

CPE 345

Modeling and Simulation

Lecture 2

January 31, 2005

Outline for today's topics

- How to run a simulation
 - Manually – using a simulation table (Chapter 2 in your textbook)
 - Using OMNET++ : An introduction
- Examples from queueing systems
 - Single server queues
 - Multi-server queues

System simulation using a simulation table

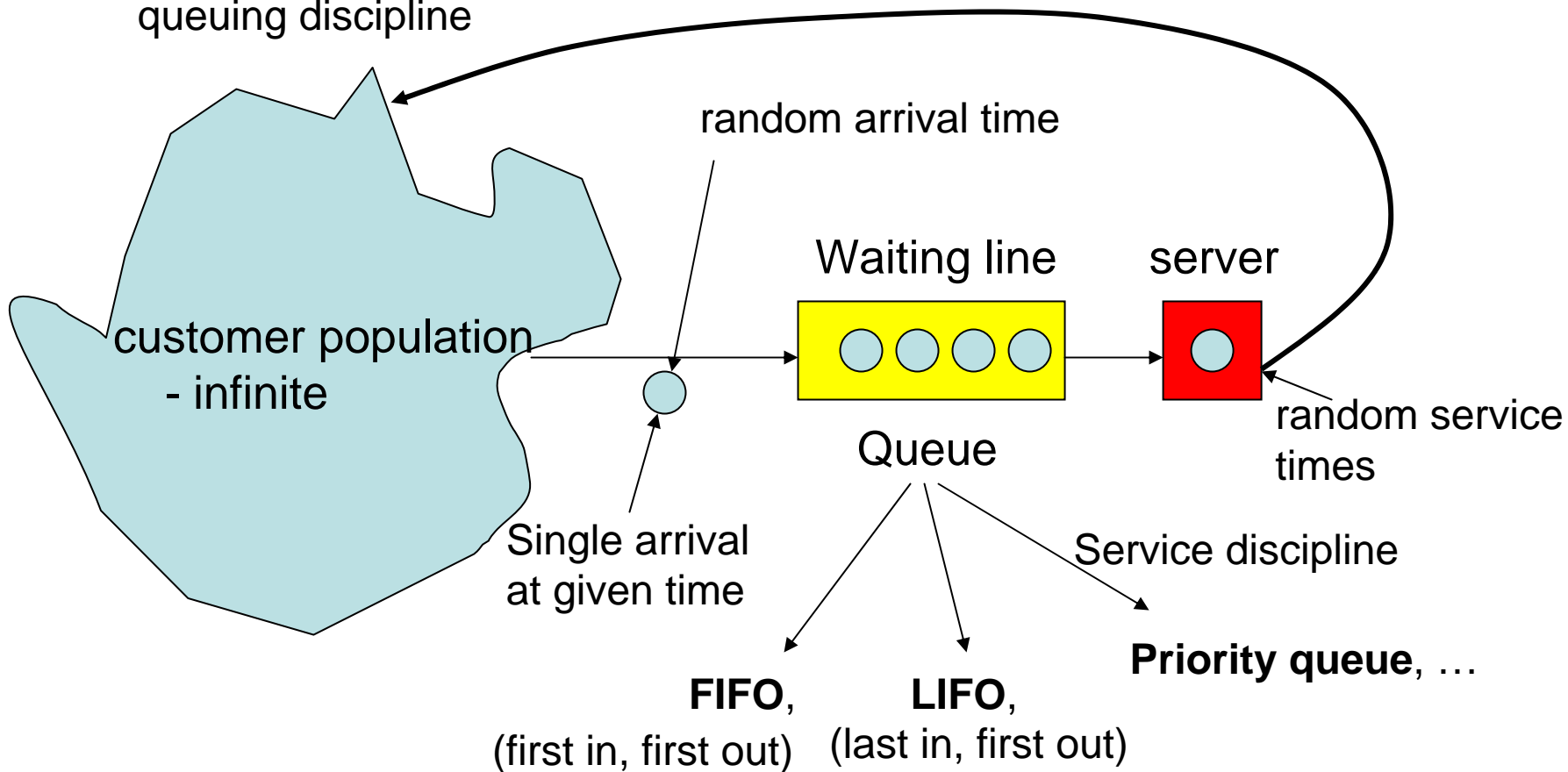
- Is it a useful approach ?
 - You will not really run the simulations manually in practice
 - Gives you an insight on how the system evolves in time
 - Didactic value
 - Helps you implement or verify a simulator (computer program)
- How to construct a simulation table ?
 - Columns in the table: inputs and reponse(s)
 - Rows: repetitions of system operation (i.e., the inputs are changing)
 - Often, inputs are random variables, and are characterized by probability distributions (probability density functions – *pdf*)

Example of a generic simulation table

Repetition	Inputs							Response
	x_{i1}	x_{i2}	x_{i3}	x_{ij}	x_{im}	
1								
2								
3								
⋮								
n								

Example: simple queueing system

- Queueing system:
 - unit population, nature of arrivals, service mechanism, system capacity, queuing discipline

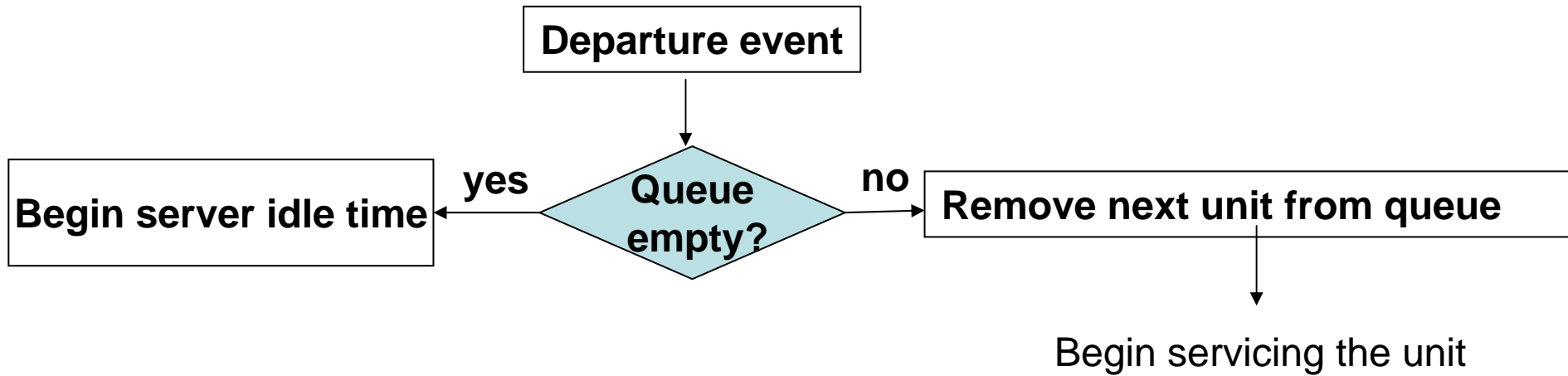


Note: Average arrival rate < average service rate - Why?

Queueing system components

- Entities: server, queue
- State:
 - number of units (customers for the bank example) in the system, Q
 - Server status: busy/idle, $S = \{B, I\}$.
- Events:
 - Arrivals
 - Departures
- Simulation clock: tracks simulated time
- Actions
 - Different actions, depending on the type of the event (arrival or departure) and the current system state
 - Flowcharts for actions following both arrival and departure events

Flowchart for departure



How does system state change?

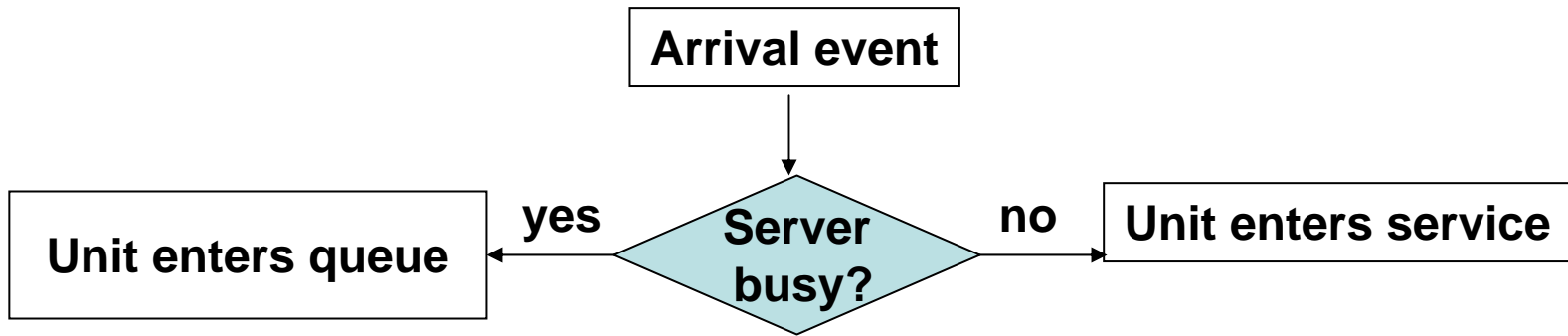
If $Q \neq 0$, $Q = Q - 1$; $S = B$;

If $Q = 0$, $Q = 0$; $S = I$.

Are these valid: If $Q=0$, $Q=0$, $S=B$?

If $Q \neq 0$, $Q = Q$; $S = I$?

Flowchart for arrival



How does system state change?

If $S \neq B$, $Q = Q$; $S = B$;

If $S = B$, $Q = Q+1$; $S = B$.

Is this valid: If $S = B$, $Q = Q+1$, $S = I$?

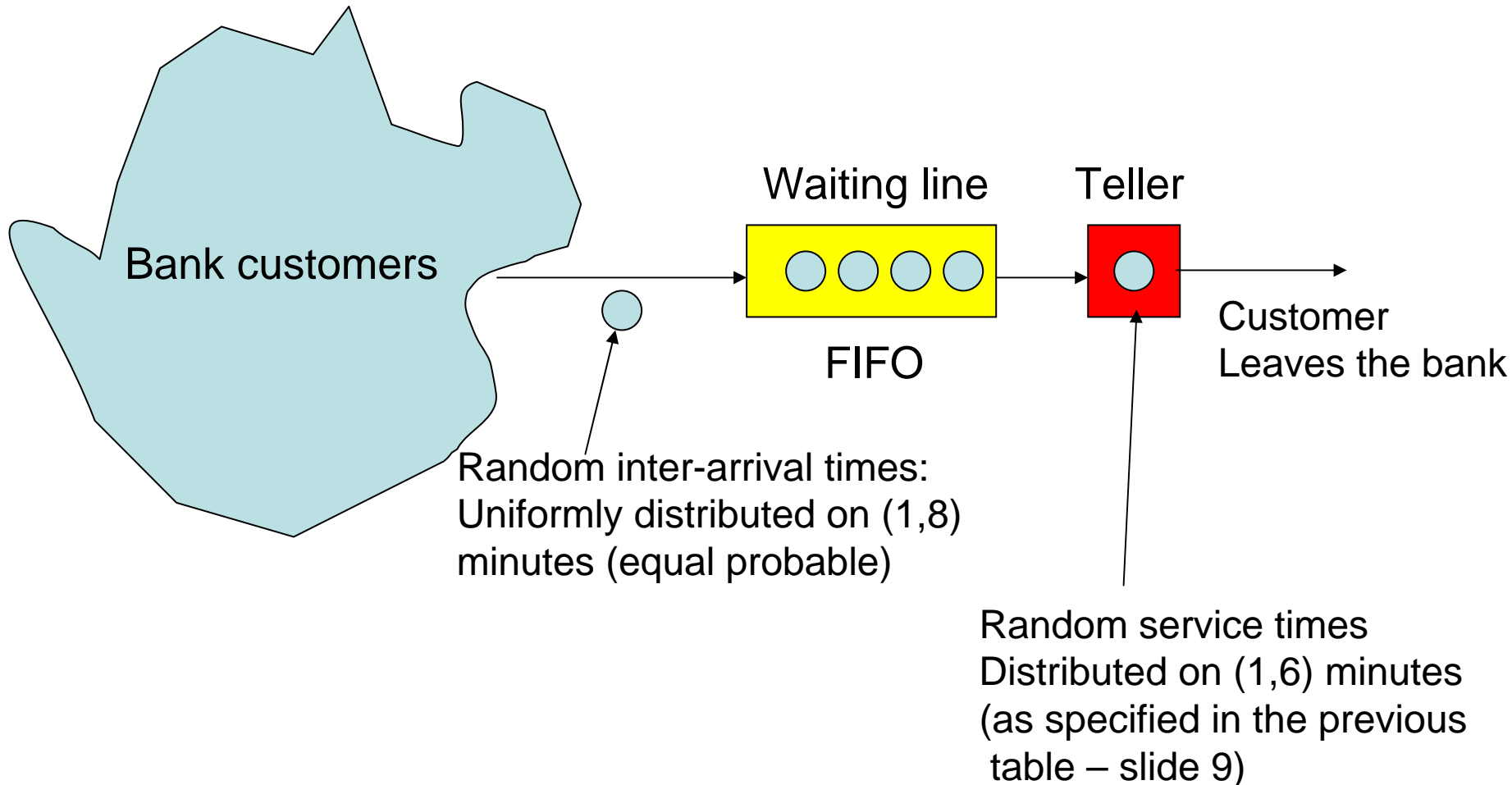
Random numbers

- Inter-arrival times and service times – random numbers
- Pure random numbers – difficult to generate
 - Generation of random numbers discussed later on in the course
- A *rand()* function (Excel, C) can be used to generate uniform random numbers in (0,1)
- Random digits can then be generated using *rand()* by using a simple algorithm
 - For example, if the service time distribution is specified as in the Table:

service time (minutes)	Probability	Cumulative probability
1	0.10	0.10
2	0.20	0.30
3	0.30	0.60
4	0.25	0.85
5	0.10	0.95
6	0.05	1.00

```
int service_time(void) {  
    r = rand()/RAND_MAX;  
    if (r < 0.1) return(1);  
    else if (r < 0.3) return(2);  
        else if (r < 0.6) return(3);  
            else if (r < 0.85) return(4);  
                else if (r < 0.95) return(5);  
                    else return(6);  
}
```

Bank teller example



Objective: simulate arrivals and service for 20 customers

Time between arrivals

Customer	Time between arrivals (min)		Customer	Time between arrivals (min)
1	-		11	1
2	8		12	1
3	6		13	5
4	1		14	6
5	8		15	3
6	3		16	8
7	8		17	1
8	7		18	2
9	2		19	4
10	3		20	5

Note: arrival times have been generated randomly as discussed before

Service Times

Customer	Service time (min)		Customer	Service time (min)
1	4		11	3
2	1		12	5
3	4		13	4
4	3		14	1
5	2		15	5
6	4		16	4
7	5		17	3
8	4		18	3
9	5		19	2
10	3		20	3

Note: service times have been generated randomly as described in textbook 12

Simulation table for the bank teller example

Customer	Time since last arrival	Arrival time	Service time	Time service begins	Time in queue	Time service ends	Customer time in system	Idle time for server
1	-	0	4	0	0	4	4	0
2	8	8	1	8	0	9	1	4
3	6	14	4	14	0	18	4	5
4	1	15	3	18	3	21	6	0
5	8	23	2	23	0	25	2	2
6	3	26	4	26	0	30	4	1
7	8	34	5	34	0	39	5	4
8	7	41	4	41	0	45	4	2
9	2	43	5	45	2	50	7	0
10	3	46	3	50	4	53	7	0

Simulation table for the bank teller example -cont

Customer	Time since last arrival	Arrival time	Service time	Time service begins	Time in queue	Time service ends	Customer time in system	Idle time for server
11	1	47	3	53	6	56	9	0
12	1	48	5	56	8	61	13	0
13	5	53	4	61	8	65	12	0
14	6	59	1	65	6	66	7	0
15	3	62	5	66	4	71	9	0
16	8	70	4	71	1	75	5	0
17	1	71	3	75	4	78	7	0
18	2	73	3	78	5	81	8	0
19	4	77	2	81	4	83	6	0
20	5	82	3	83	1	86	4	0
Total	82		68		56		124	18

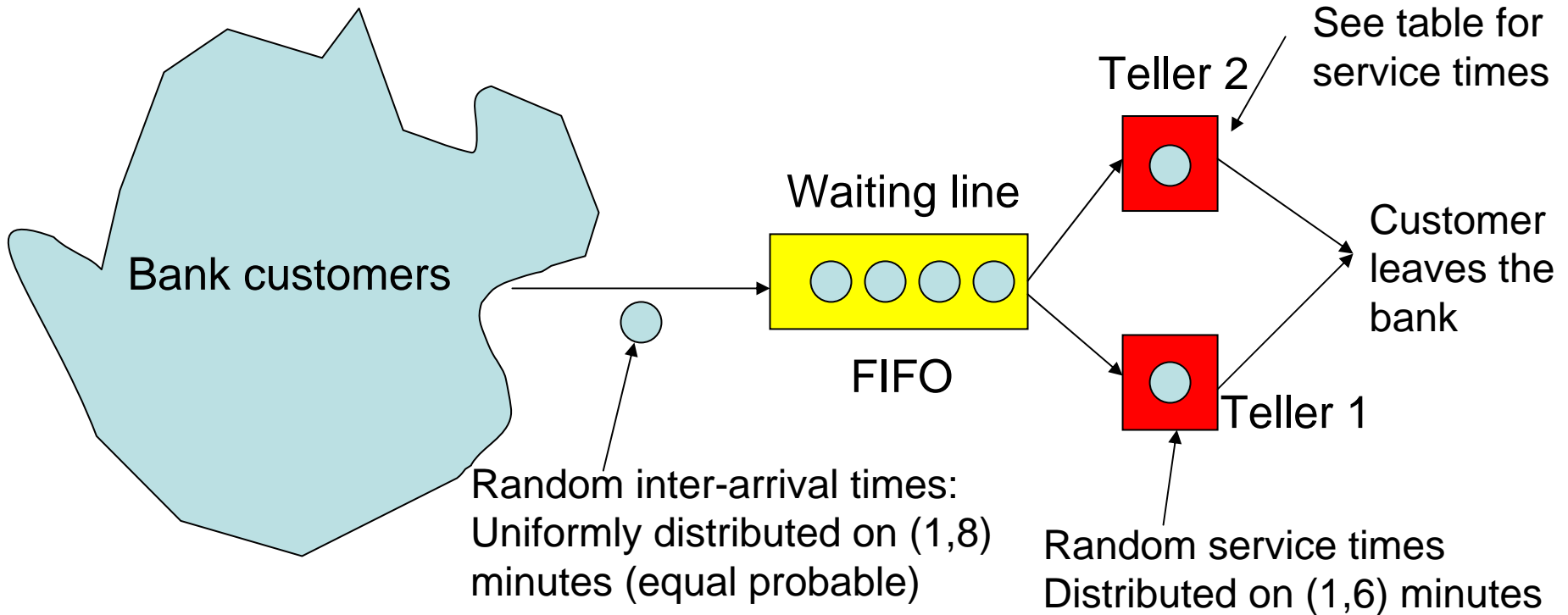
System Statistics

- Average time between arrivals:
 - (sum of all inter-arrival times)/(number arrivals -1) = $82/19 = 4.3$ min.
- Expected time between arrivals
 - $E(T) = t \cdot p(t) = 1/8(1+2+3+4+5+6+7+8) = 4.5$ min.
 - Why is this different from the simulated inter-arrival time?
- Average service time
 - (total service time)/(total number of customers) = $68/20 = 3.4$ min
- Expected service time = $1 \cdot 0.1 + 2 \cdot 0.2 + 3 \cdot 0.3 + 4 \cdot 0.25 + 5 \cdot 0.1 + 6 \cdot 0.05 = 3.2$ min
 - Why is this different from the simulated average service time?
- Average waiting time
 - (total waiting time in queue)/(number of customers who wait) = $56/13 = 4.3$ min.

More statistics

- Average time spent in the system
 - (total time that customers spend in the system)/(total number of customers) = $124/20 = 6.2$ min.
- Average time in queue+ average time in service = average time spent in the system => $2.8+3.4 = 6.2$ min.
- Probability that a customer has to wait in a queue
 - $P(\text{wait}) = (\text{number of customers that wait})/(\text{total number of customers}) = 13/20 = 0.65$
- Fraction of idle time for server
 - $P(\text{idle}) = (\text{total idle time})/(\text{total simulation time}) = 18/86 = 0.21$

What if we add another teller?



Teller 2: service time distribution

Service time (min)	Probability	Cumulative probability
3	0.35	0.35
4	0.25	0.60
5	0.20	0.80
6	0.20	1.00

Service policy:

If both tellers are idle, Teller 1 serves the next customer
Otherwise, the customer is served by the next available teller¹⁷

Simulation table for the bank teller example – two tellers

Teller 1

Teller 2

Customer	Arrival time	Service time	Time service begins	Time service ends	Time service begins	Time service ends	Time in queue	Idle time T1	Active time T2
1	0	4	0	4			0	0	0
2	8	1	8	9			0	4	0
3	14	4	14	18			0	5	0
4	15	3			15	18	0	0	3
5	23	2	23	25			0	5	0
6	26	4	26	30			0	1	0
7	34	5	34	39			0	4	0
8	41	4	41	45			0	2	0
9	43	6			43	49	0	0	6
10	46	5	46	51			0	1	0

Simulation table for the bank teller example – two tellers

Teller 1

Teller 2

Customer	Arrival time	Service time	Time service begins	Time service ends	Time service begins	Time service ends	Time in queue	Idle time T1	Active time T2
11	47	4			49	53	2	0	4
12	48	3	51	54			3	0	0
13	53	4			53	57	0	0	4
14	59	3	59	62			0	5	0
15	62	5	62	67			0	0	0
16	70	4	70	74			0	3	0
17	71	4			71	75	0	0	4
18	73	1	74	75			1	0	0
19	77	5	77	82			0	2	0
20	82	4	82	86			0	0	0
Total							6	32	21

Queueing statistics comparisons

Scenario	Queueing time (min)	Fraction of idle time teller 1 (min)	Fraction of active time teller 2 (min)	Fraction of idle time teller 2 (min)
1 Teller	4.3	0.21		
2 Tellers	2	0.372	0.244	0.756

What if questions - examples

- What if you change the service policy ?
 - If both tellers free, flip a coin and randomly choose one to service the next customer
 - Add second teller to serve only preferred customers (gold membership)
 - Two separate lines (queues)
 - Two queueing times, for regular customers and for gold customers
- What if you add a more rapid teller (with a higher salary) instead of the two slower tellers?

Which solution is the most economical appealing while guaranteeing good service for customers?

Note: a similar example for a drive-in restaurant is discussed in chapter 2 of your textbook: The Able Baker Carhop problem

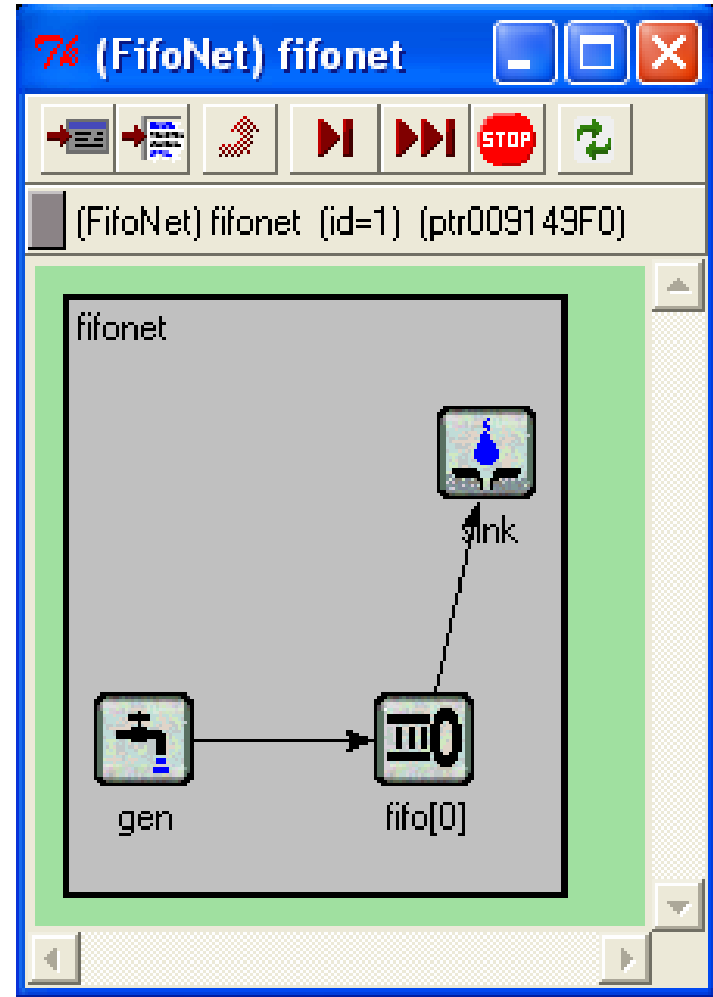
Implement the simulation using OMNET++

1. Identify the system entities

- Arrivals source – generator
- Queue
- Sink – customers (messages) that exit the system are deleted

2. Define the network to be simulated

- Create a **.ned** file that contains the network description
- The .ned file can be also generated using the graphical editor
 - Not recommended – graphical editor has bugs
 - edit the file using a simple text editor



Simulation in OMNET++

3. Define events pattern

- in OMNET++, events are represented using messages. Messages are sent from one module to another. The place where the event will occur is the destination module, and the time when the event will occur is the arrival time of the message.
- Events like “timeout” are implemented with the module sending a message to itself at a scheduled time.
- The message with earlier arrival time is executed first. If arrival times are equal, the one with smaller priority value is executed first. If priorities are also equal, the one scheduled/sent earlier is executed first.

4. Define actions for modules

5. Collect statistics

The network description

- The network topology is specified using the NED language
 - Modular description of a system
 - Simple/compound modules, channels, gates, etc
 - The .ned files are translated into C++ code by the NEDC compiler, then compiled by the C++ compiler and linked into the simulation executable
 - This is transparent to the user – just use `nmake -f Makefile.vc`
- Reserved words:
 - Import, include, channel, endchannel, simple, endsimple, module, endmodule, error, delay, datarate, const, parameters, gates, submodules, connections, gatesizes, on, if, machines, for, do, endfor, network, endnetwork, nocheck, ref, ancestor, true, false, like, input, numeric, string, bool, char

Example: fifonet1.ned

```
import  
  "gen1",  
  "fifo1",  
  "sink1";
```

*Imports gen1.ned; fifo1.ned; sink1.ned
- imports declarations from these files*

```
module FifoNet1  
  submodules:
```

Compound module

```
    gen: FF1Generator;
```

Declaration in gen1.ned

```
      parameters:
```

```
        num_messages = input,
```

*Can take string, numeric or
pointer values*

```
        ia_time = input,
```

```
        msg_length = input;
```

```
        display: "b=32,30;p=76,78;i=gen";
```

```
    fifo: FF1BitFifo;
```

Declaration in fifo1.ned

```
      parameters:
```

```
        bits_per_sec = input;
```

```
        display: "p=164,78;i=queue";
```

```
    sink: FF1Sink;
```

Declaration in fifo1.ned

```
      display: "p=244,78;i=sink";
```

```
  connections:
```

```
    gen.out --> fifo.in;
```

```
    fifo.out --> sink.in;
```

```
endmodule
```

```
network fifonet1 : FifoNet1  
endnetwork
```

Network declaration

Example: fifonet1.ned - cont

gen1.ned

```
simple FF1Generator
  parameters:
    num_messages,
    ia_time,
    msg_length;
  gates:
    out: out;
endsimple
```

fifo1.ned

```
simple FF1PacketFifo
  parameters:
    service_time : numeric;
  gates:
    in: in;
    out: out;
endsimple
```

```
simple FF1BitFifo
  parameters:
    bits_per_sec : numeric;
  gates:
    in: in;
    out: out;
endsimple
```

sink1.ned

```
// FF1Sink -- //
// destroys the
// packets and creates
// statistics
//
```

```
simple FF1Sink
  gates:
    in: in;
endsimple
```

Network description- cont.

- The simple module definitions (e.g. FF1Generator, FF1PacketFifo, FF1BitFifo, FF1Sink), as well as the activity of the modules are implemented in C++ files
 - e.g. gen1.cpp, fifo1.cpp, sink1.cpp
 - More details on this later on in the course
- All the parameters can be initialized using the omnetpp configuration settings file. If a required parameter value is missing, at run-time, the user will be prompted to enter a value for the missing parameter.
 - The *input* keyword specifies the fact that the parameter value is going to be initialized via opnetpp config. file, or at run-time
 - You can also specify default values for parameters

<param_name> = input(<default-value>, <prompt>)
<param_name> = input(<default-value>)

e.g. *num_messages = input(500, "Number of messages?");*

Homework

- Solve problem 22, chapter 2 in your book.
- Start thinking about the project
 - What system would you like to simulate, what will be your simulation objectives
 - Group members