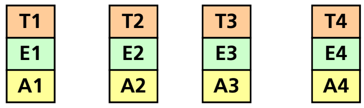
Simulation

The following is needed:

- A description of the state of the system
- The events that can occur
- Rules describing what will happen if an event occurs

The event list

Keeps track of when events shall happen

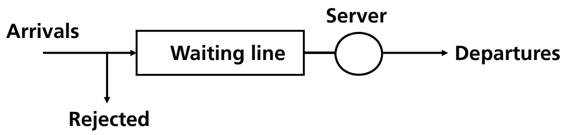


Ti = time when event Ei will take place
Ai = attributes to event I
The list is sorted: T1 < T2 < T3 < T4 etc.

How a simulation run is done

1. Extract the first element in the event list
2. Set Time = the time of the extracted event
3. Update the state of the system and insert new events if needed
4. If not finished, Go to 1

An example: a queuing system



It might be of interest to find
•Probability of rejection
•Mean (or variance) of time spent in system
•The mean number of customers in the system

The state description

Assume that we want to find the mean number of customers in the queue.

N = number of customers in the system

The appropriate state description depends on the results we desire

Events that may take place

- Arrival
- Departure (when service is ready)
- Measurement (does not change the state)

What we also need to know

Assume the following:

- The service time distribution is exponential with mean 2
- The mean time between arrivals is exponential with mean 3
- The number of places in the waiting line is infinite

Rule at arrival

```
N := N + 1;
If N=1 then
   add departure to event list ;
Add a new arrival to event list;
```

When we add the departure and arrival we have to draw a random number (exponentially distributed)

Rule at departure

```
N := N - 1;
If N>0 then
  add departure to event list ;
```

Rule at measurement

```
Write(N)
Add a new measurement to event
  list;
```

When the simulation begins

Time and state:	Event list:
Time = 0	3 Arrival
N = 0	5 measurement

Step 1

Time and state:	Event list:
Time = 3	4 Arrival
N = 1	5 measurement
	9 Departure

Step 2

Time and state:
Time = 4
N = 2

Event list:
5 measurement
9 Departure
10 Arrival

Step 3

Time and state:
Time = 5
N = 2

Event list:
9 Departure
10 Arrival
14 Measurement

Step 4

Time and state:
Time = 9
N = 1

Event list:
10 Arrival
12 Departure
14 Measurement

```
begin
   a := 3; (* mean time between arrivals = 3 *)
   s := 2; (* mean service time = 3 *)
   m := 10; (* mean time between measurements = 10 *)
   simulationlength := 1000;
   No_in_queue := 0;
   time := 0;
   insert_event(measurement,Exp(m));
   insert_event(arrival, Exp(a));
   while time < simulationlength do
   begin
      dummy := FirstInQueue(eventlist);
      time := dummy.eventtime;
      case dummy.eventkind of
         arrival: arrive;
         departure: depart;
         measurement: measure;
      end;
   end;
end.
```

```
procedure arrive;
begin
   if No_in_queue = 0 then
      insert_event(departure,Exp(s));
   No_in_queue := No_in_queue + 1;
   insert_event(arrival, Exp(a));
end;

procedure depart;
begin
   No_in_queue := No_in_queue - 1;
   if No_in_queue > 0 then
      insert_event(departure, Exp(s));
end;

procedure measure;
begin
   write(utfil, No_in_queue);
   insert_event(measurement, Exp(m));
end;
```