Network Simulation Basic

CPSC 441

Winter 2020

Sina Keshvadi
University of Calgary

Slides mainly from: CPSC 531: Systems Modeling and Simulation

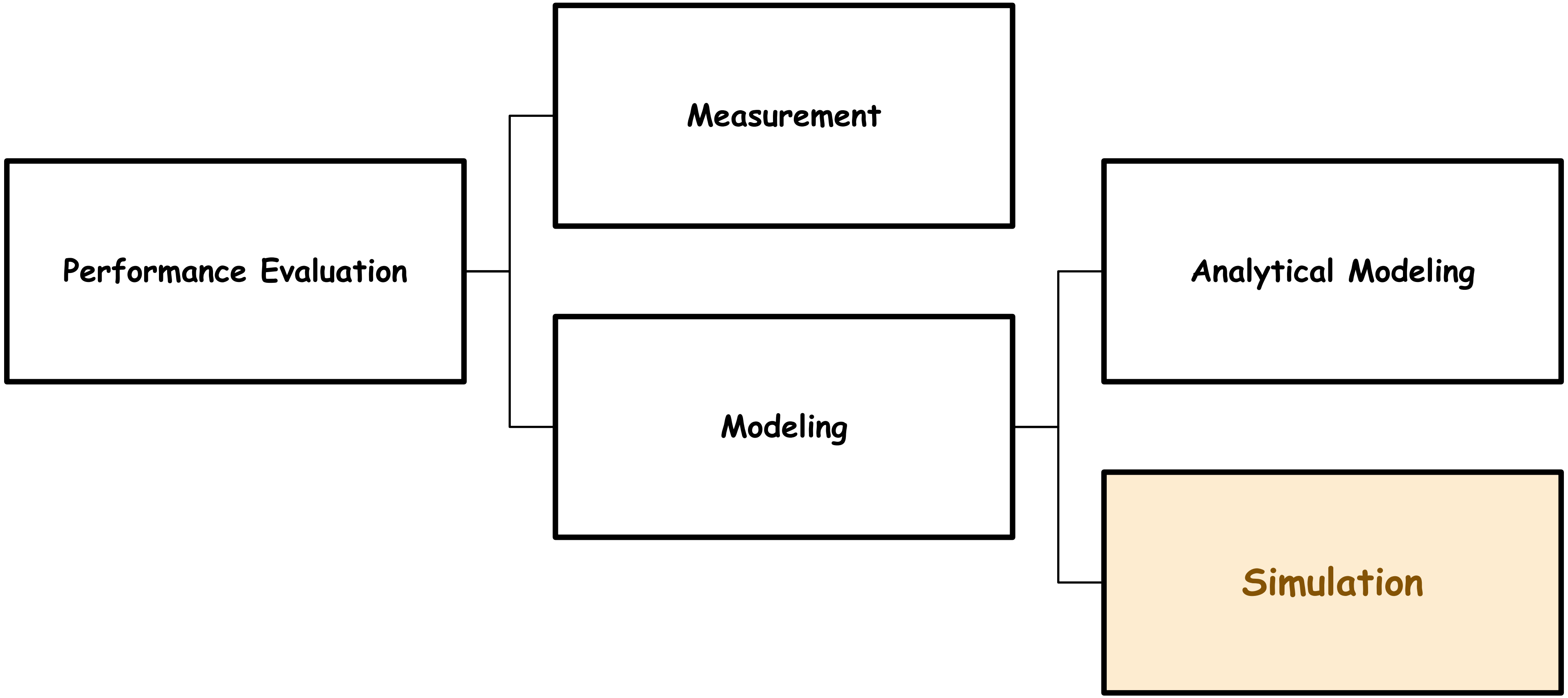
Modeling

- What is a model?
  - An abstract representation of a (real) system that captures the essential characteristics or properties of the system
  - Often requires making simplifying assumptions about how the system actually works
- Examples:
  - Model airplane; molecular model; performance model
- Modeling is an essential tool in computer system performance evaluation
- Note that modeling is both an ‘art’ and a ‘science’

“All models are wrong; some models are useful.”

- George Box, 1976

PERFORMANCE EVALUATION



Simulation Modeling

UNIVERSITY OF CALGARY

- Develop a simulation program that implements the model
- Run the simulation program and use the data collected to estimate the performance measures of interest (typically using randomization)
- A system can be studied at an arbitrary level of detail
- It may be costly to develop and run the simulation program.

Advantages of Simulation

- New policies and procedures can be explored without disrupting the ongoing operation of the real system
- New designs can be tested without committing resources for their acquisition
- Time can be compressed or expanded to allow for a speed-up or slow-down of the phenomenon under study
- Insight can be obtained about the interactions of variables, and which ones have the most impact on system performance
- Can obtain answers to “What if…” questions

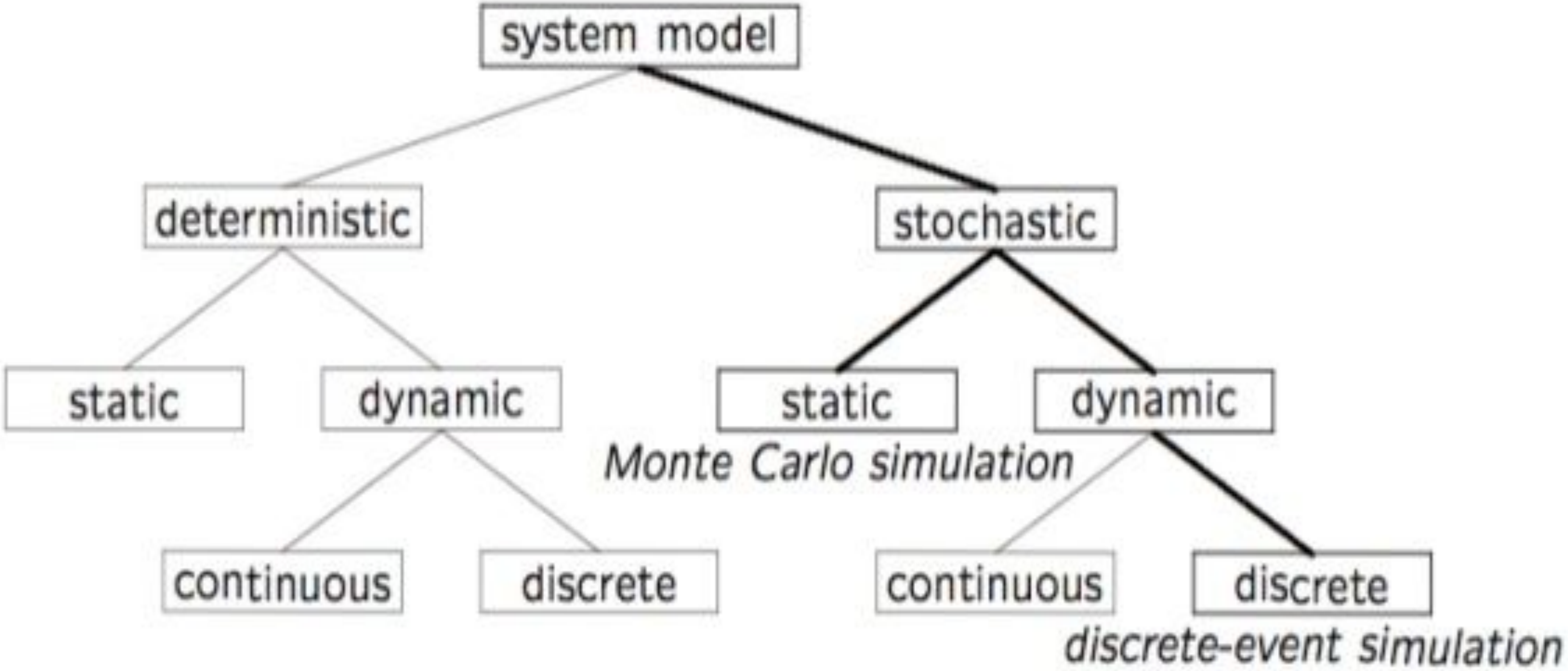
WHEN TO USE SIMULATION

- The problem cannot be solved analytically (or analytical model is too complex)
  - Assignment 3 and 4!
- Direct experiment cannot be done
  - Network protocol scalability test
- Building the real system is expensive
  - Reservoir simulation

“The hardest part about simulation is deciding what NOT to model.”
- Moe Lavigne, Stentor, Summer 1995

Simulation Model Taxonomy

UNIVERSITY OF CALGARY



Monte Carlo Simulation

- Static simulation (no time dependency)
- To model probabilistic phenomenon
- Can be used for evaluating non-probabilistic expressions using probabilistic methods
- Can be used for estimating quantities that are “hard” to determine analytically or experimentally

Named after **Count Montgomery de Carlo**, who was a famous Italian gambler and random number generator (1792-1838).

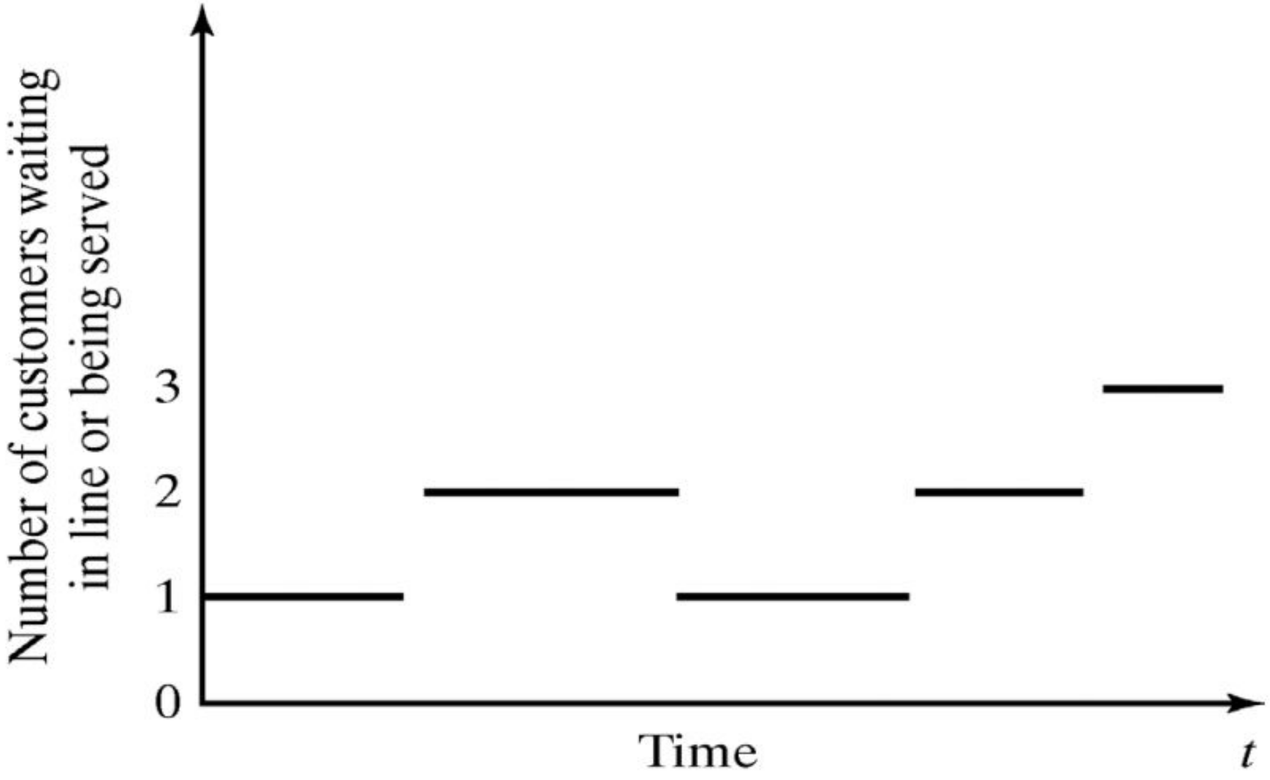
Discrete-Event Simulation

- A simulation model with three features
  - Stochastic: some variables in the simulation model are random
  - Dynamic: system state evolves over time
  - Discrete-Event: changes in system state occur at discrete time instances

Discrete and Continuous Systems

A discrete system is one in which the system state changes only at a discrete set of points in time

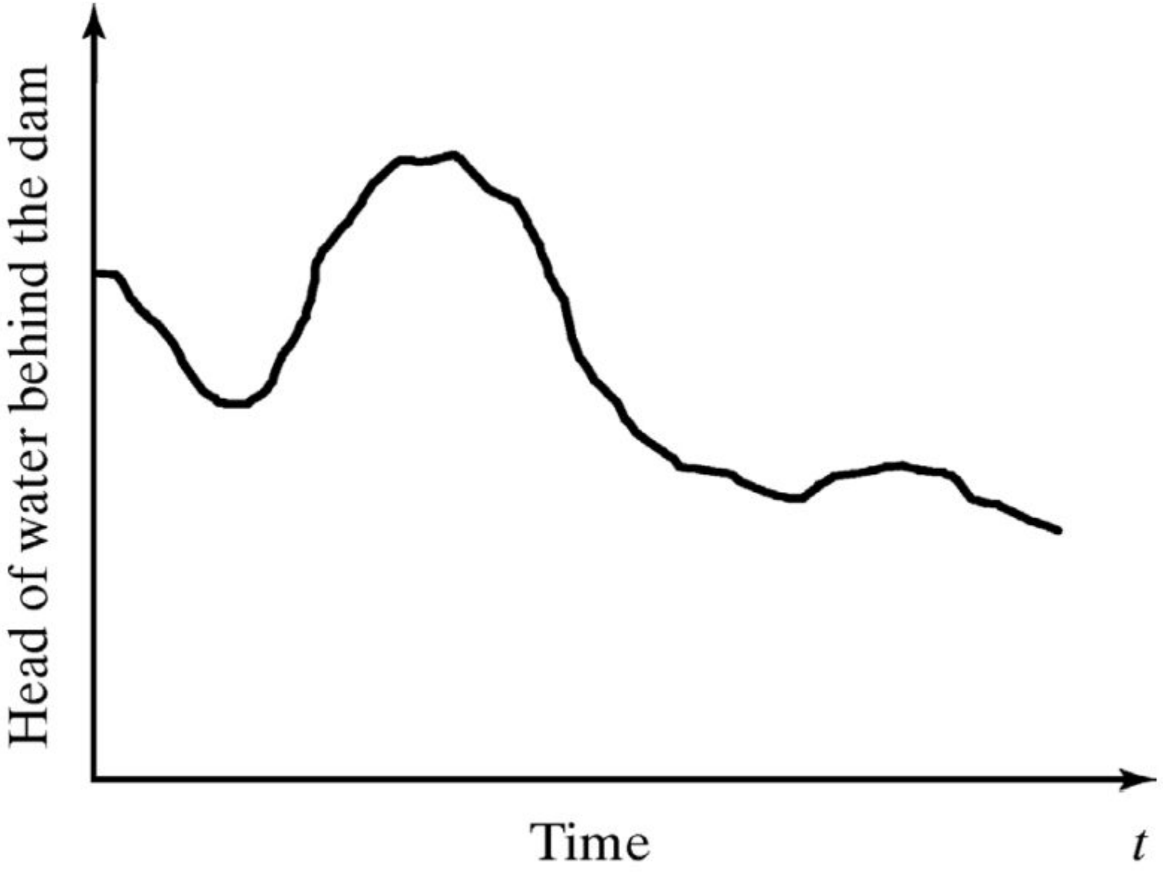
- Example: A restaurant



Discrete and Continuous Systems

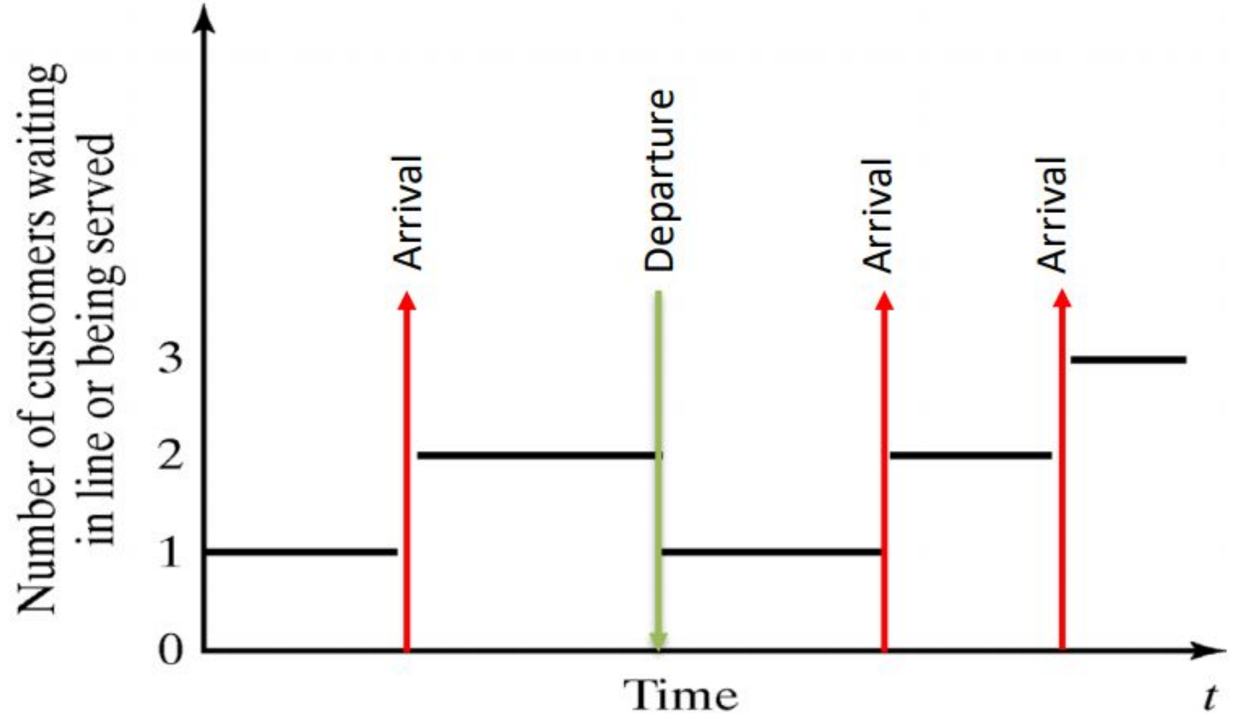
A continuous system is one in which the system state changes continuously over time

- Example: Water level in Bow River (or Bearspaw dam)



Discrete-Event Simulation

- A simulation model in which system state evolves over a discrete sequence of events in time
  - System state changes only when an event occurs
  - System state does not change between the events



Simulation Model Development

- How to develop a simulation model:
  - Determine the goals and objectives
  - Build a conceptual model
  - Convert into a specification model
  - Convert into a computational model
  - Verify the model
  - Validate the model
- Typically an iterative process

Three Model Levels

- Conceptual Model
  - Very high level (perhaps schematic diagram)
  - How comprehensive should the model be?
  - What are the state variables?
  - Which ones are dynamic, and which are most important?
- Specification Model
  - On paper: entities, interactions, requirements, rules, etc.
  - May involve equations, pseudocode, etc.
  - How will the model receive input?
- Computational Model
  - A computer program
  - General-purpose programming language or simulation language?

Simulation Software

- General purpose programming languages
  - Flexible and familiar
  - Well suited for learning DES principles and techniques
  - E.g., C++, Java
- Simulation programming languages
  - Good for building models quickly
  - Provide built-in features (e.g., queue structures)
  - Graphics and animation provided
  - Domain specific
    - Network protocol simulation: ns2, Opnet
    - Electrical power simulation: ETAP
    - Design and engineering: Ansys, Autodesk
    - Process simulation: Simul8

Verification and Validation

- Verification
  - Computational model should be consistent with specification model
  - Did we build the **model right**?
- Validation
  - Computational model should be consistent with the system being analyzed.
  - Did we build the **right model**?
  - Can an expert distinguish simulation output from system output?

Concepts in Discrete-Event Simulation (1 of 2)

- **Model**: an abstract representation of a (real) system
- **System**: a collection of entities that interact together over time (e.g., people, machines, CPU, Web server)
- **System State**: a collection of variables that contain all the information necessary to adequately describe the system at any time (e.g., occupancy)
- **Entity**: any object or component in the system (e.g., a server, a customer, a machine)
- **Attributes**: the properties of a given entity
- **List**: a collection of associated entities, ordered in some logical fashion (e.g., sets, queues)

Concepts in Discrete-Event Simulation (2 of 2)

- **Event**: an instantaneous occurrence that changes the state of a system (e.g., an arrival of a new customer)
- **Event list**: a list of event notices for future events, ordered by time of occurrence, also called the future event list (FEL)
- **Activity** (unconditional wait): a duration of time of specified length that is known when it begins (e.g., a service time)
- **Delay** (conditional wait): a duration of time of unspecified indefinite length, which is not known until it ends (e.g., customer delay while waiting in line)
- **Clock**: a variable representing simulated time, which can be either continuous or discrete

Example 1: ABC Call Center

- A computer technical support center with personnel taking calls and providing service:
    - Three support staff: Alice, Bob, Chris (multiple support channel)
    - A simplifying rule: alphabetical tie-breaker if > 1 staff are idle

- Goal: to find out how well the current arrangement works in terms of the response time of the system

- Random variables:
    - Arrival time between calls
    - Service time (different distributions for Alice, Bob, and Chris)

States

The ABC Call Center System is a discrete-event model with the following components:

- System state:
  - The number of callers waiting to be served at time t
  - Indicator that Alice is idle or busy at time t
  - Indicator that Bob is idle or busy at time t
  - Indicator that Chris is idle or busy at time t
- Entities: neither the caller nor the servers need to be explicitly represented, except in terms of the state variables, unless certain per-caller or per-server statistics are desired

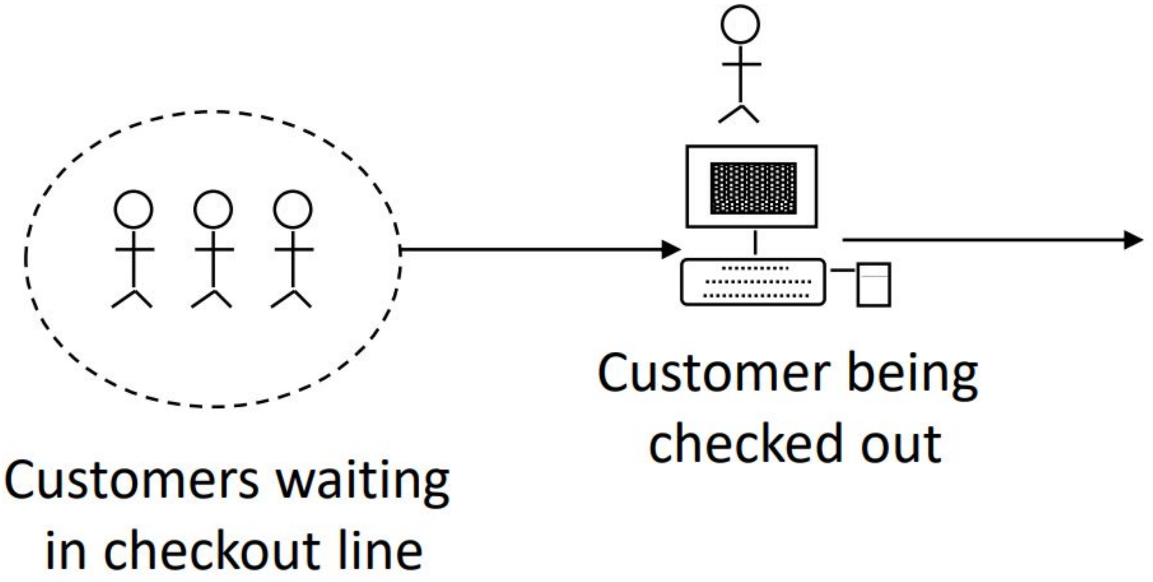
Events

- Events:
  - Arrival of a call
  - Service completion by Alice
  - Service completion by Bob
  - Service completion by Chris
- Activities:
  - Inter-arrival time
  - Service time by Alice
  - Service time by Bob
  - Service time by Chris
- Delay: a caller’s wait in queue until Alice, Bob, or Chris becomes free

Example 3: Grocery Store

Grocery Store with single checkout

- Single-channel queue:
  - The system consists of those customers waiting plus the one (if any) checking out
  - For this example, a stopping time of 60 minutes is set



Components of Grocery Store Example

- **System state**:
  - $LQ\ t$: # of customers waiting in line at time t (excluding the customer being checked out)
  - $LS\ t$: # of customer being checked out (1 or 0) at time t
- **Entities**: the server and customers are not explicitly modeled, except in terms of the state variables
- **Events**: arrival (A), departure (D), stopping event (E)
- **Event notices** (event type, event time):
  - (A, t) representing an arrival event to occur at future time t
  - (D, t) representing a customer departure at future time t
  - (E, 60) representing the simulation stop event at future time 60
- **Activities**: inter-arrival time and service time
- **Delay**: customer time spent in waiting line

Compile and run the source code.

Components of a Simulation

- In DES simulation:
  - The simulation is driven by events
  - The simulation time advances based on sequence of events
  - System state changes with events

- Requirements:
  - Time advance algorithm
  - Event scheduling
  - Event processing

Let’s read Assignment 3

What is the A3’s Goal?

Entities?

System states?

Events?

Activities?

Generic Simulation Program (1 of 4)

- Initialization
  - Initialize clock to zero
  - Initialize state variables and statistical counters
  - Initialize event list (with already known future events)

Generic Simulation Program (2 of 4)

- Main loop (repeat until the condition for terminating the simulation is met)
  - Determine the most imminent event and remove it from the event list (suppose this event is of type i)
  - Advance clock to the time of this event
  - Invoke event routine for type i

Generic Simulation Program (3 of 4)

UNIVERSITY OF CALGARY

- Event routine (a separate routine for each event type)
  - Update state variables
  - Update statistical counters
  - When required, add future events to the event list

Generic Simulation Program (4 of 4)

- Report generator
  - Invoked when simulation has terminated
  - Compute and output performance measures of interest

Let’s discuss about the Assignment 3

Source

CPSC 531: Systems Modeling and Simulation

at http://pages.cpsc.ucalgary.ca/~carey/CPSC531/slides.html