# Introduction to Simulation

## Adapted by Prof. Gheith Abandah

©2006 Raj Jain www.rajjain.com


- Simulation: Key Questions
- Introduction to Simulation
- Common Mistakes in Simulation
- Other Causes of Simulation Analysis Failure
- Checklist for Simulations
- Terminology
- Types of Models

©2006 Raj Jain www.rajjain.com

# Simulation: Key Questions

- What are the common mistakes in simulation and why most simulations fail?
- What language should be used for developing a simulation model?
- What are different types of simulations?
- How to schedule events in a simulation?
- How to verify and validate a model?
- How to determine that the simulation has reached a steady state?
- How long to run a simulation?

©2006 Raj Jain www.rajjain.com

# Common Mistakes in Simulation

1. Inappropriate Level of Detail:
   More detail $\Rightarrow$ More time $\Rightarrow$ More Bugs $\Rightarrow$ More CPU
   $\Rightarrow$ More parameters $\neq$ More accurate
2. Improper Language
   General purpose $\Rightarrow$ More portable, More efficient, More time
3. Unverified Models: Bugs
4. Invalid Models: Model vs. reality
5. Improperly Handled Initial Conditions
6. Too Short Simulations: Need confidence intervals

©2006 Raj Jain www.rajjain.com

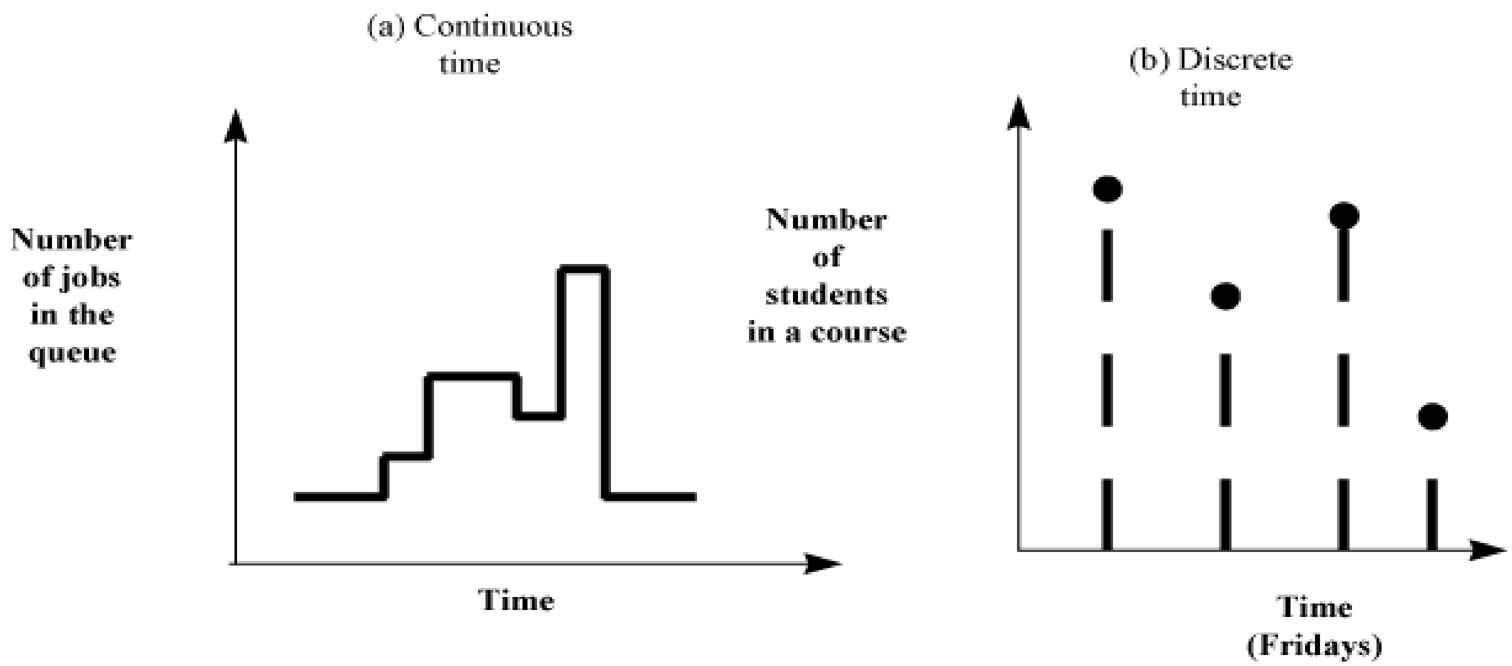
# Terminology

- **State Variables**: Define the state of the system
  Can restart simulation from state variables
  E.g., length of the job queue.
- **Event**: Change in the system state.
  E.g., arrival, beginning of a new execution, departure

©2006 Raj Jain www.rajjain.com

# Types of Models

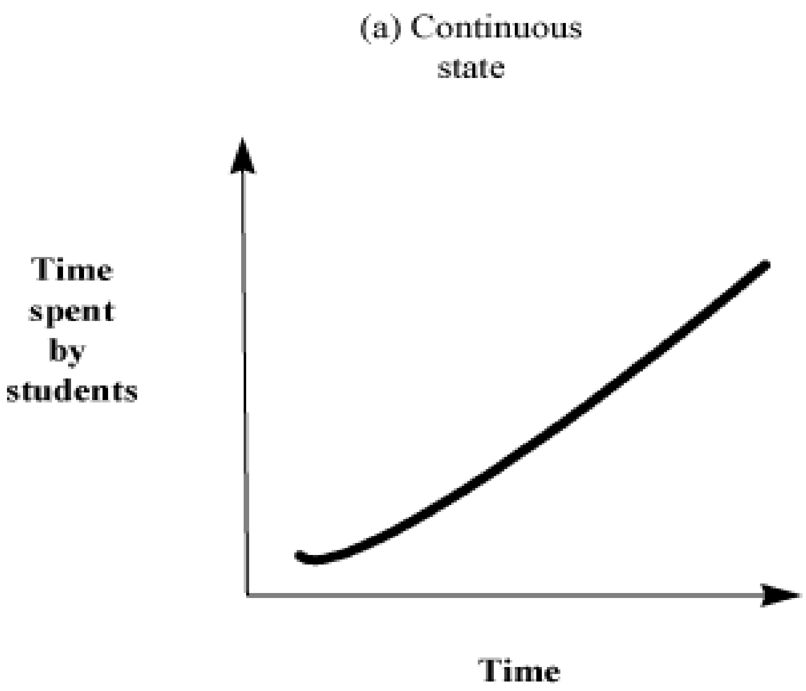
- **Continuous Time Model**: State is defined at all times
- **Discrete Time Models**: State is defined only at some instants

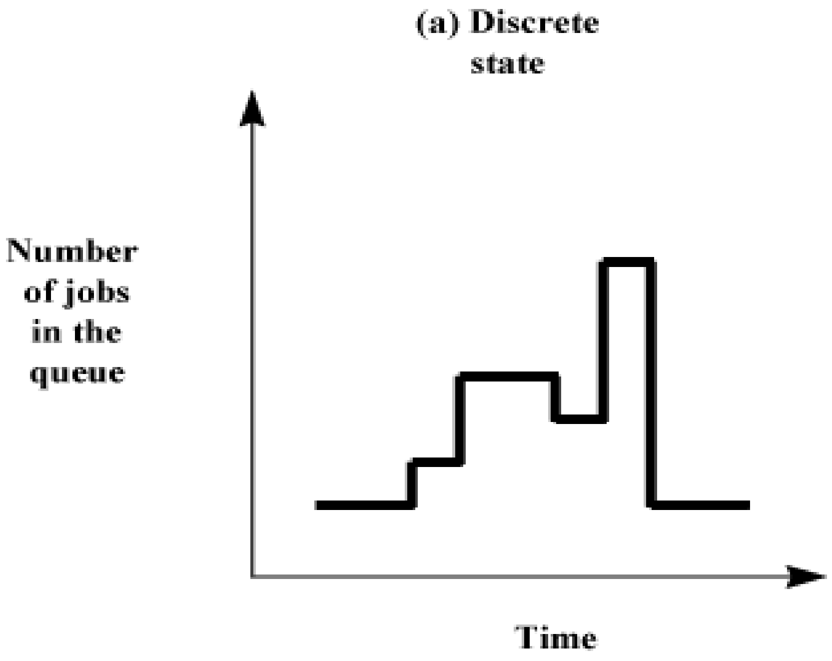



©2006 Raj Jain www.rajjain.com

# Types of Models (Cont)

- **Continuous State Model**: State variables are continuous
- **Discrete State Models**: State variables are discrete




©2006 Raj Jain www.rajjain.com

# Types of Models (Cont)

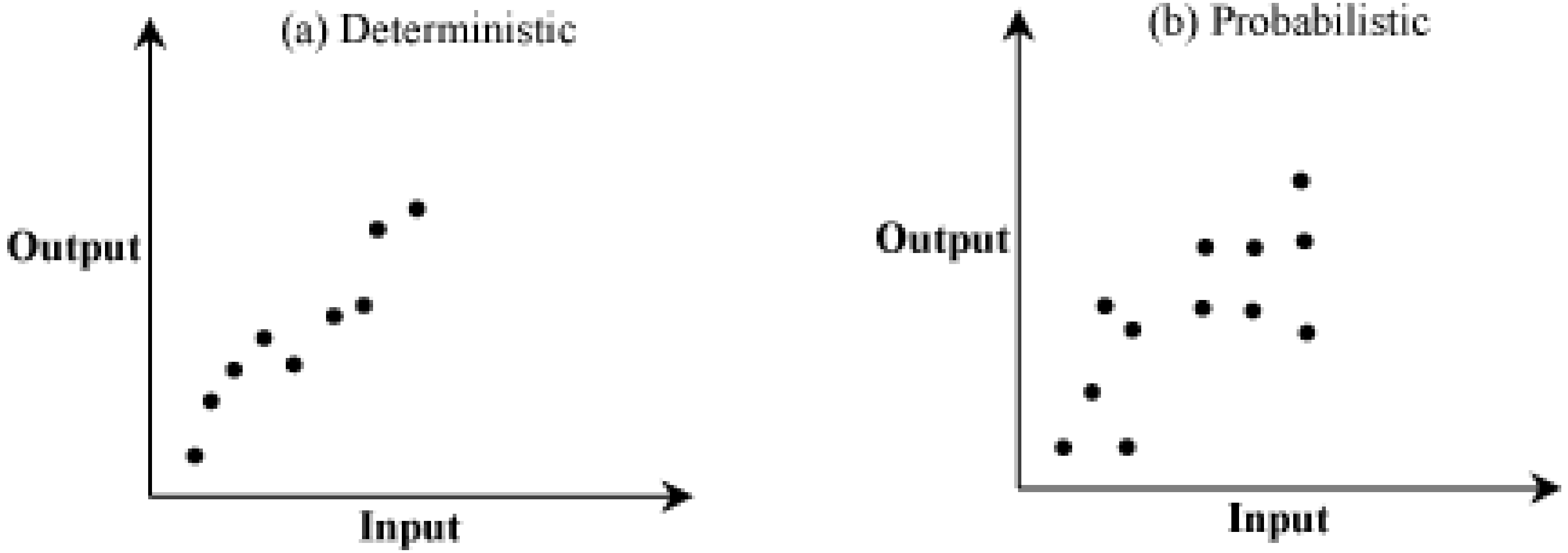
- Discrete state = Discrete event model
- Continuous state = Continuous event model
- Continuity of time ≠ Continuity of state

- Four possible combinations:
  1. discrete state/discrete time
  2. discrete state/continuous time
  3. continuous state/discrete time
  4. continuous state/continuous time models

©2006 Raj Jain www.rajjain.com

# Types of Models (Cont)

- **Deterministic and Probabilistic Models**:



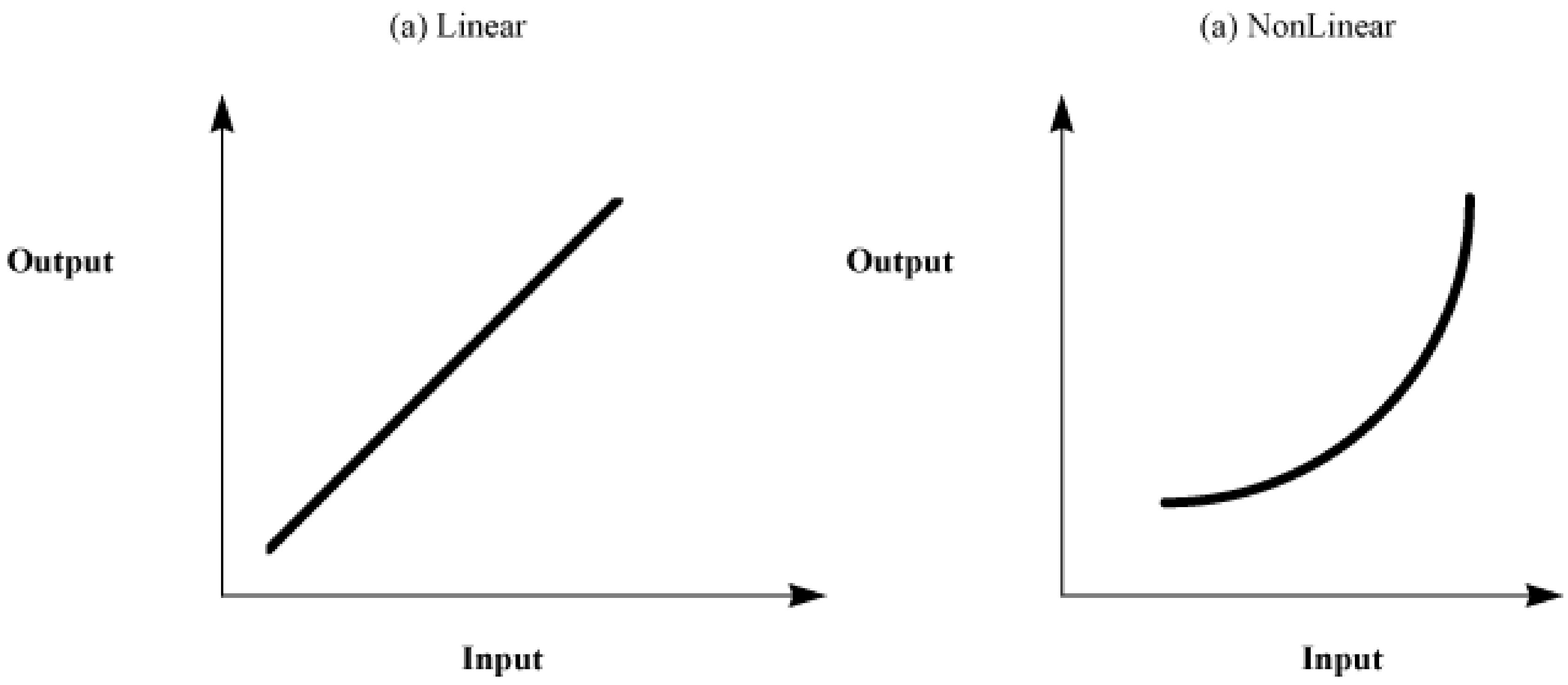

- **Static and Dynamic Models**:

CPU scheduling model vs. $E = mc^2$.

©2006 Raj Jain www.rajjain.com

# Linear and Nonlinear Models

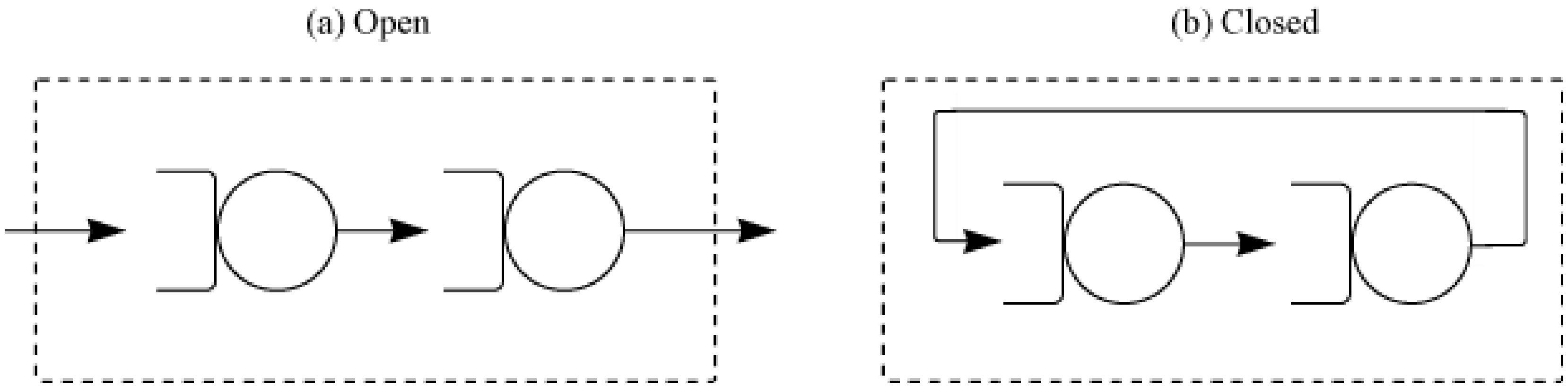
- Output = fn(Input)




©2006 Raj Jain www.rajjain.com

# Open and Closed Models

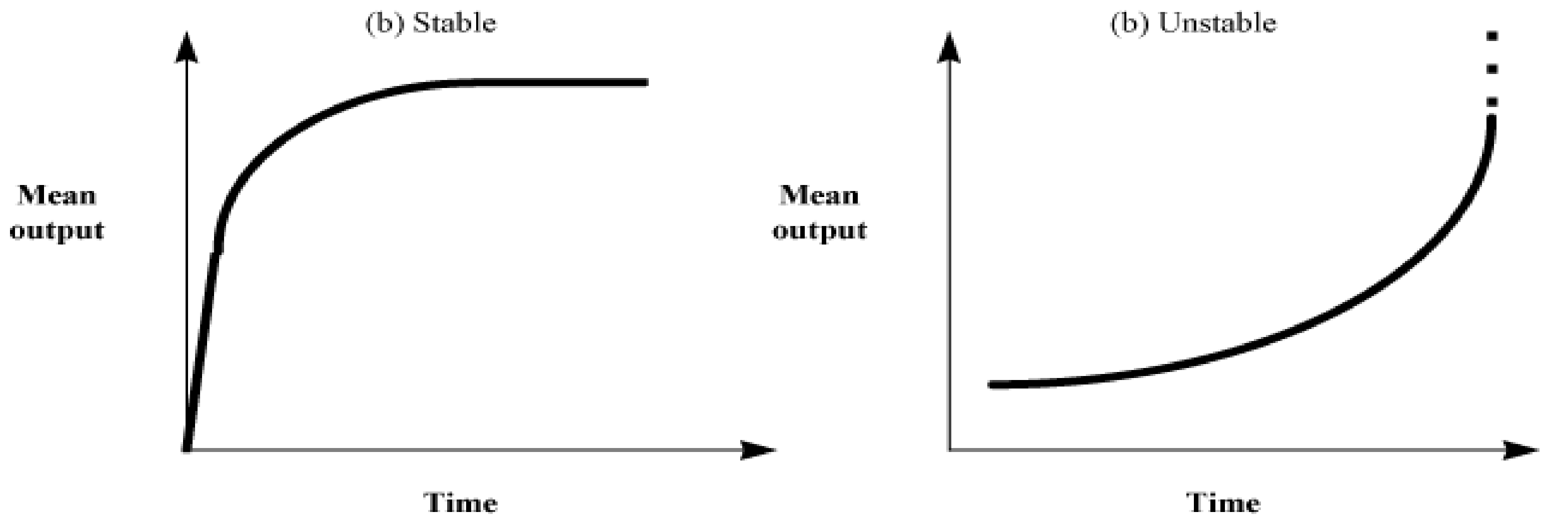
- External input ⇒ open




©2006 Raj Jain www.rajjain.com

# Stable and Unstable Models

- Stable ⇒ Settles to steady state
- Unstable ⇒ Continuously changing.




©2006 Raj Jain www.rajjain.com

# Computer System Models

- Continuous time
- Discrete state
- Probabilistic
- Dynamic
- Nonlinear
- Open or closed
- Stable or unstable

©2006 Raj Jain www.rajjain.com

# Selecting a Language for Simulation

1. Simulation language
2. General purpose
3. Extension of a general purpose language
4. Simulation package

©2006 Raj Jain www.rajjain.com

# Simulation Languages

- Save development time
- Built-in facilities for time advancing, event scheduling, entity manipulation, random variate generation, statistical data collection, and report generation
- More time for system specific issues
- Very readable modular code
- Verilog

©2006 Raj Jain www.rajjain.com

# General Purpose Language

- Analyst's familiarity
- Easy availability
- Quick startup
- Time for routines for event handling, random number generation
- Other Issues: Efficiency, flexibility, and portability
- Recommendation: Learn at least one simulation language.
- C++

©2006 Raj Jain www.rajjain.com

# Extensions of a General Purpose Language

- Examples: SimPy for Python and JSL for Java
- Collection of routines to handle simulation tasks
- Compromise for efficiency, flexibility, and portability.

©2006 Raj Jain www.rajjain.com

# Simulation Packages

- Example: NS2, SimpleScalar
- Input dialog
- Library of data structures, routines, and algorithms
- Big time savings
- Inflexible $\Rightarrow$ Simplification

©2006 Raj Jain www.rajjain.com

# Types of Simulations

1. Emulation: Using hardware or firmware
   E.g., Terminal emulator, processor emulator
   Mostly hardware design issues
2. Monte Carlo Simulation
3. Trace-Driven Simulation
4. Discrete Event Simulation

©2006 Raj Jain www.rajjain.com

# Types of Simulation (Cont)

Monte Carlo method *[Origin: after Count Montgomery de Carlo, Italian gambler and random-number generator (1792-1838).] A method of jazzing up the action in certain statistical and number-analytic environments by setting up a book and inviting bets on the outcome of a computation.*

- The Devil's DP Dictionary
McGraw Hill (1981)

©2006 Raj Jain www.rajjain.com

# Monte Carlo Simulation

- Static simulation (No time axis)
- To model probabilistic phenomenon
- Need pseudorandom numbers
- Used for evaluating non-probabilistic expressions using probabilistic methods.

©2006 Raj Jain www.rajjain.com

# Trace-Driven Simulation

- Trace = Time ordered record of events on a system
- Trace-driven simulation = Trace input
- Used in analyzing or tuning resource management algorithms, paging, cache analysis, CPU scheduling
- **Example**: Trace = Page reference patterns
- Should be independent of the system under study

  E.g., trace of pages fetched depends upon the working set size and page replacement policy

  - Not good for studying other page replacement policies
  - Better to use pages referenced

©2006 Raj Jain www.rajjain.com

# Advantages of Trace-Driven Simulations

1. Credibility
2. Easy Validation: Compare simulation with measured
3. Accurate Workload: Models correlation and interference
4. Detailed Trade-Offs:
   Detailed workload $\Rightarrow$ Can study small changes in algorithms
5. Less Randomness:
   Trace $\Rightarrow$ deterministic input $\Rightarrow$ Fewer repetitions
6. Fair Comparison: Better than random input
7. Similarity to the Actual Implementation:
   Trace-driven model is similar to the system

©2006 Raj Jain www.rajjain.com

# Disadvantages of Trace-Driven Simulations

1. Complexity: More detailed
2. Representativeness: Workload changes with time
3. Finiteness: Few minutes fill up a disk
4. Single Point of Validation: One trace = one point
5. Detail
6. Trade-Off: Difficult to change workload

©2006 Raj Jain www.rajjain.com

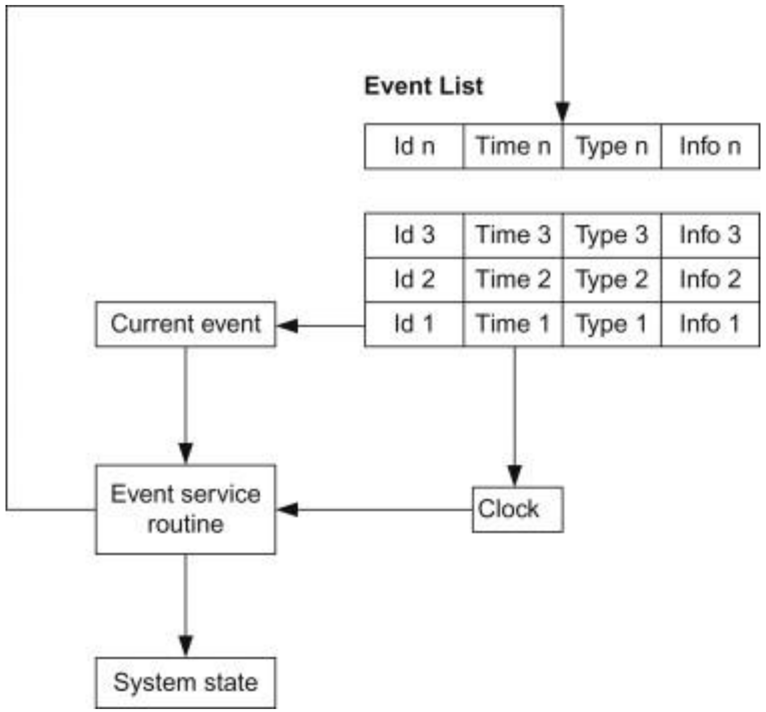
# Discrete Event Simulations

- Concentration of a chemical substance $\Rightarrow$ **C**ontinuous event simulations
- Number of jobs $\Rightarrow$ Discrete event
- Discrete state $\neq$ discrete time

©2006 Raj Jain www.rajjain.com

# Components of Discrete Event Simulations

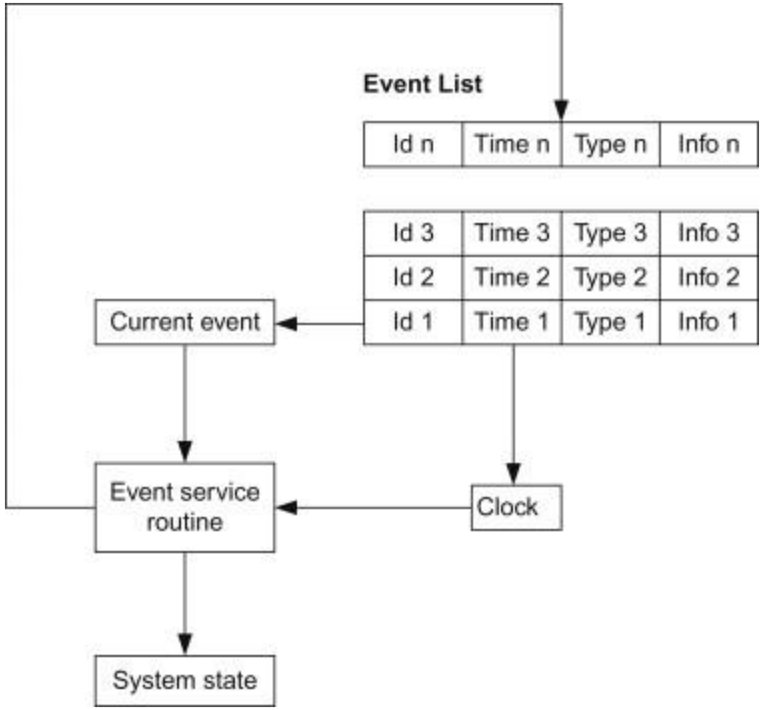
1. Event Scheduler
   (a) Schedule event X at time T.
   (b) Hold event X for a time interval dt.
   (c) Cancel a previously scheduled event X.
   (d) Hold event X indefinitely
   (e) Schedule an indefinitely held event.
2. Simulation Clock and a Time Advancing Mechanism
   (a) Unit-time approach
   (b) Event-driven approach




©2006 Raj Jain www.rajjain.com

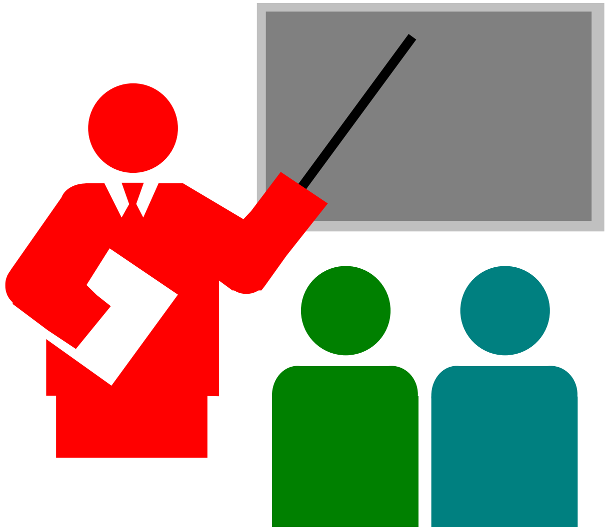
# Components of Discrete Events Sims (Cont)

3. System State Variables
   Global = Number of jobs
   Local = CPU time required for a job
4. Event Routines: One per event.
   E.g., job arrivals, job scheduling, and job departure
5. Input Routines: Get model parameters Very parameters in a range.
6. Report Generator
7. Initialization Routines: Set the initial state. Initialize seeds.
8. Trace Routines: On/off feature
9. Dynamic Memory Management: Garbage collection
10. Main Program




©2006 Raj Jain www.rajjain.com

# Summary



1. Common Mistakes: Detail, Invalid, Short
2. Discrete Event, Continuous time, nonlinear models
3. Monte Carlo Simulation: Static models
4. Trace driven simulation: Credibility, difficult trade-offs

©2006 Raj Jain www.rajjain.com