Ns-2 Primer

Presented by
Siva Dasaraju
Computer Science
WMU

Ns-2, the network simulator

- A discrete event simulator
  - Simple model
- Focused on modeling network protocols
  - Wired, Wireless, satellite
  - TCP, UDP, multicast, unicast
  - Web, Telnet, Ftp
  - Ad hoc routing, sensor networks

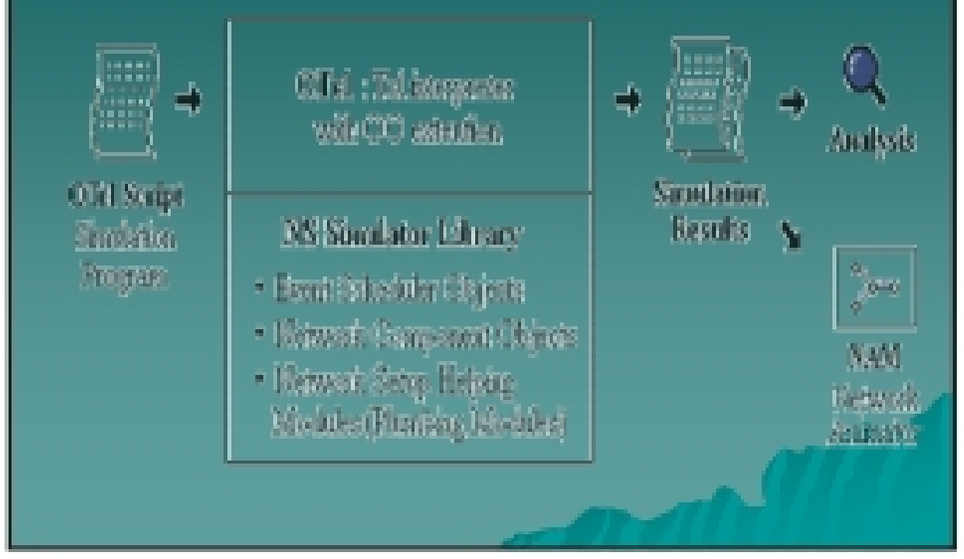
Discrete event Simulation

- Model world as events
  - Simulator has list of events (event-driven)
  - Process: take each one, run it, until done
  - Each event takes some amount of time, simulated/actual
  - Uses single thread of control (only one event in execution at a given time)

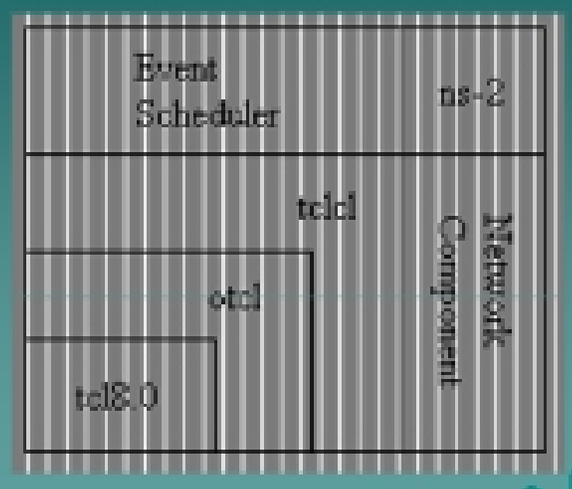
NS Architecture

- Object Oriented (C++, OTcl)
- Data/Control Separation
  - C++ for data
    - Manipulate bytes, packet headers, implementation algorithms
    - Run time speed important than turn-around time(run simulation, find bug, fix bug, recompile, re-run)
    - Suits detailed protocol implementation
  - OTcl for control
    - Simulation scenario configurations
    - Manipulating existing C++ objects
    - Fast to write and change
    - Run time less important (since configuration happens once)
- OTcl/C++ share a class hierarchy, and TclCL can be used to share functions, variables

Simplified User's View



Architectural View



Basic Tcl

variables:

```
set x 10
puts "x is $x"
```

functions and expressions:

```
set x 10
set y [Fact x]
set y [expr x*x]
```

control flow:

```
if {$x > 0} { return $x } else {
    return [expr -$x] }
for {set k 0} {$k<10} {incr k} {
    puts stdout $k
}
```

procedures:

```
proc Fact {x} {
    set i 1; set product 1
    while {$i <= $x} {
    set product [expr $product * $i]
    incr i
}
return $product
}
Factorial 10
```

Also lists, associative arrays, etc.

Basic OTcl

Class Person

```
# constructor:
Person instproc init {age} {
    $self instvar age_
    set age_ $age
}
# method:
Person instproc greet {} {
    $self instvar age_
    puts "$age_ years old: How are
    you doing?"
}
```

 # subclass:

```
Class Kid -superclass Person
Kid instproc greet {} {
    $self instvar age_
    puts "$age_ years old kid:
    What's up, dude?"
}

set a [new Person 45]
set b [new Kid 15]
$a greet
$b greet
```

NS Programming

- Create the event scheduler
- Turn on tracing
- Create network
- Setup routing
- Insert errors
- Create transport connection
- Create traffic
- Transmit application-level data

Creating Event Scheduler

- Create event scheduler
  set ns [new Simulator]
- Schedule events
  $ns at <time> <event>
    - <event>: any legitimate ns/tcl commands
  $ns at 5.0 "finish"
- Start scheduler
  $ns run

Hello World

```
#simple.tcl
    set ns [new Simulator]
    $ns at 1 "puts \"Hello
      World!\""
    $ns at 1.5 "exit"
    $ns run
% ns simple.tcl
Hello World!
%
```

Tracing

- Packet tracing (packet format)
  - On all links
  - On particular links
- Event tracing
  - Record events in trace file
- Queue monitor
- Flow monitor
- Nam