

# **Lab1: Introduction to the Wireless Emulator**

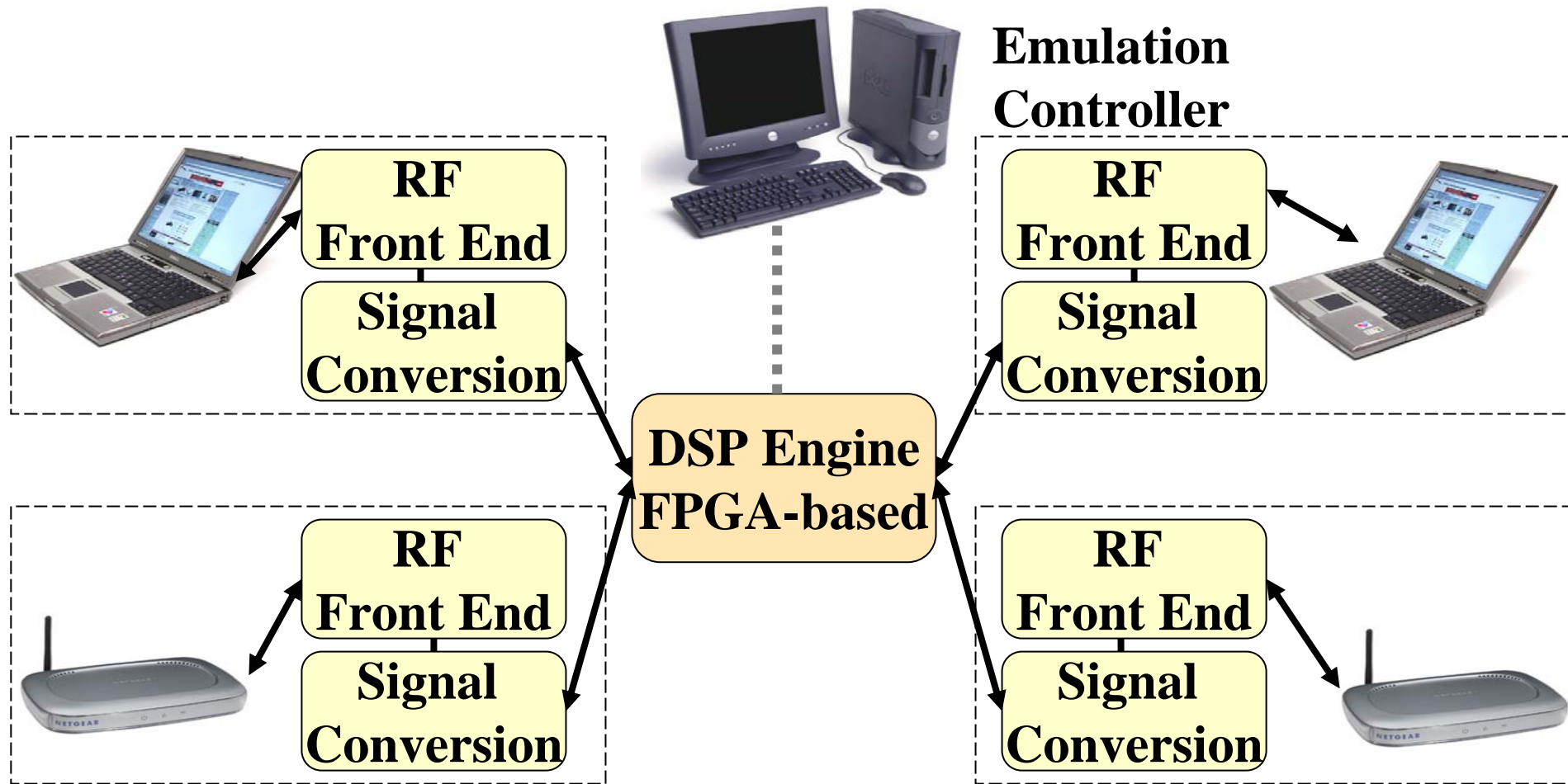
**Yang Su**

Laboratory for Software Technology  
ETH Zurich

# Outline

- **CMU wireless emulation testbed**
- Programming the emulator

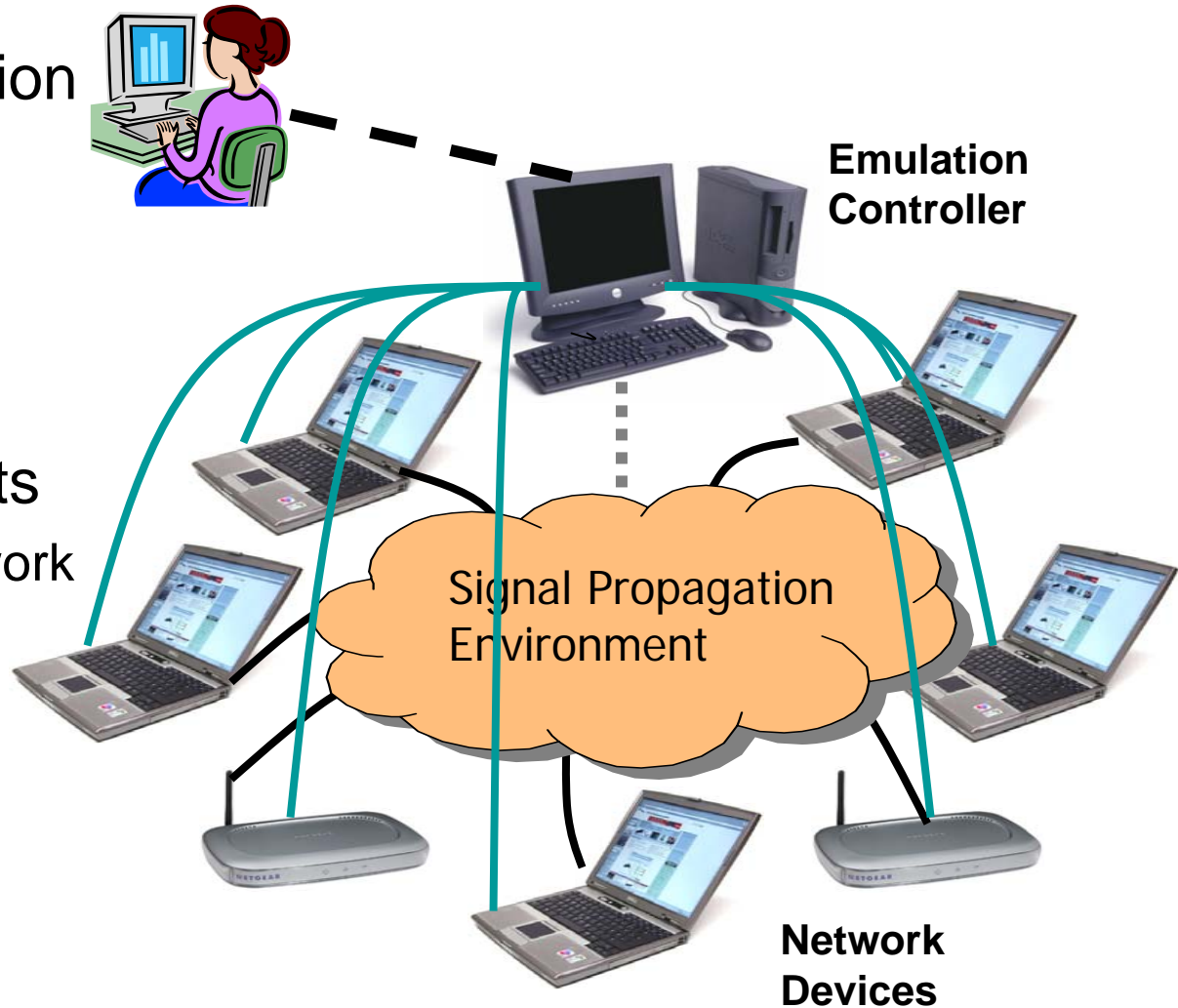
# Hardware Architecture



Network devices and signal conversion modules reside in shielded chassis.

# Typical Usage

- ssh access to emulation controller
- Timesharing
  - Entire emulator
  - Subset of machines
- Control of experiments
  - Via private wired network
  - Interactive
  - XML scripts
  - Java



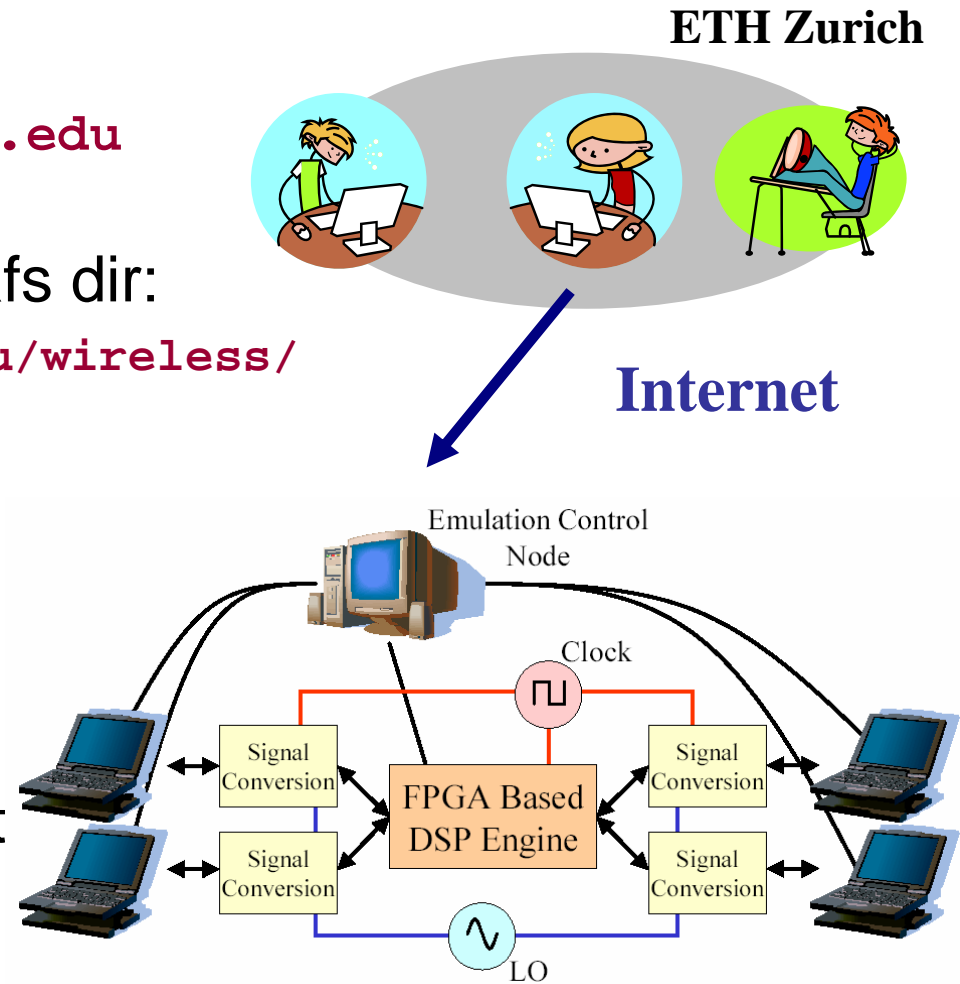
# Reservation

- Multiple users share the testbed but only one user at one time can run experiment
- Make reservation each time before using the emulator
- Reservation is served FCFS
- To be efficient, we need to cooperate to each other:
  - No over-booking
  - If finish earlier, release the remaining reservation

<http://gold.aura.cs.cmu.edu:8080/emulator/Testbed.Login>

# Remote Access

- Emulation control node:
  - `emucontrol-1.ece.cmu.edu`
- Find your account in my afs dir:
  - `/afs/ethz.ch/user/y/ysu/wireless/`
- In your time slot, you own the whole testbed
- Rest of the time, you can login the control node, but have no access to the emulator and emulator nodes → **take the time to debug your program**



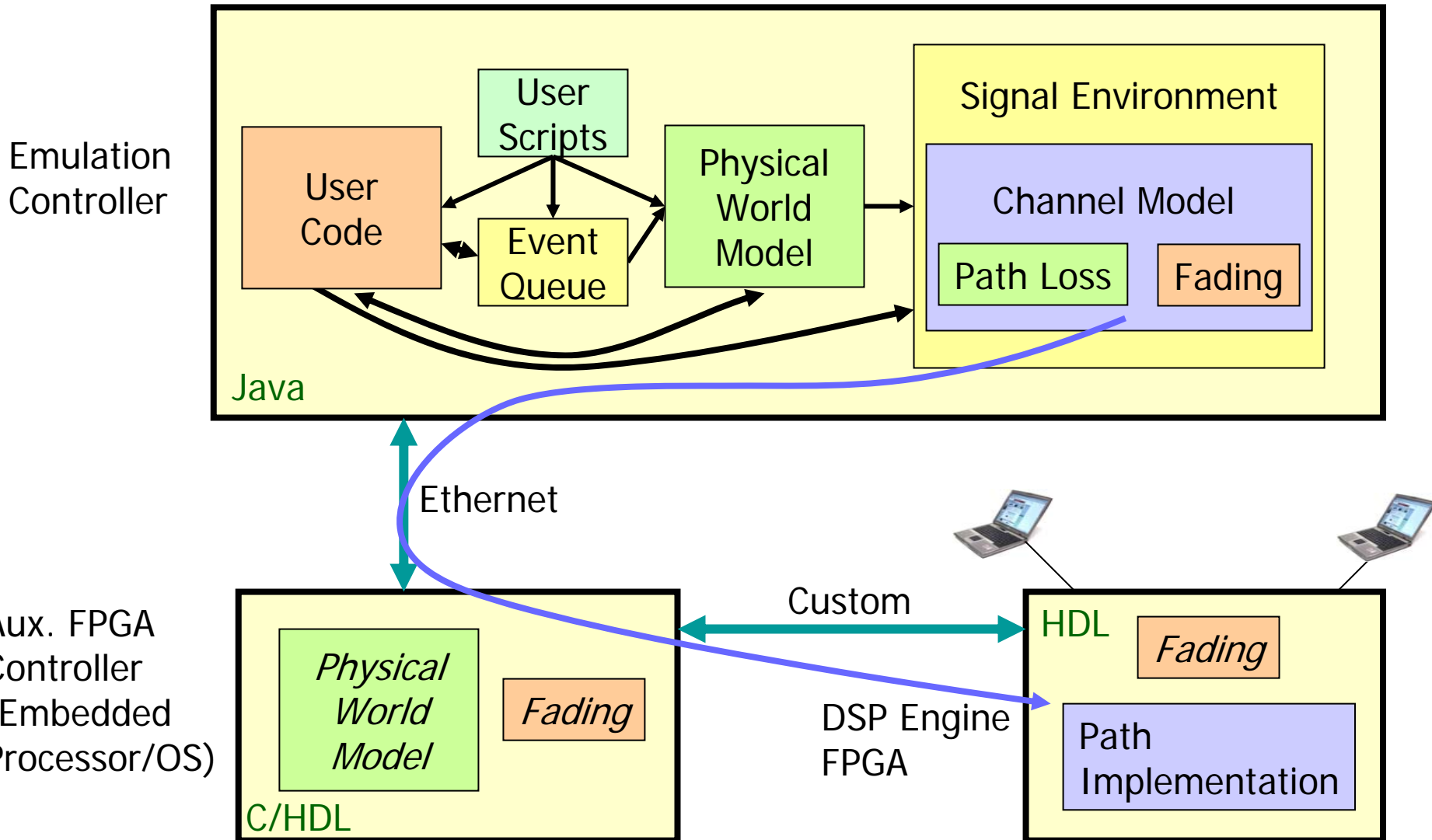
# Demo

- How to reserve a time slot
- Login the server and explore the working environment

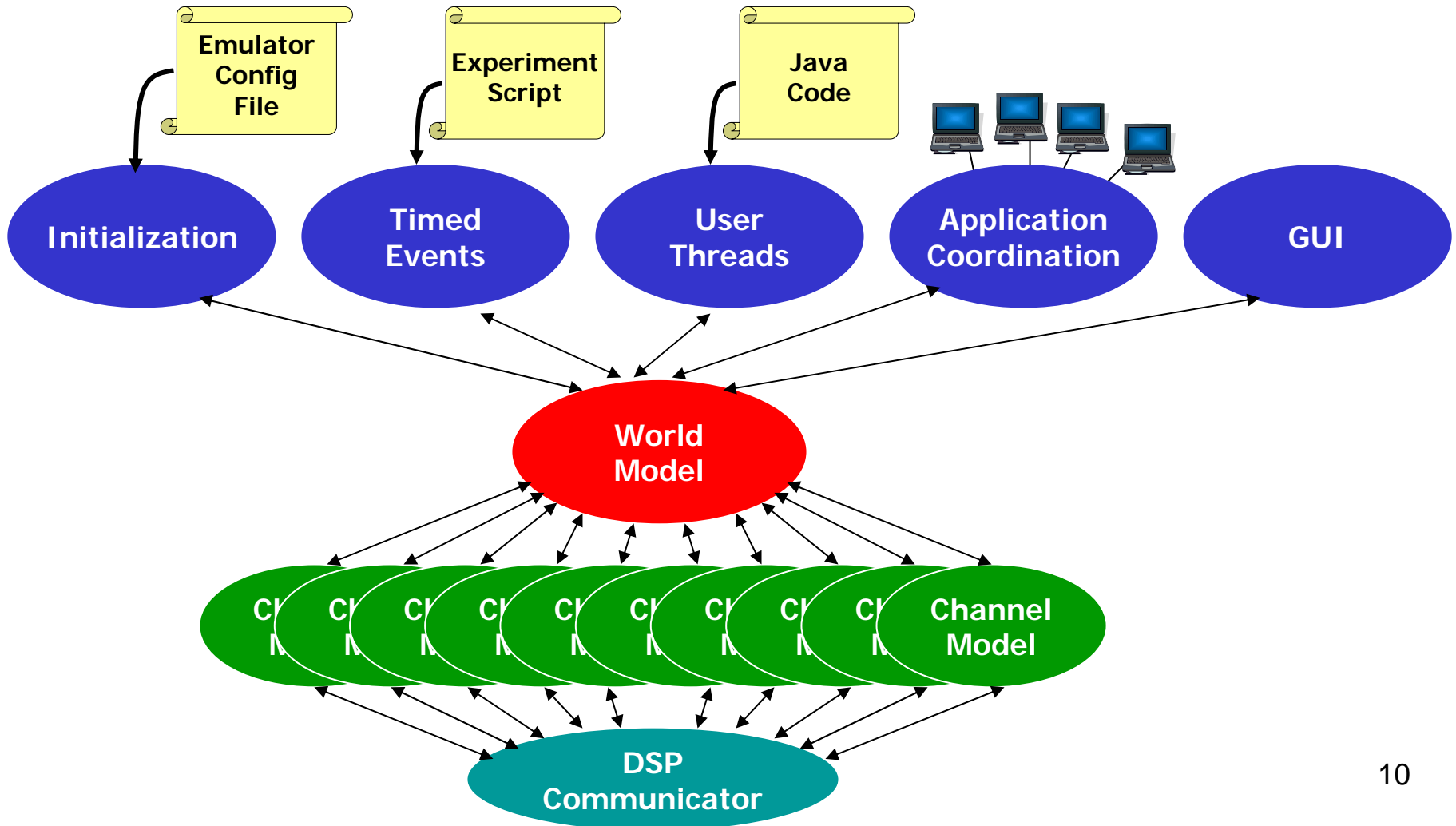
# Outline

- CMU wireless emulation testbed
- **Programming the emulator**

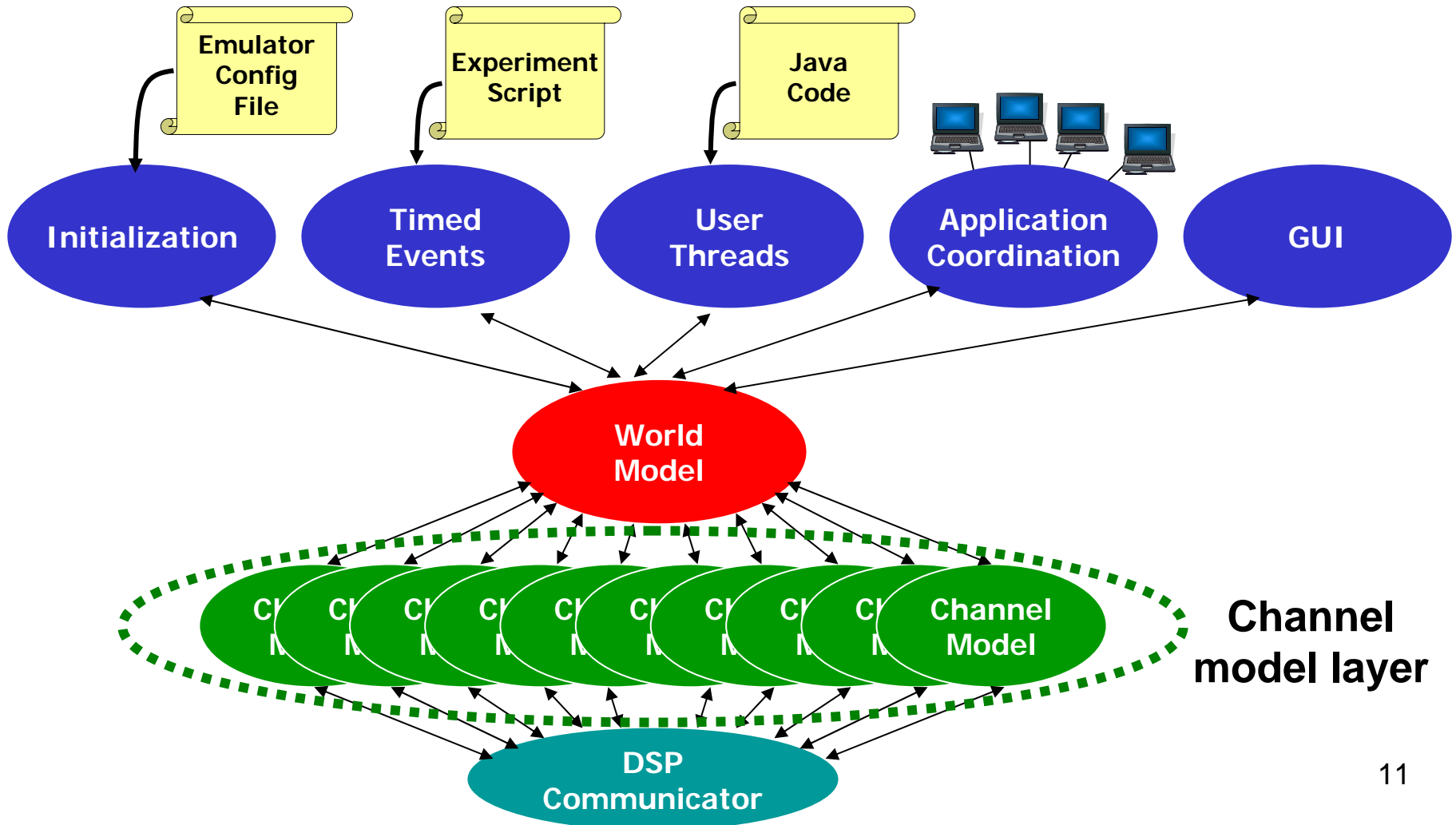
# Software Architecture



# Software Architecture: Layered-view



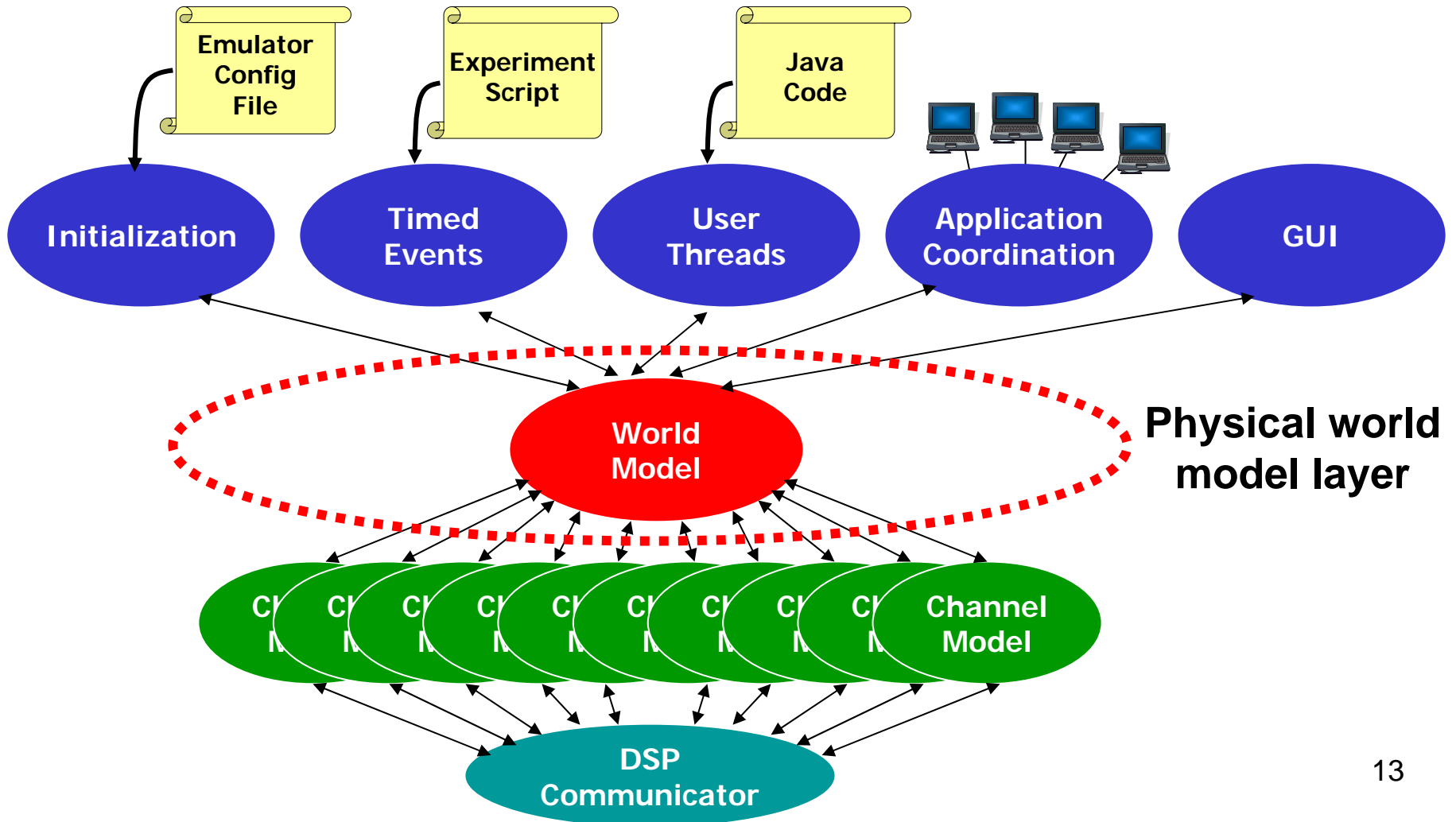
# Software Architecture: Layered-view



# Channel Model

- Model the signal propagation channels between the emulator nodes
  - Path-based
  - Large-scale path loss
  - Small-scale fading
  - Propagation delay

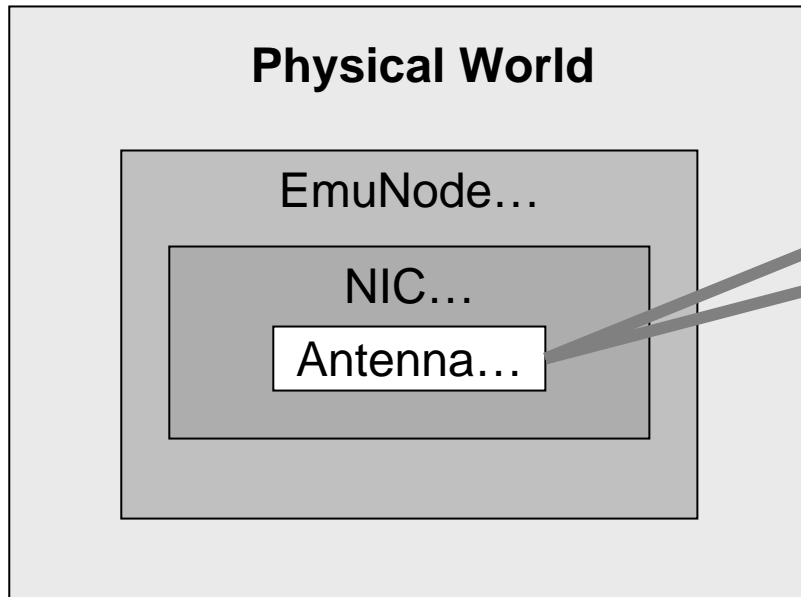
# Software Architecture: Layered-view



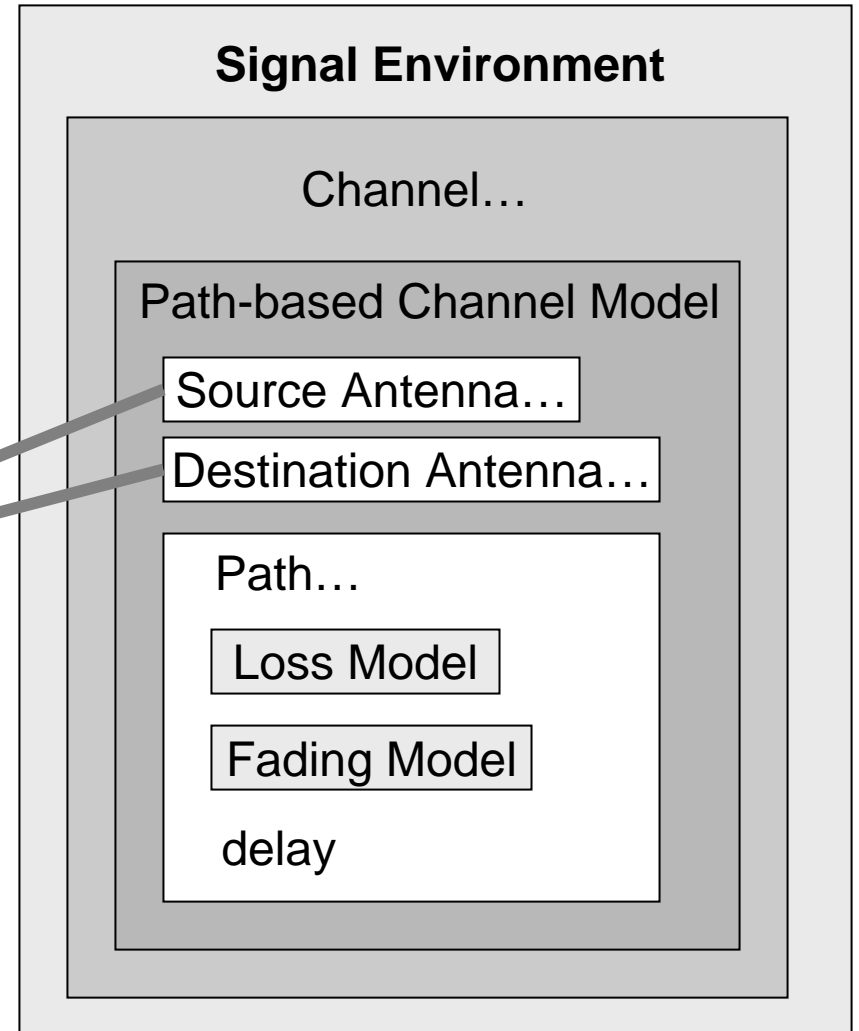
# Physical World Model

- Keep track of the properties of the nodes and the world they move in
  - Location of antennas
  - Direction of antennas
  - Speed and direction of movement

# Physical world vs. Channel

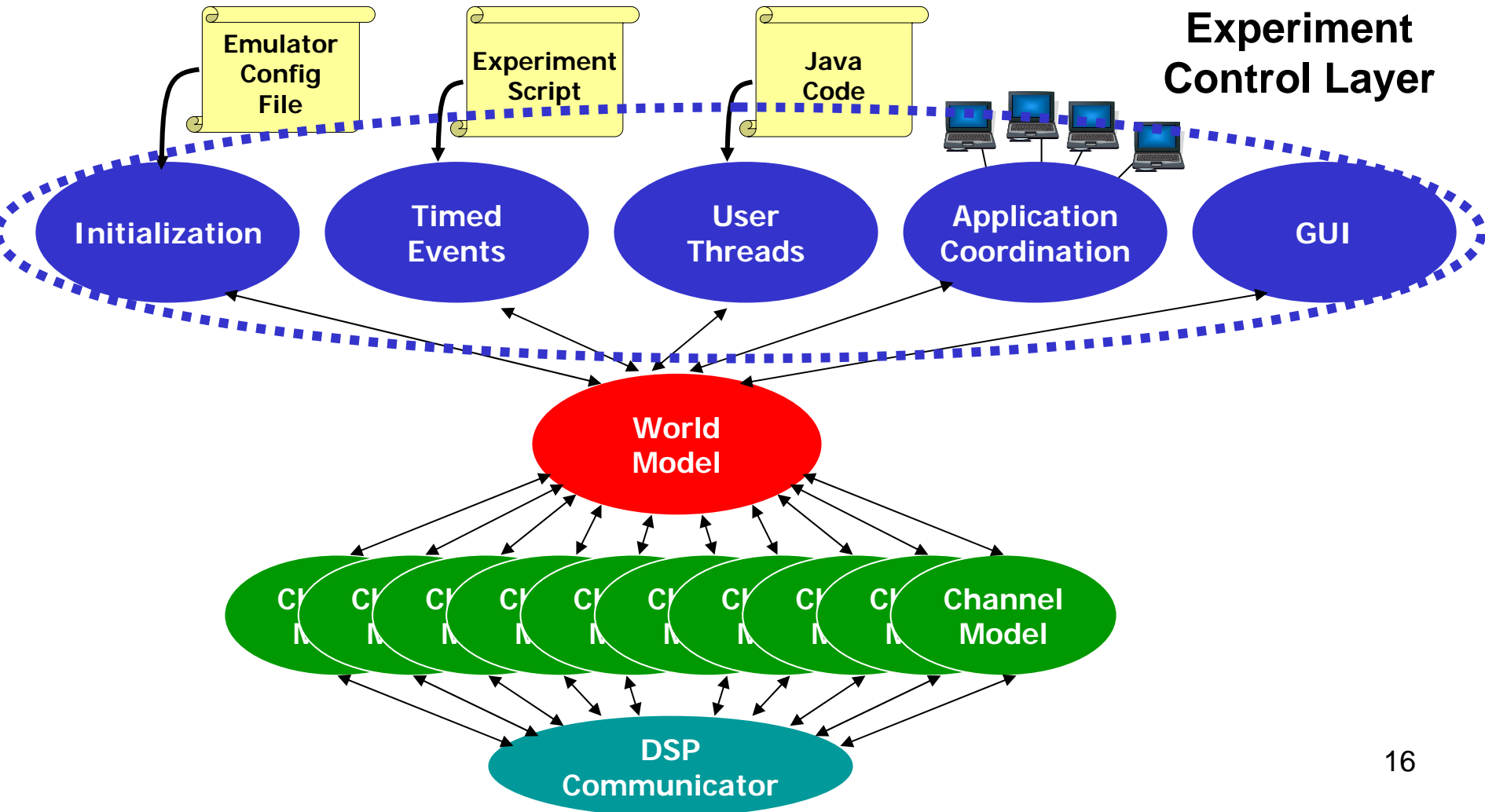


a)



b)

# Software Architecture: Layered-view



# Experiment Control

- Interpret user's experiment description
- Control the emulator by changing the state of the physical world model and channel model on the fly
- Different ways for users to describe/control the experiments
  - GUI
  - XML script
  - Java program

# Experiment Control

- Experiment initialization
  - Reads in the emulator configuration file and initializes the emulator hardware and software
- Timed events
  - Maintains a queue of events that change the state of the physical world and of the nodes
  - Events are ordered by time and executed at the appropriate time
  - For simple experiments, events are read from a script file or generated by the user through a GUI
  - For more complex experiments, “power users” can write Java code to generate timed events directly

# Experiment Control

- Application coordination
  - Control applications on the emulator nodes
  - Applications can be traffic generators (iperf, netperf) or real applications (e.g. Web clients and servers)
- User threads
  - Execute the java code
- GUI
  - Fast interactive exploration of design space

# Experiment Script

- Channel configuration
- Emulator nodes configuration
- Mobility specification
- GUI
- Java program
- Events

# How to run applications

- Three approaches to run applications in your experiment
  - XML script: **Exec** event
  - Run applications manually from emulator nodes
  - Use Java framework
- Today, we only look at the first two

# Run application with script

```
<EventDef>
  ...
  <EventGroup time="5.0" concurrent="false">
    <Exec>
      <node>emu-5</node>
      <cmd>iwconfig ath0</cmd>
      <log>
        <location>central</location>
        <stdoutFile>test.stdout.txt</stdoutFile>
        <stderrFile>test.stderr.txt</stderrFile>
      </log>
      <asynchronous>true</asynchronous>
      <exitOnError>false</exitOnError>
    <Exec/>
  </EventGroup>
  ...
</EventDef>
```

# Run application manually

- Scripts are difficult to debug
- Keep them simple
- Use XML scripts/GUI to define the channel model, physical world and mobility
- Login into emulator nodes to run applications manually

# Emulator Nodes

- Laptops with linux
- Ethernet subnet: 192.168.10.xxx
  - who's on this subnet  
`ping -b 192.168.10.255`
- Access control via `sudo`
- Madwifi driver for wireless NIC
  - Load the driver  
`sudo modprobe ath_pci`
  - Destroy the current virtual wireless interface  
`sudo wlanconfig ath0 destroy`
  - Create a virtual wireless interface in ad hoc mode  
`sudo wlanconfig ath create wlandev wifi0 mode ahdemo`
  - Configure the wireless interface: `iwconfig`

# Demo

- Demo1: GUI
- Demo2: run application manually
- Demo3: run application from XML script

# Reading

- **A software Architecture for Physical Layer Wireless Network Emulation**, Glenn Judd and Peter Steenkiste, WiNTECH 2006
- **Using Emulation to Understand and Improve Wireless Networks and Applications**, Glenn Judd and Peter Steenkiste, NSDI 2005