Discussing Discrete Events Simulation.

The *ns-2* case.

Andrés ARCIA
TELECOM Institut ; TELECOM Bretagne
Département Réseau Sécurité et Multimédia - RSM Department

November 2008

Let's start by defining a Discrete Event Simulation:

A **Discrete Event Simulation** is used to represent a system's behavior through a series of time-events. So,

- A **system** is a number of inter-related objects.
- **Objects** are self-contained entities with attributes used to represent different characteristics of the system.
- A **time-event** is a discrete point in time capable of instantaneously changing state variables.

An example of a Discrete System is:

A Bank Serving Customers: The customers wait in line to be served by a bank-teller at time t_0 and the bank-teller process his requirement and finishes at time t_1. *Between t_0 and t_1 the system has not changed.*

Simulation is useful to:

- Identify crucial variables of the system and its interactions
- Experiment with new scenarios at low cost
- Providing controllable situations for the system under study
- Allowing playing with time: compress or expand it

And we don’t like it because:

- It provides statistical estimates and **not exact** characteristics of the system.
- All results depend on the system’s model (*no matter what effort you’ve made...*)

Characteristics of a Simulation

- At the beginning of the simulation $t = 0$
- Requires a time keeping mechanism to manage time (clock, list of events, etc.)
- The clock advances to the next in-order event, so times advances from one event to another
- There must be a stopping condition to end the simulation

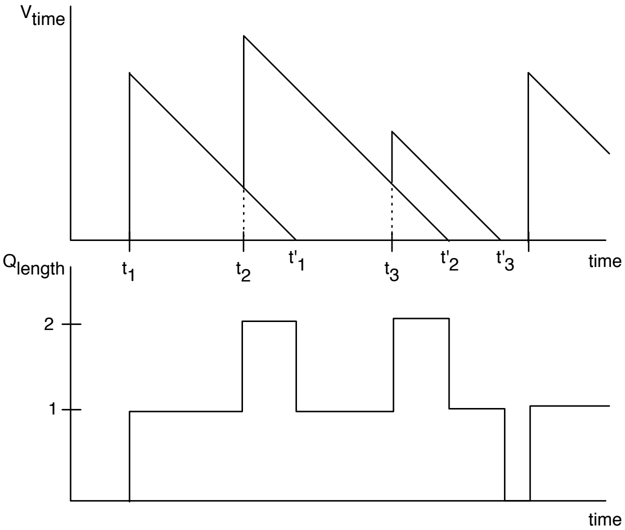
Let's see the next-event time advance approach in a packet switched network's node.

Let's say K is a packet that should be forwarded by a node:

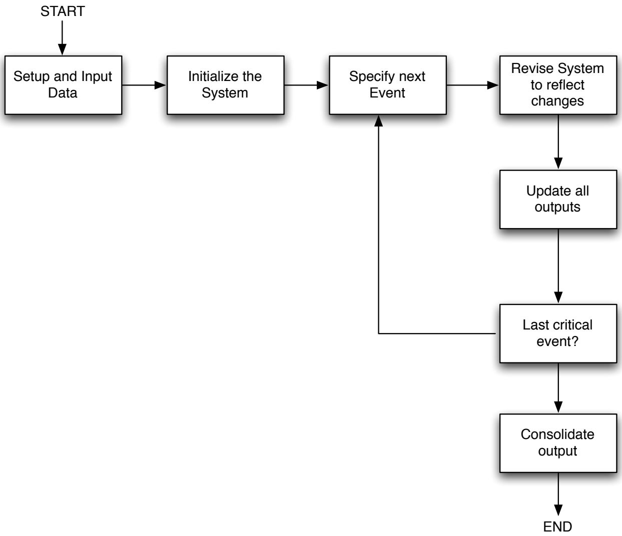
- t_K is the event arrival time into the sub-system (the node)
- $A_K = t_K - t_{K-1}$ the interval separating two packet arrivals
- S_K is the service time of the packet K with size T_K for a node serving exactly one packet at a time
- D_K the delay of the packet K in the system equal to the **time in queue** plus the **service time**.

so, $t'_K = t_K + D_K$ is the completion time.

Packets Processing in a Node



To summarize the time management:



A case of study: ns-2

- A discret event simulator that models:
    - packets, links, queues, protocols
    - has a simulation visualizer (NAM, network animator)
    - trace can be played back
    - extensive error model
- evolved since 1989, REAL by Keshav, then 1995 ns by Floyd et al. at International CS Inst (Berkley, California).
- ns-2 is a pretty much stable simulator; current version 2.33.

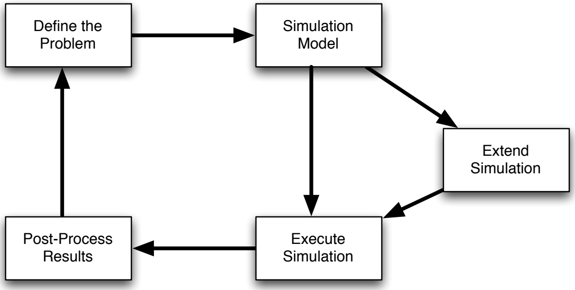
Components of ns-2

- To deal *ns-2*
  - Specify the simulation, then generate traces
  - Relies on: TCL/TK, Otcl and TclCl
- nam, the network animator
  - animates traces from simulation
  - GUI to create simple topologies
- To prepare before simulation: topology and traffic
- To process after simulation: traces with awk, perl, etc.

Status at 2008

- For the ns-2.33
  - aprx. 310000 LOC C/C++ and aprx. 167000 LOC Tcl/OTcl
  - aprx. 120 test suites + examples inside
  - an up-to-date ns manual
  - a **new book**: Introduction to Network Simulator NS2 (11/08)
  - The best thing: *top-notch feedback!*
- platform
  - MACOS, FreeBSD, Linux, Solaris, Windows
- widely used in the research community
- active discussion list and pretty fast-reacting community

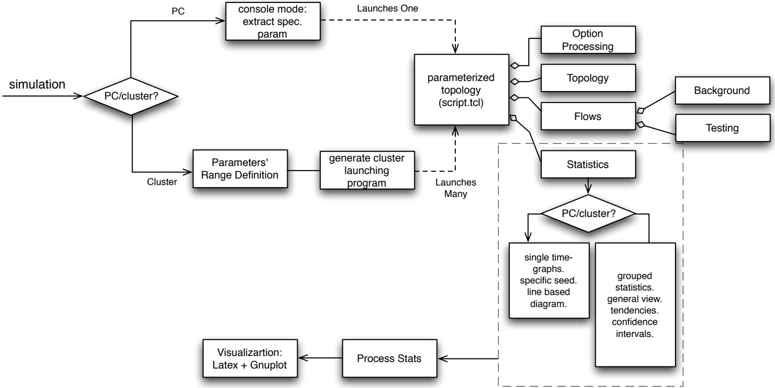
Development Model for ns-2



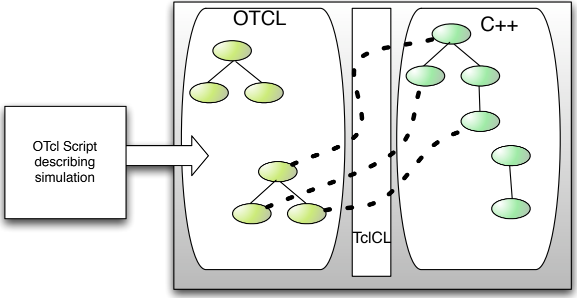
What to do in each stage:

- Create simulation
  - describe network, protocols, sources, sinks
  - specify in OTCL that controls the C++ core
- Execute Simulation
  - Simulator have a list of events (including packets), executes next event in time, until explicit stop
  - Events happens in virtual time that takes arbitrarily long real time.
  - Single thread control
- Post processing, some nice efforts
  - RPI graphics and statistic package
  - **TCP-LAB** to automate TCP scenarios and statistic collection (an extension of RPI)

Large Scale Simulation with *TCP-LAB*


simulation
PC/cluster?
PC
console mode: extract spec. param
Launches One
Cluster
Parameters' Range Definition
generate cluster launching program
Launches Many
parameterized topology (script.tcl)
Option Processing
Topology
Flows
Background
Testing
Statistics
PC/cluster?
single time-graphs. specific seed. line based diagram.
grouped statistics. general view. tendencies. confidence intervals.
Process Stats
Visualizartion: Latex + Gnuplot

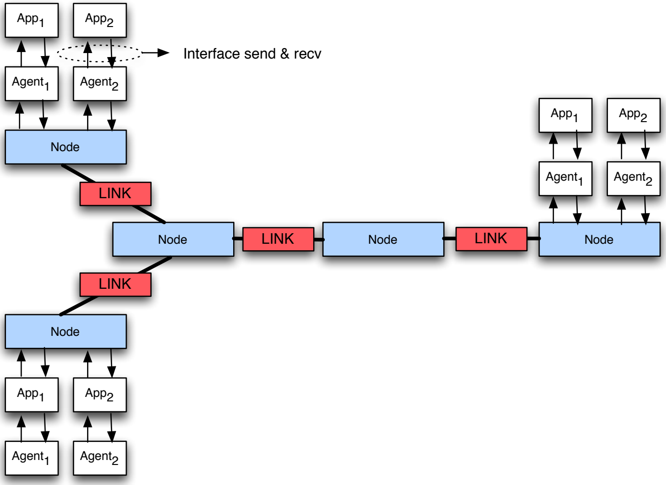
TCL/C++ model for ns-2



C++ vs. OTcl

- C++
  - packet processing and protocol implementation
  - efficient code, fast and highly debuggable
- TCL
  - Topology specification
  - Scheduleable actions: tracing, modifying behaviour
  - Resetting C++ parallel system
  - Easy the experimental process: change parameters and relaunch.
  - Allows simple parallelization of simulations.

Nodes, Links and Apps: what the scheduler sees!

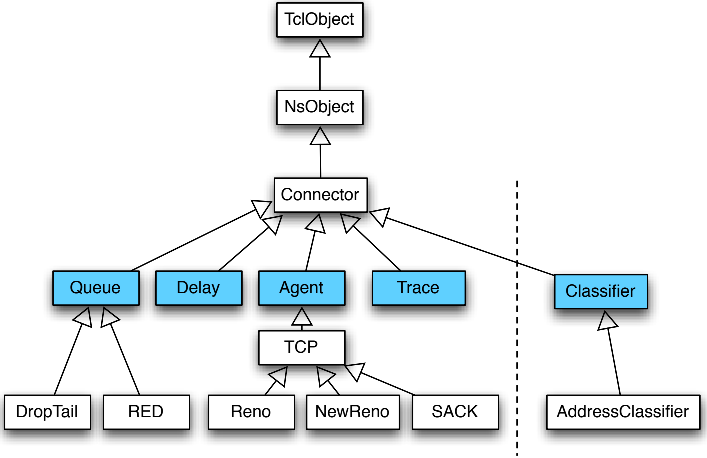


What's available nowadays in NS-2.33

Remember the scheduler is the **main responsible** for the simulation duration.

- **Simple** List Scheduler
  - Add Event → $O(N)$
  - Modify Event → $O(N)$
  - Consume/Delete Event → $O(1)$
- **Heap** Scheduler
  - Add Event → $O(1)$
  - Modify Event → $O(\log N)$
  - Consume Event → $O(1)$
  - Delete Event → $O(\log N)$
- **Calendar Queue** Scheduler
  - Improved heap that supports well the scale.

Simplified Class Hierarchy



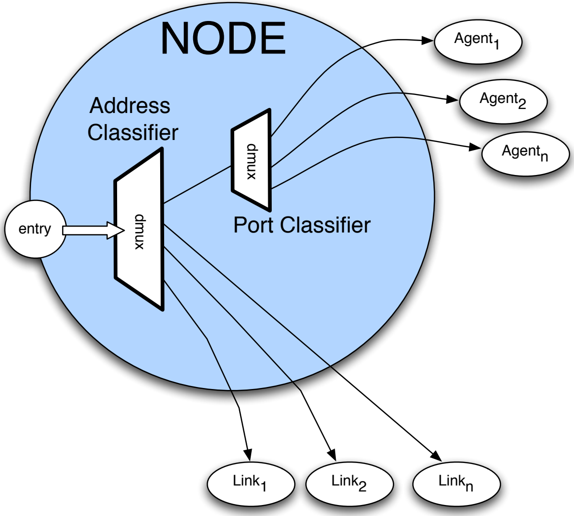
Mayor Types

- **Applications** → Communication trigger, passive receivers, traffic models.
- **Agents** → Packet consumer and generators (i.e. TCP :))
- **Nodes** → Addressable entity
- **Link** → Set of queues

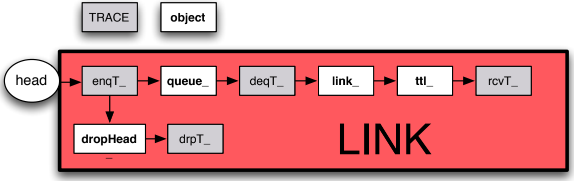
Inner Types

- Classifier
  - Table of n slots each pointing to a TclObject
  - `classify():identifies destination slot for a packet`
  - `AddressClassifier and PortClassifier found within Nodes`
- Connector
  - Receive packets and transmit to `target_`
  - Basis for Agents and Links (i.e., Queuing + Delay)

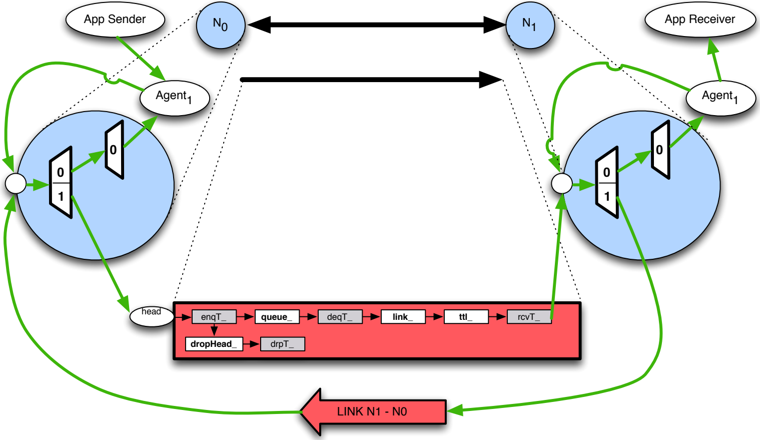
Simplified Node Architecture



Simplified Link Architecture



Simplified Topoology Architecture



Thank you.

Got Questions?