Discrete-Event Simulation

14.11.2001

Contents

- Models and some modelling terminology
- How a discrete-event simulation works
- The classic example - the queue in the bank
- Example for a discrete-event simulation

Simulation

- A definition of simulation:
    1. Imitation of the operation of a real-world system
    2. Generation of an artificial history of a system
    3. Observation of the artificial history
- Simulation is performed using a *model*

Advantages of Simulation

- Simulation has many advantages:
  - study new designs without interrupting real system
  - study new designs without needing extra resources
  - Improve understanding of system
  - Manipulate time
  - Less dangerous / expensive / intrusive

Model

- A model:
  - is a set of assumptions about the operation of the system

- These assumptions can be:
  - algorithmic (sequence of steps)
  - mathematical (equations)
  - logical (conditions)

- This model can be "run" in order to study the real system

Abstraction & Idealisation

There are two main techniques for building models:

- Abstraction
- Idealisation

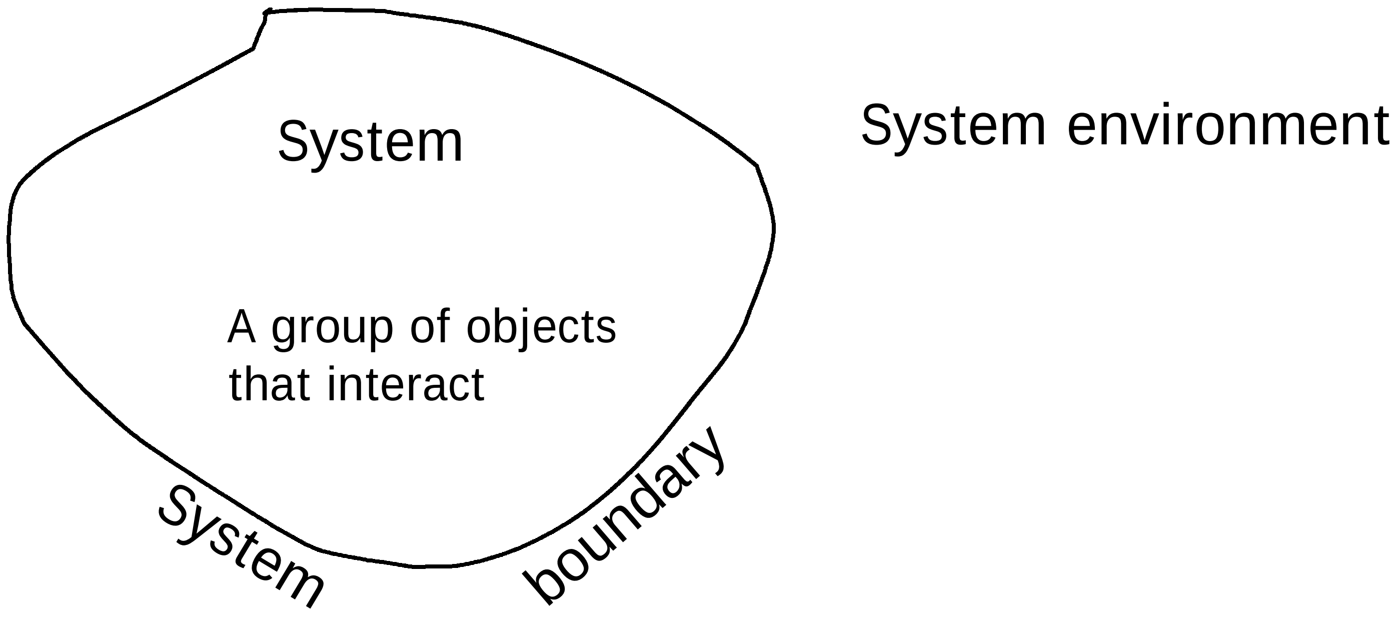
Abstraction means leaving out unnecessary details

- Represent only selected attributes of a customer

Idealisation means replacing real things by concepts

- Replace a set of measurements by a function

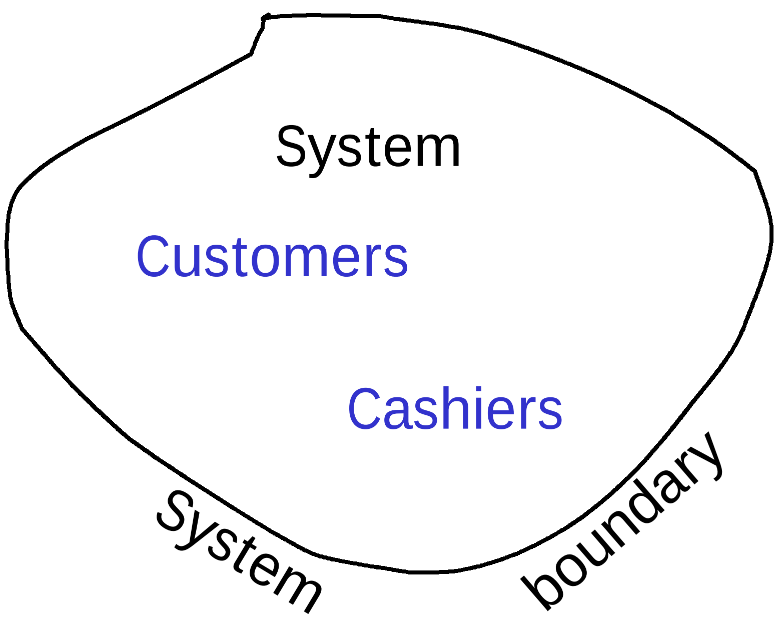
Systems



Placing the system boundary is the first difficult task in modelling

Systems

Example: Modelling a bank:



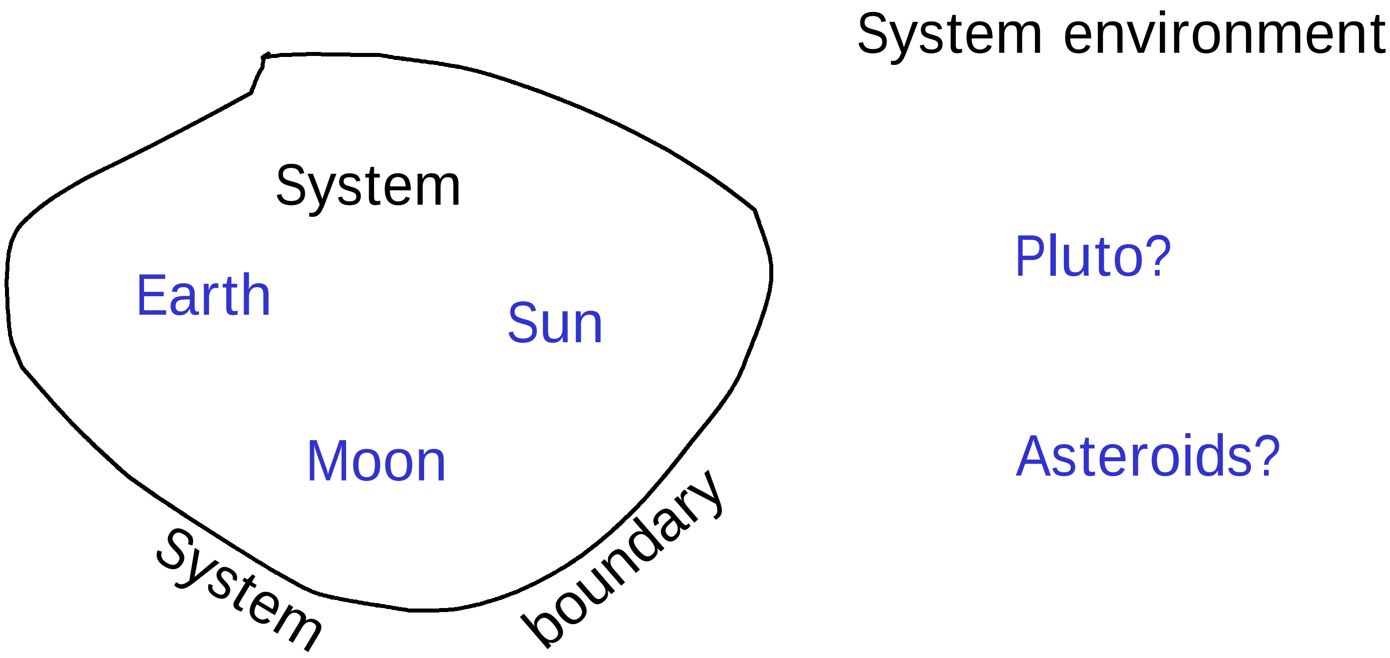
System environment

Time of day

Special events

Systems

Example: Modelling the Earth's orbit:



Entities, Attributes & Activities

An entity is an object of interest in the system

customer manager cashier

An attribute is a (relevant) property of an entity

account balance gender skills

Attributes are state variables

Activities & Delays

An activity… … is a duration of known length

check balance drink coffee serve customer

Activities form part of the model specification

A delay… … is a duration of unknown length

Waiting time in queue

Delays form part of the simulation result

State, State Variables

The (system) state… … is a description which is
- complete and
- minimal

at any point in time

A state variable… … is a variable needed to describe the state

length of queue (0, 1, 2, …)

current activity of manager (sleeping, drinking coffee, …

Events

An event… … is an occurrence which

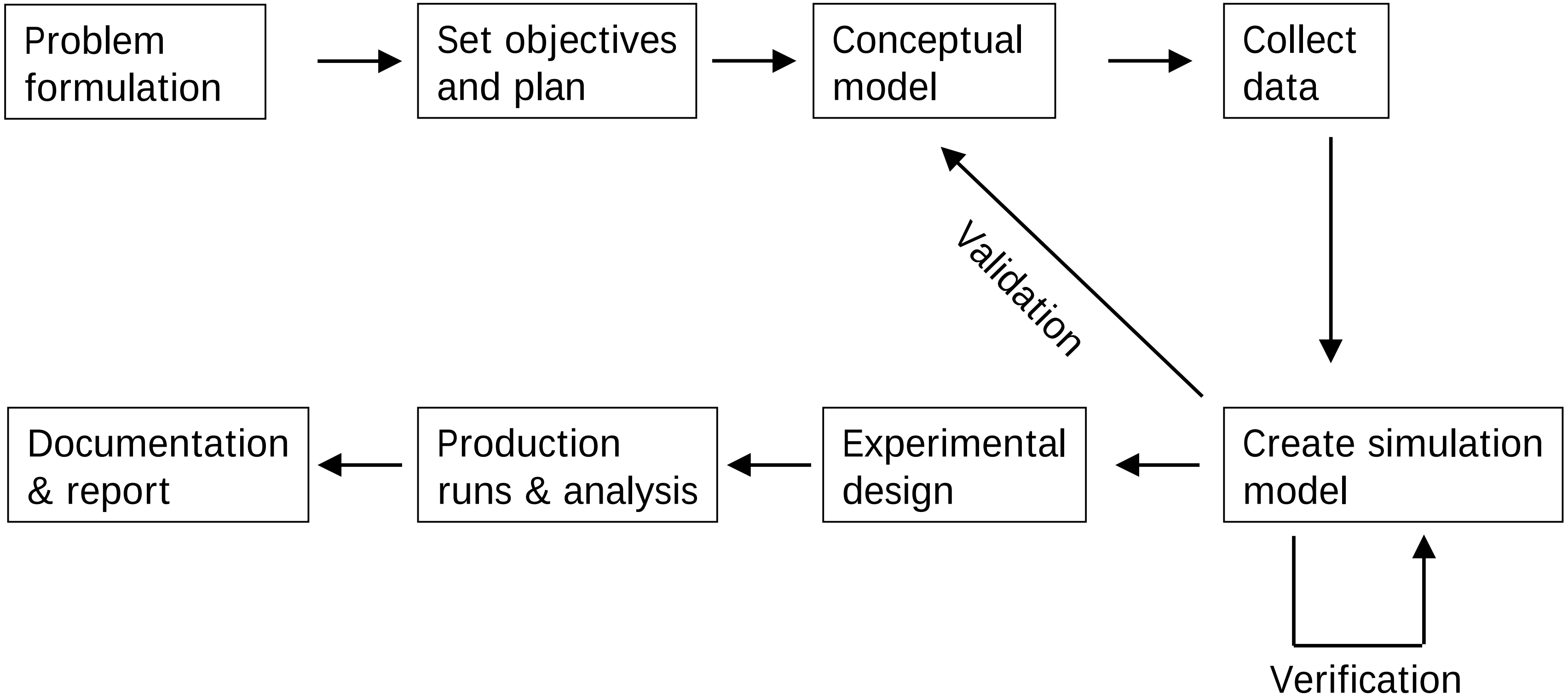
- is instantaneous
- may change the state of the system

Customer arrives from outside

Manager wakes up

Service is completed

Steps in a Simulation Study



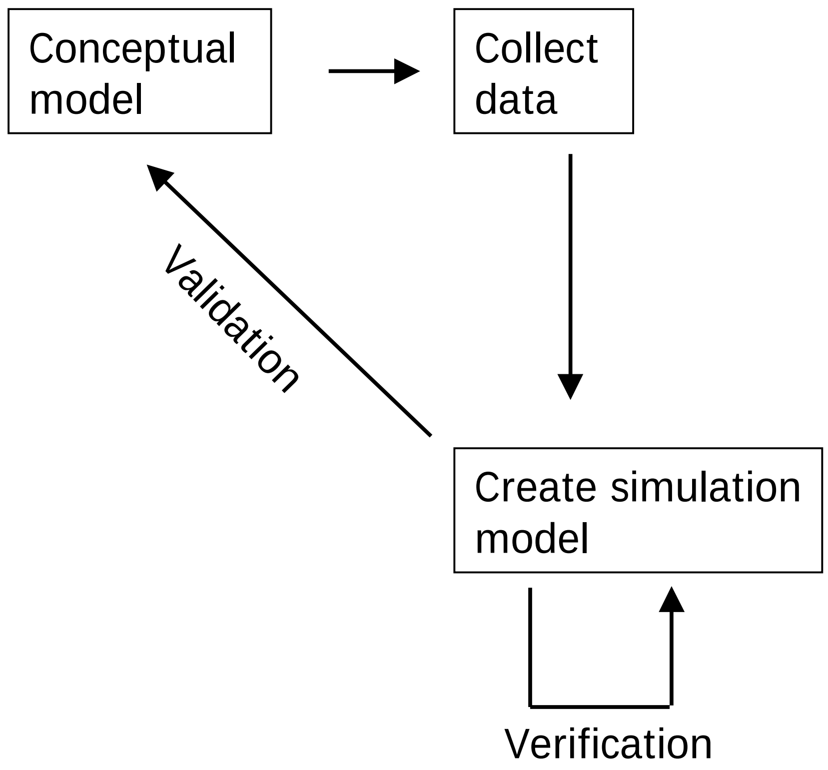
Validation & Verification

Validation...

... means creating a
correct model
("building the right model")

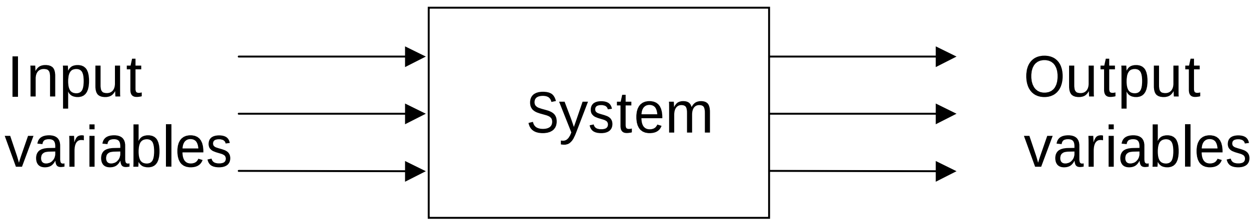
Verification...

... means writing a
correct program
("building the model right")



Input-Output Transformation

- View the (real or simulated) system as a "black box"



- Black box transforms input variables into output variables

- Input variables are obtained by observing the real system

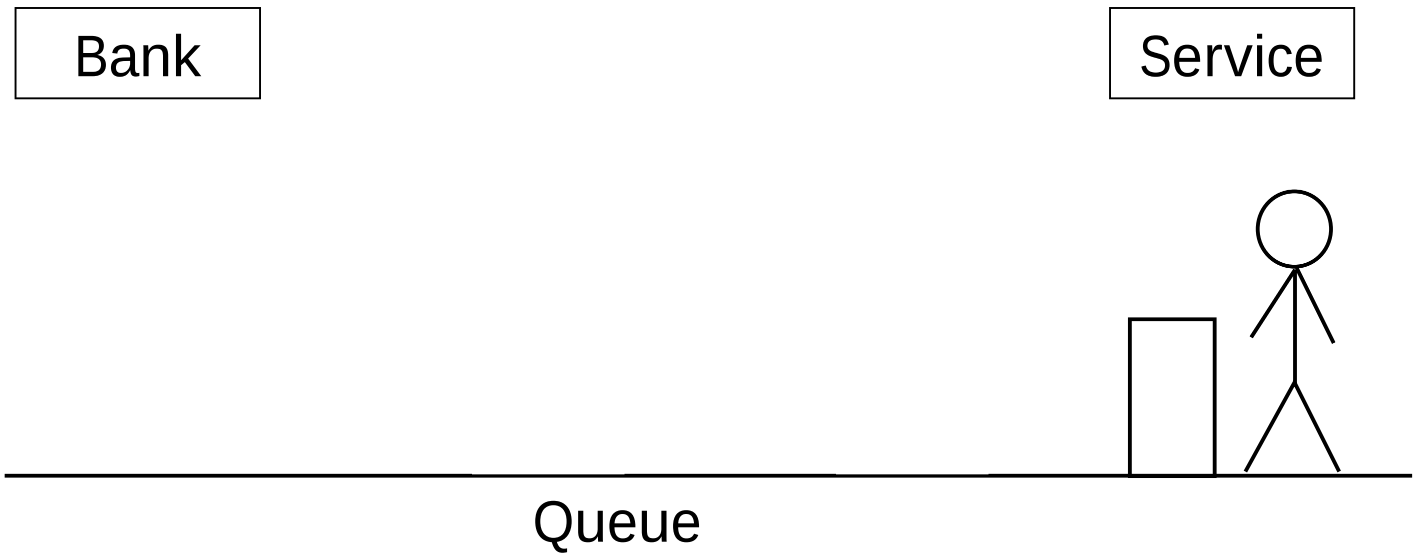
- Output variables are obtained by observing the real system and from the simulation experiment

Model Specification

- Discrete-event modelling raises the following questions:
- How does each event affect the system state and attributes?
- How are activities defined?
  - What events mark the beginning and the end?
  - What conditions (if any) must hold?
- How are delays defined?
- How must the simulation be initialised?

A Simulation Classic

- The single-server queue:

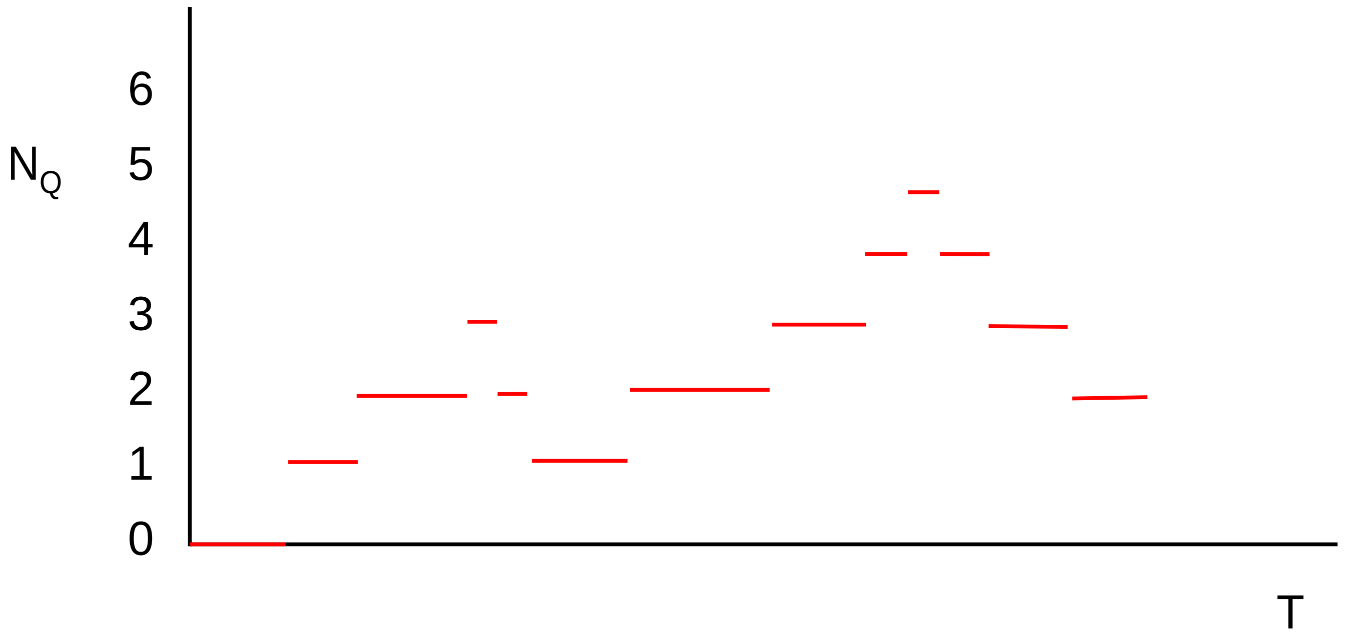


A Simulation Classic

- One possible problem formulation:
  - "Customers have to wait too long in my bank"
- A typical objective:
  - Determine the effect of an additional cashier on the mean queue length
- Data needed:
  - Inter-arrival times of customers
  - Service times

A Simulation Classic

- A typical simulation result:



Event Notice, Event List

- Event notice… … A data record specifying an event

  The event notice must contain all the information necessary to execute the event (in particular the time it is scheduled to occur)

- (Future) event list…… A list of event notices for future events

  The event list is the main data structure in a discrete-event simulator

The Event List

- The (future) event list (FEL) controls the simulation
- The FEL contains all future events that are scheduled
- The FEL is ordered by increasing time of event notice
- Example FEL (at some simulation time $\leq t_1$):

$$\boxed{(t_1, \text{Event}_1)} \rightarrow \boxed{(t_2, \text{Event}_2)} \rightarrow \boxed{(t_3, \text{Event}_3)} \rightarrow \boxed{(t_4, \text{Event}_4)}$$

$$t_1 \leq t_2 \leq t_3 \leq t_4$$

Conditional and Primary Events

- A primary event… … An event whose occurrence is scheduled at a certain time

Arrivals of customers

- A conditional event… … An event which is triggered by a certain condition becoming true

Customer moving from queue to service

Conditional and Primary Events

- Primary event:

```
WHENEVER T >= TArrive
DO
   NQueue^  := NQueue + 1;
   TArrive^ := T + IArrive;
END
```

- Conditional event:

```
WHENEVER (NServer = 0) AND (NQueue > 0)
DO
   NQueue^   := NQueue - 1;
   NServer^  := NServer + 1;
   TService^ := T + IService;
END
```

The Event List

- Example: Simulation of the *Mensa*:
- Some state variables:

  # people in line 1

  # people at meal line 1 & 2

  # people at cashier 1 & 2

  # people eating at tables

(t_1, Arrival) → (t_2, Service completion at cashier 2) → (t_3, Service completion at meal 1) → (t_4, Finish eating) → (t_5, Finish eating)

The Event List

- Operations on the FEL:
  - Insert an event into FEL (at appropriate position!)
  - Remove first event from FEL for processing
  - Delete an event from the FEL

- The FEL is thus usually stored as a linked list

The Event List

- The simulator spends a lot of time processing the FEL
- In some cases, up to 40% of overall simulation time!
- Efficiency is thus very important
- Sequential search when inserting must be avoided: $O(n)$
- Solution: more advanced data structures (trees): $O(\log n)$

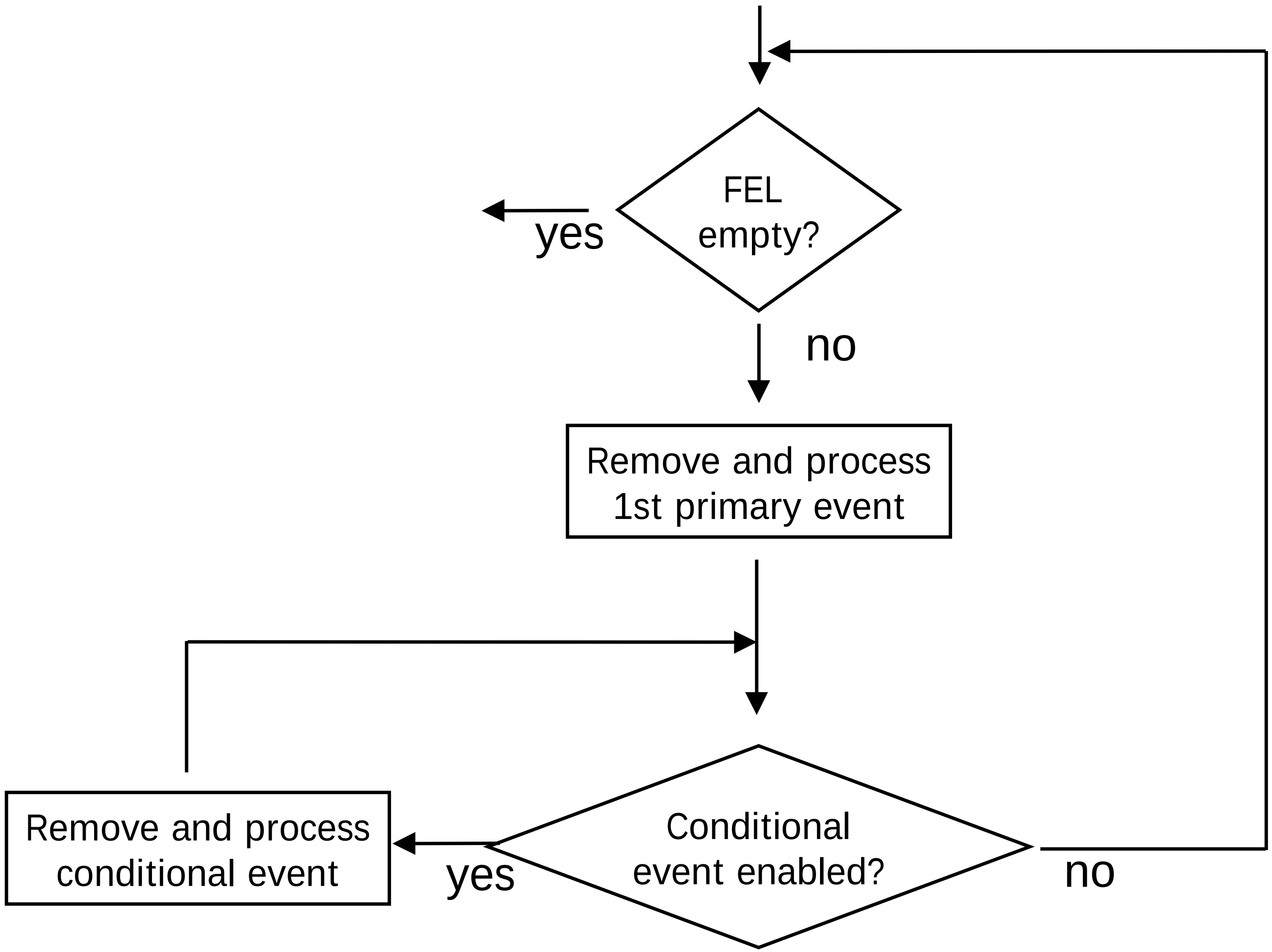
The Event List

- Example: Event "Customer arrives" in the bank model:

```
# Customer arrives
WHENEVER T >= TArrive
DO
   NQueue^  := NQueue + 1;
   TArrive^ := T + IArrive;
   DISPLAY ("T= %f Customer arrives\n",T);
END
```

- Executing this event at time **`T`** causes the event
  `(T + IArrive, Customer arrives)`
  to be inserted into the FEL

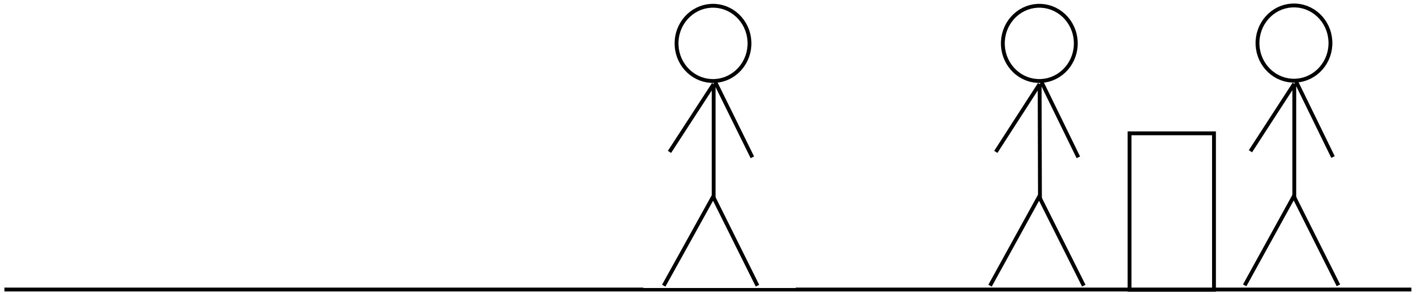
The Simulation Algorithm



The Simulation Algorithm

- Remove and process 1st primary event:
  - Remove 1st primary event from FEL
  - Advance simulation time
  - Update state variables
  - Enter new future events into FEL
  - Compute statistics
- Every discrete-event simulator works like this (even if the programming model looks different!)

The Simulation Algorithm



(t_1, Arrival)

(t_1, Move up) → (t_2, Arrival)

(t_2, Arrival) → (t_3, Service complete)

(t_4, Arrival) → (t_3, Service complete)

Timing

- Sometimes, activities last for an exact amount of time:
  - The clock cycle in a computer
  - Each phase of a traffic light
  - A certain operation in a manufacturing line
- Such exact times are *deterministic*

Timing

- Usually, activities last for varying amounts of time:
  - Inter-arrival times at bank
  - Service times at bank
  - Time to failure for a machine
  - Time that a user program runs
- Such times are *random* or *stochastic*

Random Variables

- If any quantity in a simulation is random, then …

  … the simulation result must also be random

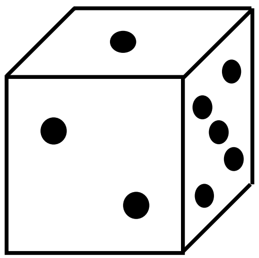
- This can make things complicated:

  - The simulator will need to use random variables

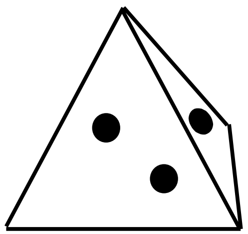
  - We will need to do some statistics

Random Variables

- The best-known random variable:



Discrete and uniformly distributed on [1,6]



Discrete and uniformly distributed on [1,4]

Simulating the Bank by Hand

Simulation clock: 15

Arrival interval	Customer arrives	Begin service	Service duration	Service complete
5	5	5	2	7
1	6	7	4	11
3	9	11	3	14
3	12	14	1	15

Computing Statistics

Average waiting time for a customer: 1.25

Arrival interval	Customer arrives	Begin service	Service duration	Service complete
5	5	5	2	7
1	6	7	4	11
3	9	11	3	14
3	12	14	1	15

Computing Statistics

P(customer has to wait): 0.75

Arrival interval	Customer arrives	Begin service	Service duration	Service complete
5	5	5	2	7
1	6	7	4	11
3	9	11	3	14
3	12	14	1	15

Computing Statistics

P(Server busy): 0.66

Arrival interval	Customer arrives	Begin service	Service duration	Service complete
5	5	5	2	7
1	6	7	4	11
3	9	11	3	14
3	12	14	1	15

Computing Statistics

Average queue length: 0.33

Arrival interval	Customer arrives	Begin service	Service duration	Service complete
5	5	5	2	7
1	6	7	4	11
3	9	11	3	14
3	12	14	1	15