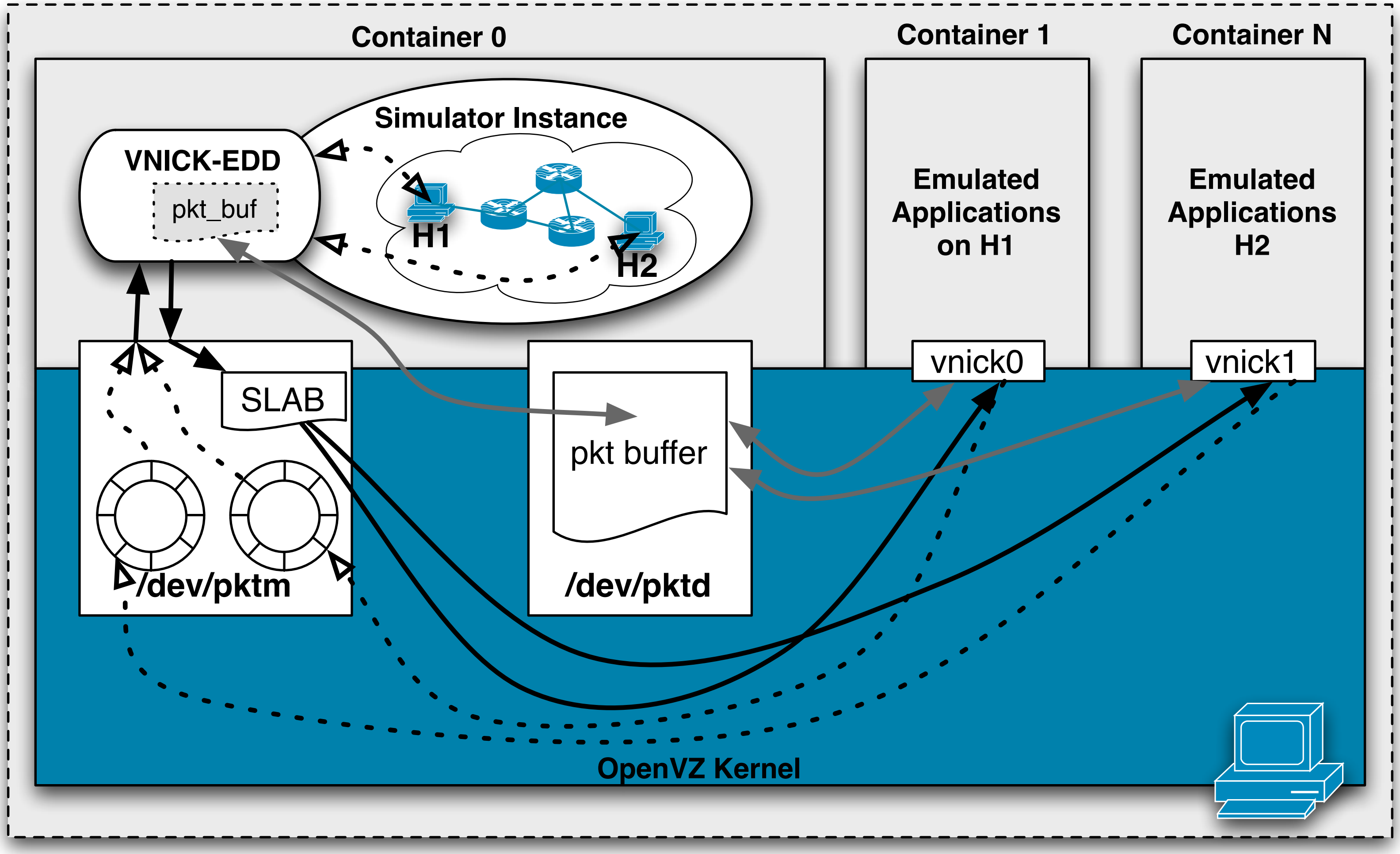
Remote Connections: OpenVPN



Local Connections: Bridging

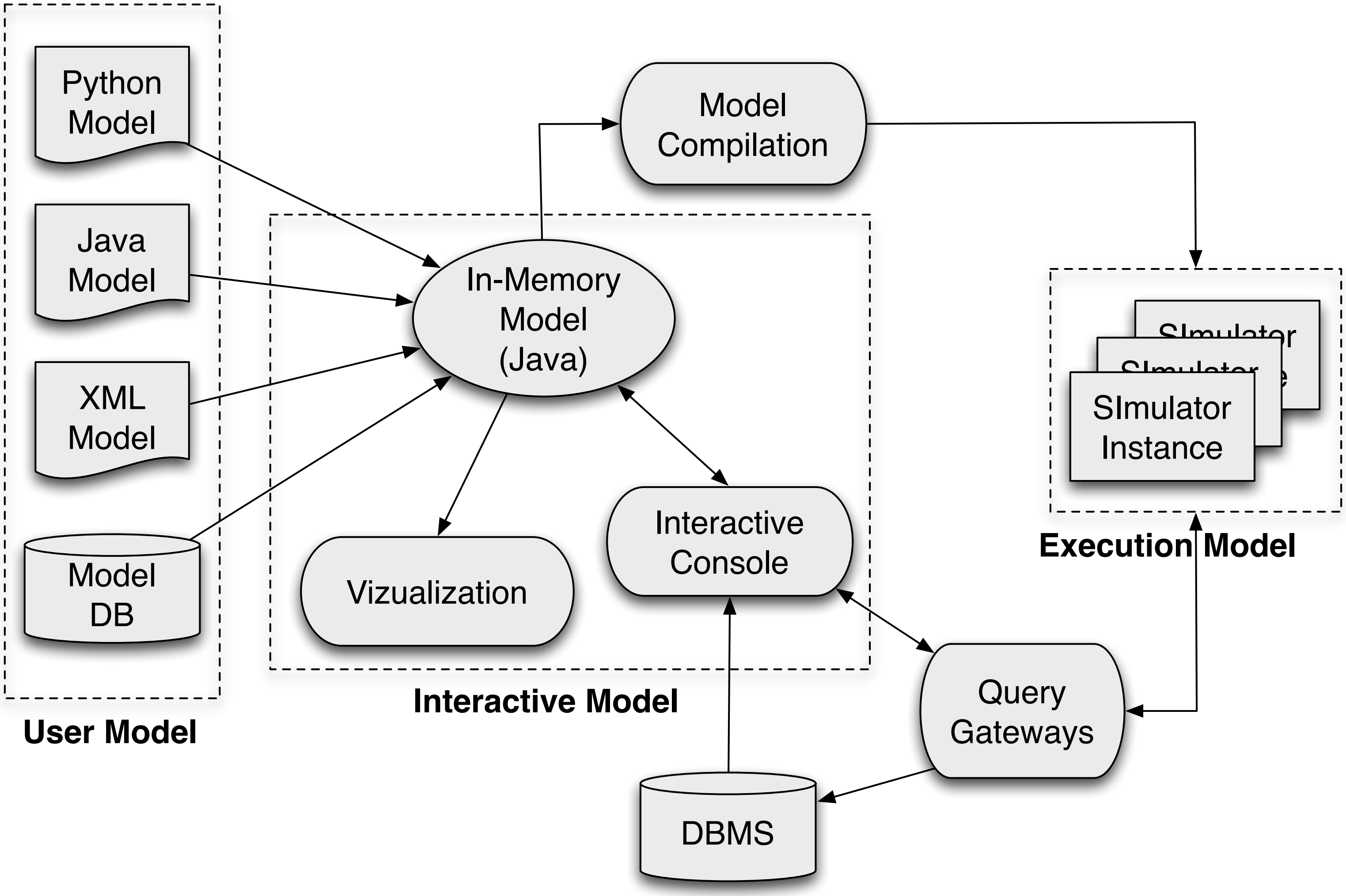


Local Connections: Kernel Bypass

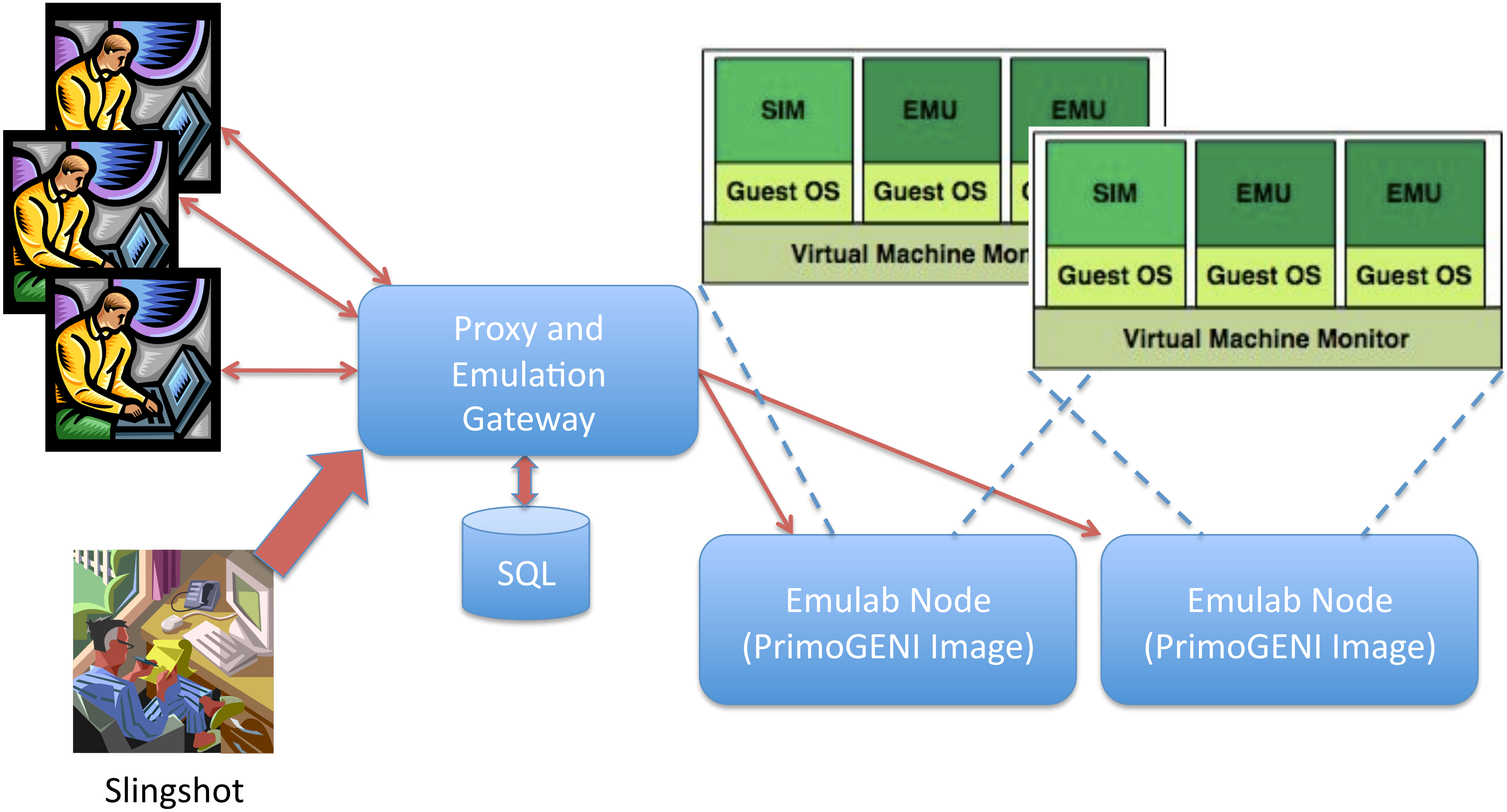


Joint work with Yan Luo and Craig Maisley (UML) and Raju Rangaswami (FIU)

PrimoGENI Experiment onto ProtoGENI cluster



Onto ProtoGENI Cluster



Proxy and Emulation Gateway

SQL

Emulab Node (PrimoGENI Image)

Emulab Node (PrimoGENI Image)

Slingshot

Demonstration