STEVENS INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY®

Discrete Event Simulation

SYS-611: Simulation and Modeling

Paul T. Grogan, Ph.D.
Assistant Professor
School of Systems and Enterprises

Agenda

1. Queuing Model, Revisited
2. DES Model Constructs
3. DES World Views

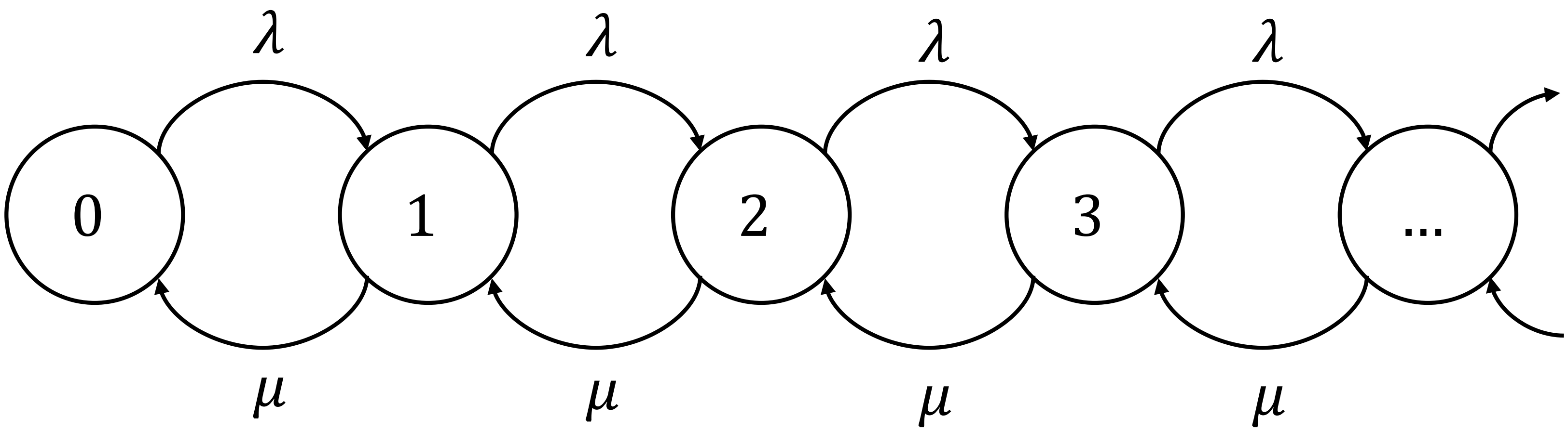
Reading: S.M. Ross "The Discrete Event Simulation Approach," Ch. 7 in *Simulation*, 5th Edition, 2013.

Optional: A.M. Law, "Basic Simulation Modeling," Ch. 1 in *Simulation Modeling and Analysis, 5th Edition, 2014.*

Queuing Model, Revisited

Queuing Model Review

- Continuous-time Markov Process
- State q measures number customers in system
- Inter-arrival periods ($X \sim \text{exponential}(\lambda)$)
- Service times ($Y \sim \text{exponential}(\mu)$)



Results of Queuing Theory

- Steady-state assumption helps to solve for the stationary stochastic distribution P_i in terms of P_0
- Geometric series converges to analytic result

$$\bar{L}_q = \frac{\lambda^2}{\mu(\mu - \lambda)}, \qquad \bar{L} = \frac{\lambda}{\mu - \lambda}$$

$$\bar{W}_q = \frac{\lambda}{\mu(\mu - \lambda)}, \qquad \bar{W} = \frac{1}{\mu - \lambda}$$

M/M/1 Model Simulation

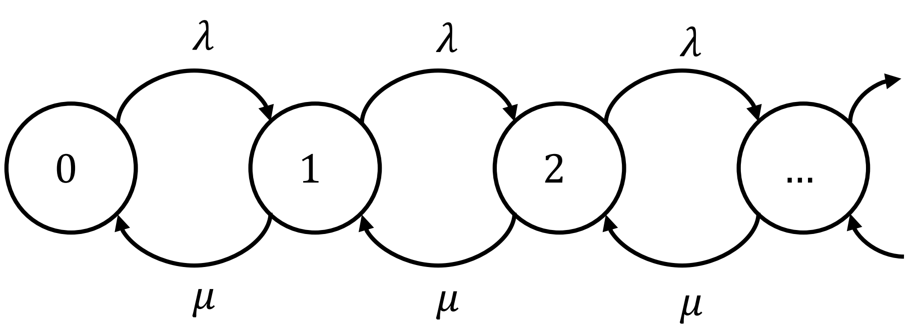
If $q(t) = 0$:

$$q(t + x) = 1$$

If $q(t) > 0$:

$$q(t + \Delta t) = \begin{cases} q(t) + 1 \text{ if } x < y \\ q(t) - 1 \text{ if } y < x \end{cases}$$

$$\Delta t = \min(\mathrm{x}, \mathrm{y})$$



$$x = -\frac{\ln(1 - r_x)}{\lambda}, \lambda = 2/3$$

$$y = -\frac{\ln(1 - r_y)}{\mu}, \mu = 4/3$$

State transition function:

If $q(t) = 0$ or $x < y$:

$$q(t + x) = q(t) + 1$$

If $q(t) > 0$ and $y < x$:

$$q(t + y) = q(t) - 1$$

M/M/1 Model Limitations

- Simplified for mathematical convenience:
- Exponentially-distributed arrivals, service times
  - Independent with long-term average rates
  - Memoryless: can resample x, y for every event
- Single queue, single server
  - Simplifies state representation
- No human behaviors (reneging, balking, etc.)
  - Simplifies state transitions

Customer-based Simulation

Allow different arrival and service duration models by indexing states with customer i instead of event i

- t_{enter}^{i}: time customer i enters the system
- L_q^i: length of the queue when customer i enters
- t_{serv}^{i}: time customer i service starts
- W_q^i: waiting time in the queue for customer i
- t_{exit}^{i}: time customer i exists the system
- W^i: total waiting time for customer i

Customer-based Simulation

x: Inter-arrival Period
y: Service Time

t_{enter}: Entry Time
t_{serv}: Time Served
t_{exit}: Exit Time

L_q: Queue Length
W_q: Queue Wait Time
W: Total Wait Time

i	x	t_{enter}	L_q	t_{serv}	W_q	y	t_{exit}	W
1	**0.43**					**0.90**		
2	**4.49**					**1.66**		
3	**0.95**					**0.02**		
4	**0.03**					**0.46**		
5	**0.28**					**0.51**		

Customer-based Simulation

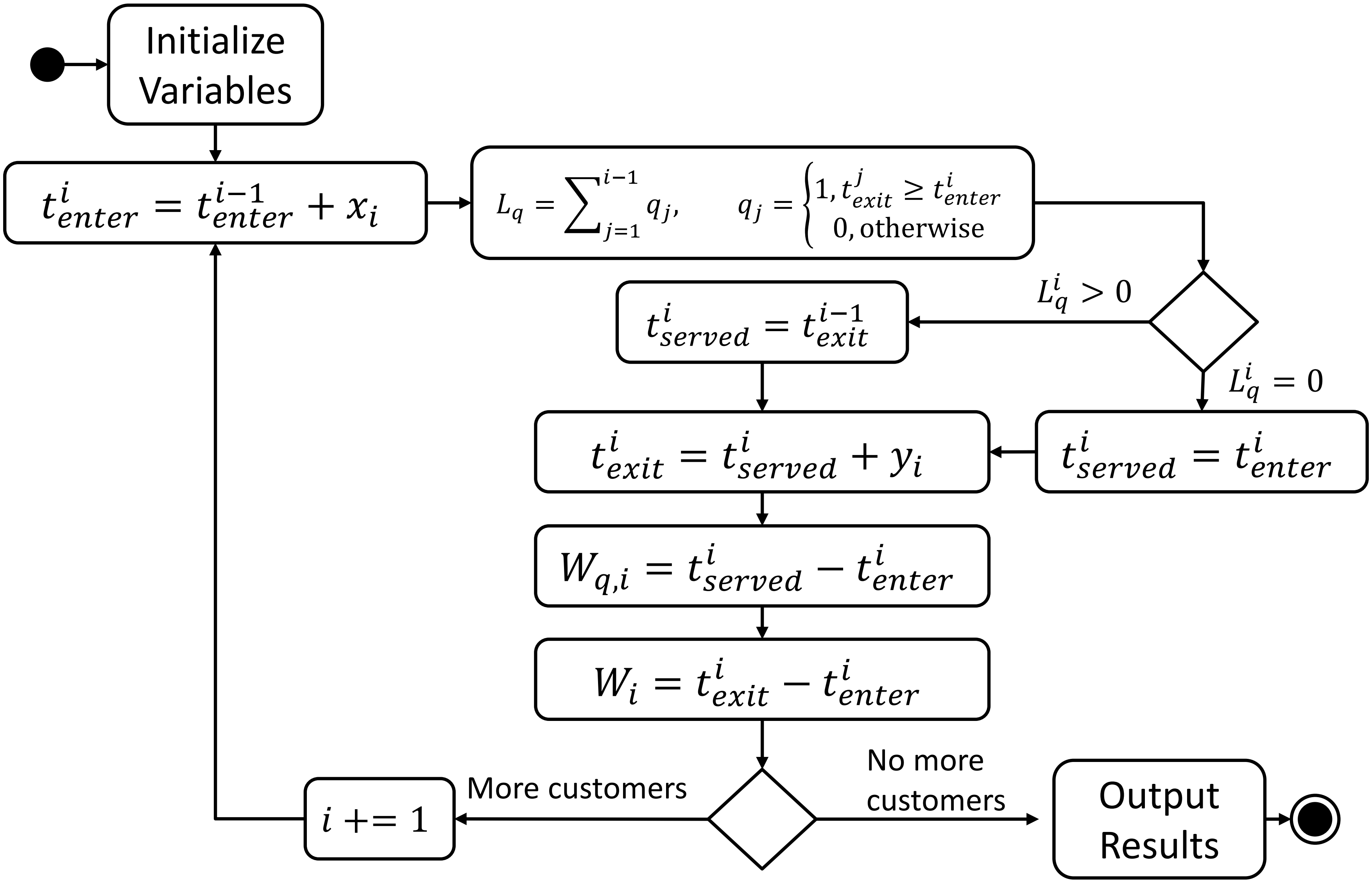
x: Inter-arrival Period
y: Service Time

t_{enter}: Entry Time
t_{serv}: Time Served
t_{exit}: Exit Time

L_q: Queue Length
W_q: Queue Wait Time
W: Total Wait Time

i	x	t_{enter}	L_q	t_{serv}	W_q	y	t_{exit}	W
1	**0.43**	0.00+0.43= 0.43	0	0.43	0.43-0.43= 0	**0.90**	0.43+0.90= 1.33	1.33-0.43= 0.90
2	**4.49**	0.43+4.49= 4.92	0	4.92	4.92-4.92= 0	**1.66**	4.92+1.66= 6.58	6.58-4.92= 1.66
3	**0.95**	4.92+0.95= 5.87	1	6.58	6.58-5.87= 0.71	**0.02**	6.58+0.02= 6.60	6.60-5.87= 0.73
4	**0.03**	5.87+0.03= 5.90	2	6.60	6.60-5.90= 0.70	**0.46**	6.60+0.46= 7.06	7.06-5.90= 1.16
5	**0.28**	5.90+0.28= 6.18	3	7.06	7.06-6.18= 0.88	**0.51**	7.06+0.51= 7.57	7.57-6.18= 1.39

Customer-based Sim Behavior



Customer-based Sim (Excel)

	A	B	C	D	E	F	G	H	I	J	
1	Customer	x	t_enter	L_q	t_served	y	t_exit	W_q	W		L_q_
2	1	0.44	0.44	0	0.44	2.21	2.64	2.21	2.21		
3	2	4.25	4.69	0	4.69	0.13	4.81	0.13	0.13		
4	3	0.74	=B4+C3	0	5.43	0.57	6.00	0.57	0.57		

$$t_{enter}^{i} = t_{enter}^{i-1} + x_i$$

	A	B	C	D	E	F	G	H	I
1	Customer	x	t_enter	L_q	t_served	y	t_exit	W_q	W
2	1	0.44	0.44	0	0.44	2.21	2.64	2.21	2.21
3	2	4.25	4.69	0	4.69	0.13	4.81	0.13	0.13
4	3	0.74	5.43	=COUNTIF(G2:G3,">="&C4)			6.00	0.57	0.57

$$L_q = \sum_{j=1}^{i-1} q_j, \qquad q_j = \begin{cases} 1, t_{exit}^{j} \geq t_{enter}^{i} \\ 0, \text{otherwise} \end{cases}$$

	A	B	C	D	E	F	G	H	I
1	Customer	x	t_enter	L_q	t_served	y	t_exit	W_q	W
2	1	0.39	0.39	0	0.39	0.71	1.11	0.71	0.71
3	2	0.24	0.63	1	1.11	0.24	1.35	0.24	0.71
4	3	3.05	3.68	0	=IF(D4=0,C4,G3)		3.73	0.05	0.05
5	4	1.91	5.59	0	5.59	0.44	6.03	0.44	0.44

$L_q^i > 0$

$$t_{served}^{i} = t_{exit}^{i-1}$$

$L_q^i = 0$

$$t_{served}^{i} = t_{enter}^{i}$$

	A	B	C	D	E	F	G	H	I	J	
1	Customer	x	t_enter	L_q	t_served	y	t_exit	W_q	W		L_q_
2	1	0.92	0.92	0	0.92	0.64	1.55	0.64	0.64		
3	2	0.39	1.31	1	1.55	0.26	1.82	0.26	0.51		
4	3	2.13	3.44	0	3.44	1.69	=F4+E4	1.69	1.69		

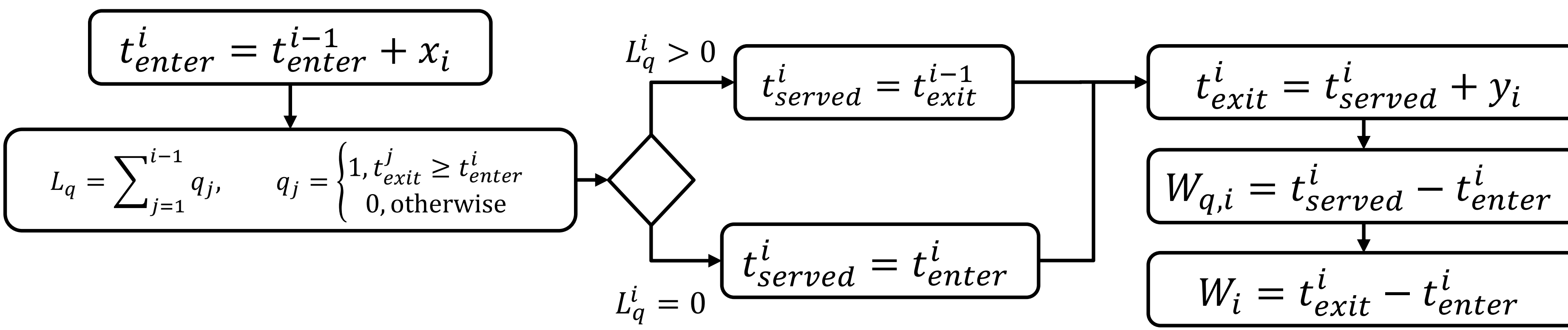
$$t_{exit}^{i} = t_{served}^{i} + y_i$$

	A	B	C	D	E	F	G	H	I	J	
1	Customer	x	t_enter	L_q	t_served	y	t_exit	W_q	W		L_q
2	1	1.62	1.62	0	1.62	0.31	1.93	0.31	0.31		
3	2	1.95	3.56	0	3.56	0.26	3.83	0.26	0.26		
4	3	3.00	6.57	0	6.57	2.34	8.90	=G4-E4	2.34		

$$W_{q,i} = t_{exit}^{i} - t_{served}^{i}$$

Customer-based Sim (Python)

```
for i in range(num_customers):
    t_enter[i] = t_enter[i-1] + x[i] if i > 0 else x[i]
    q_length[i] = np.sum(t_exit[0:i] > t_enter[i]) if i > 0 else 0
    t_served[i] = t_exit[i-1] if q_length[i] > 0 else t_enter[i]
    t_exit[i] = t_served[i] + y[i]
    q_wait[i] = t_served[i] - t_enter[i]
    total_wait[i] = t_exit[i] - t_enter[i]
```



Customer-based Simulation

$\lambda = 2/3$ $\mu = 4/3$

1000-customer Sim.:

- $\bar{W} = 1.49$ min.
- $\bar{W}_q = 0.74$ min.
- $\bar{L}_q{}^* = 1.05$ cust.
- $\bar{L}^* = \bar{L}_q{}^* + 1 = 2.05$ cust.

Queuing Theory:

- $\bar{W} = \frac{1}{\mu-\lambda} = 1.50$ min.
- $\bar{W}_q = \frac{\lambda}{\mu(\mu-\lambda)} = 0.75$ min.
- $\bar{L} = \frac{\lambda}{\mu-\lambda} = 1.00$ cust.
- $\bar{L}_q = \frac{\lambda^2}{\mu(\mu-\lambda)} = 0.50$ cust.

- Length observations *only* recorded at arrivals
- Biased sample, more customers arrive in long lines (by definition)
- Issue of "**random incidence**": need to take independent samples

Customer-based Simulation

Strengths

- Can use arbitrary process generators for all arrival times and service times (if state independent)
- Straight-forward, intuitive state updating

Limitations

- Need to maintain history of all customers
- Difficult to calculate time-average results
- Difficult to extend to new problems:
  - Balking, reneging

DES Model Constructs

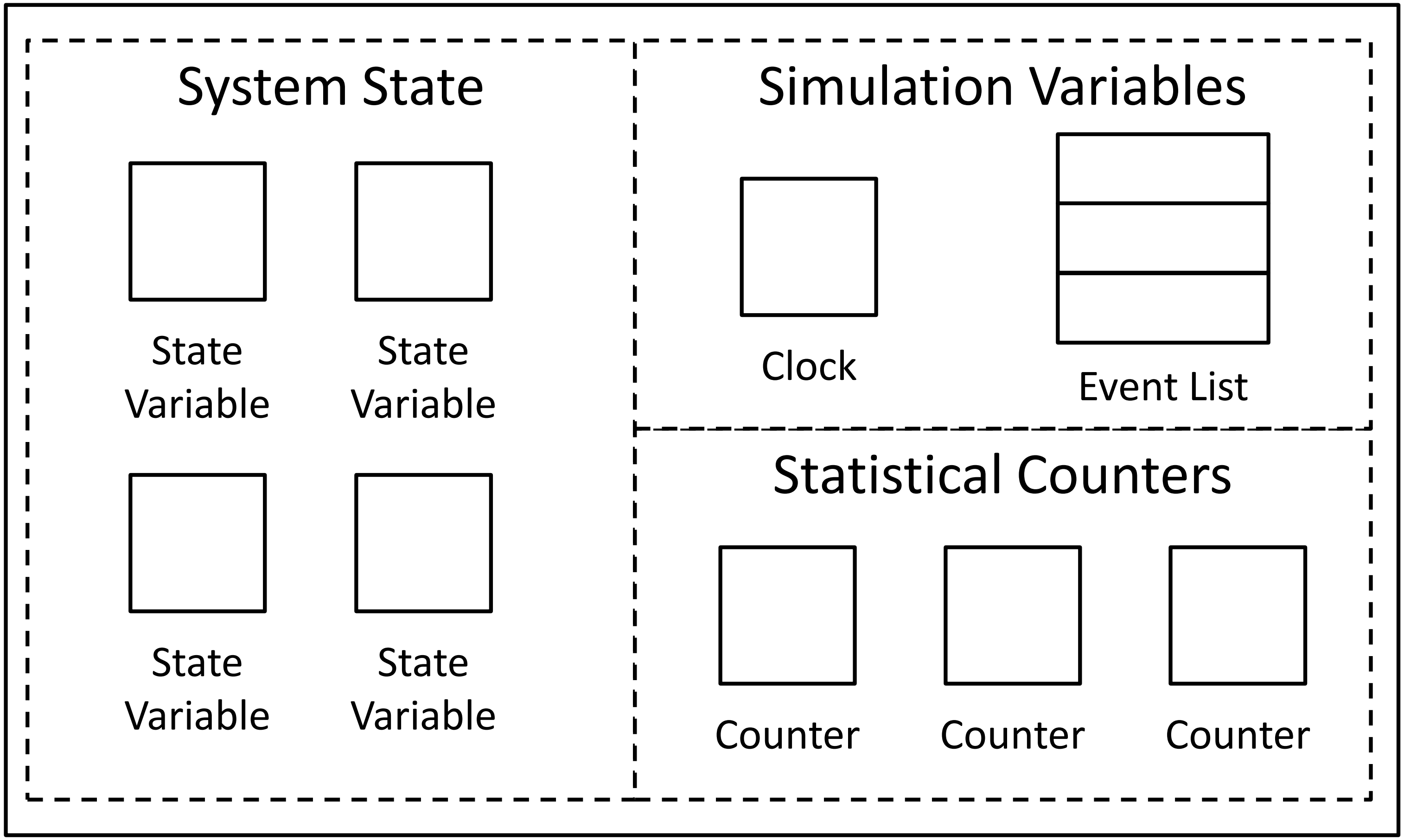
Discrete Event Simulation

- **Discrete event simulation (DES)** is a modeling approach which represents state transitions at particular instants in time (events)
  - Efficiently models systems with occasional state transitions (e.g. logistics, operations)
  - Example: queuing model with general (non-Markov) arrivals and services
- Discrete time simulation is a special type of DES with evenly-distributed events (clock ticks)

DES Model Constructs

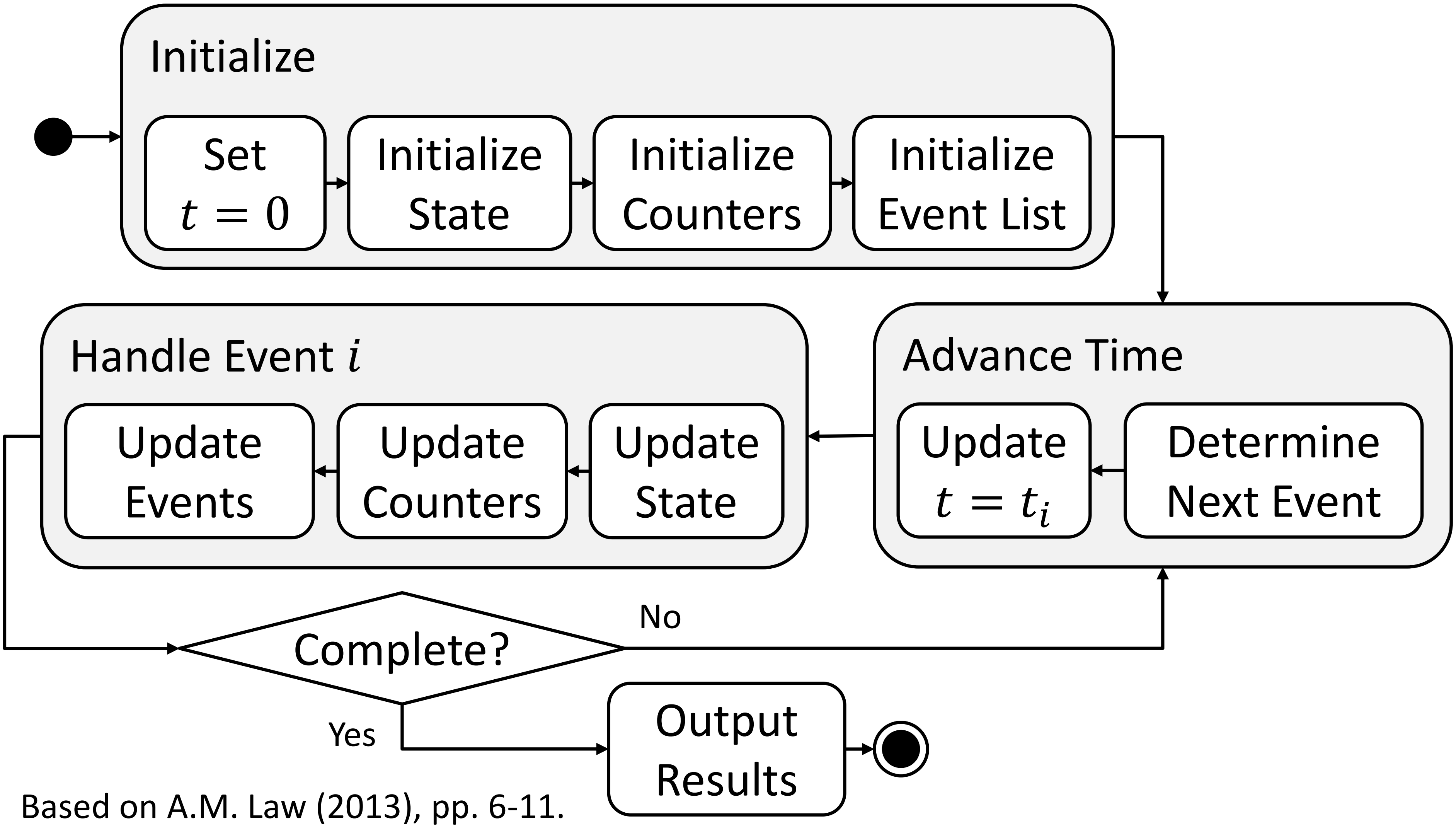
- **Time variable**: keeps track of the current simulation time
- **Event list/stack**: List of future events with associated execution times
- **System state variables**: Stores variables which persist across time
- **Counter variables**: Derived state variables to record useful observations

DES Model Structure



Based on A.M. Law (2013), pp. 6-11.

DES Model Behavior

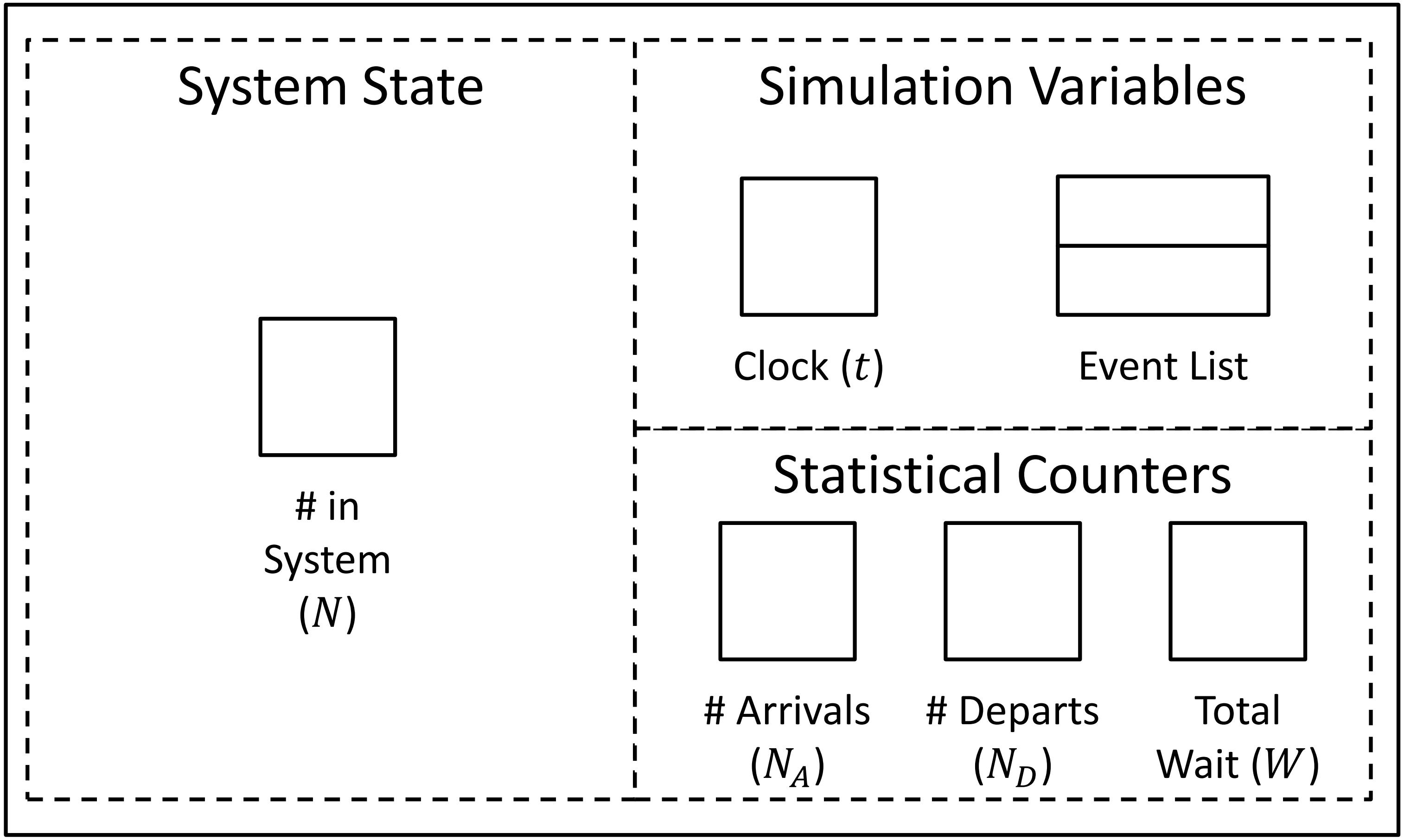


Based on A.M. Law (2013), pp. 6-11.

Event-based Queuing Model

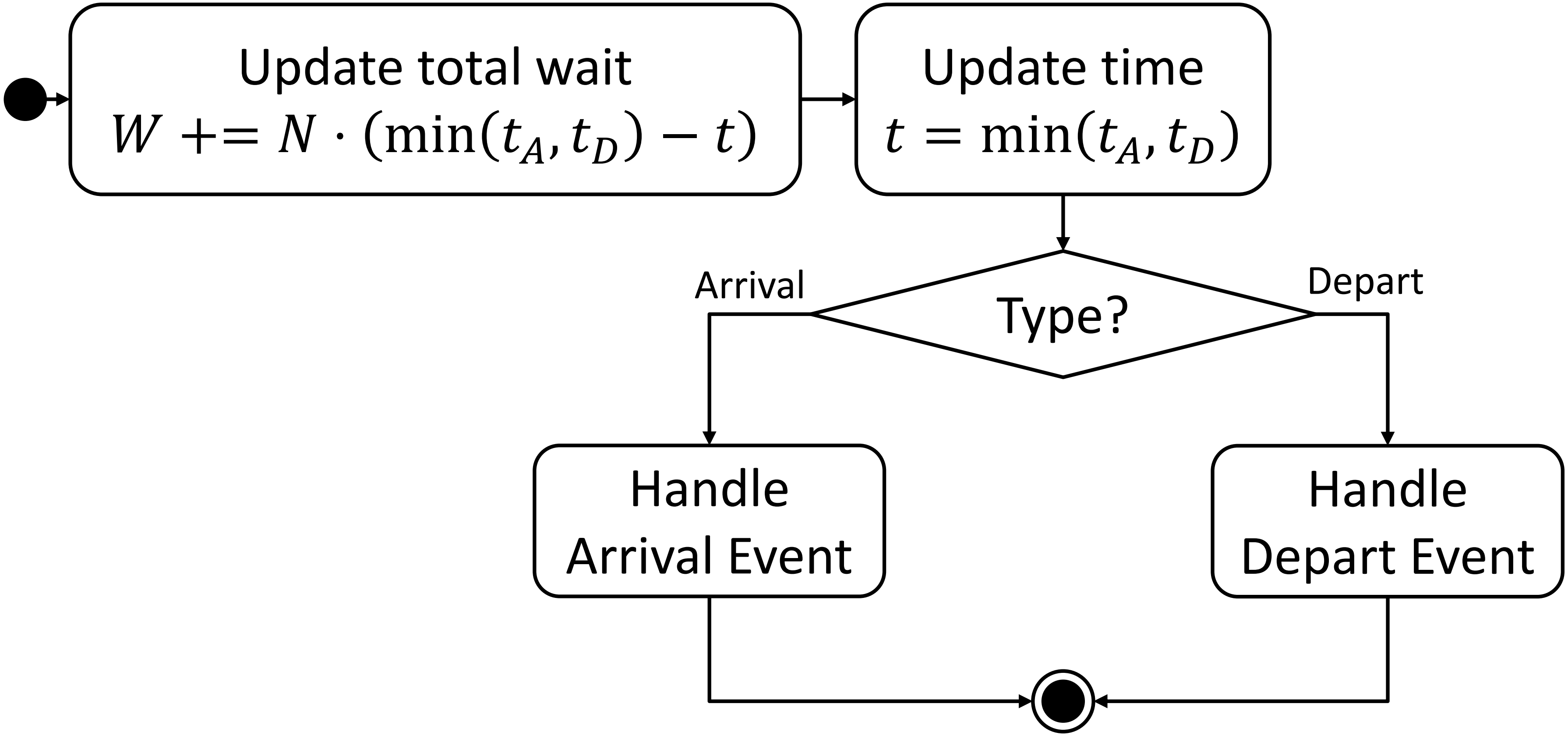
Inter-arrival times: 0.4, 1.2, 0.5, 1.7

Service times: 2.0, 0.7, 0.2, 1.1

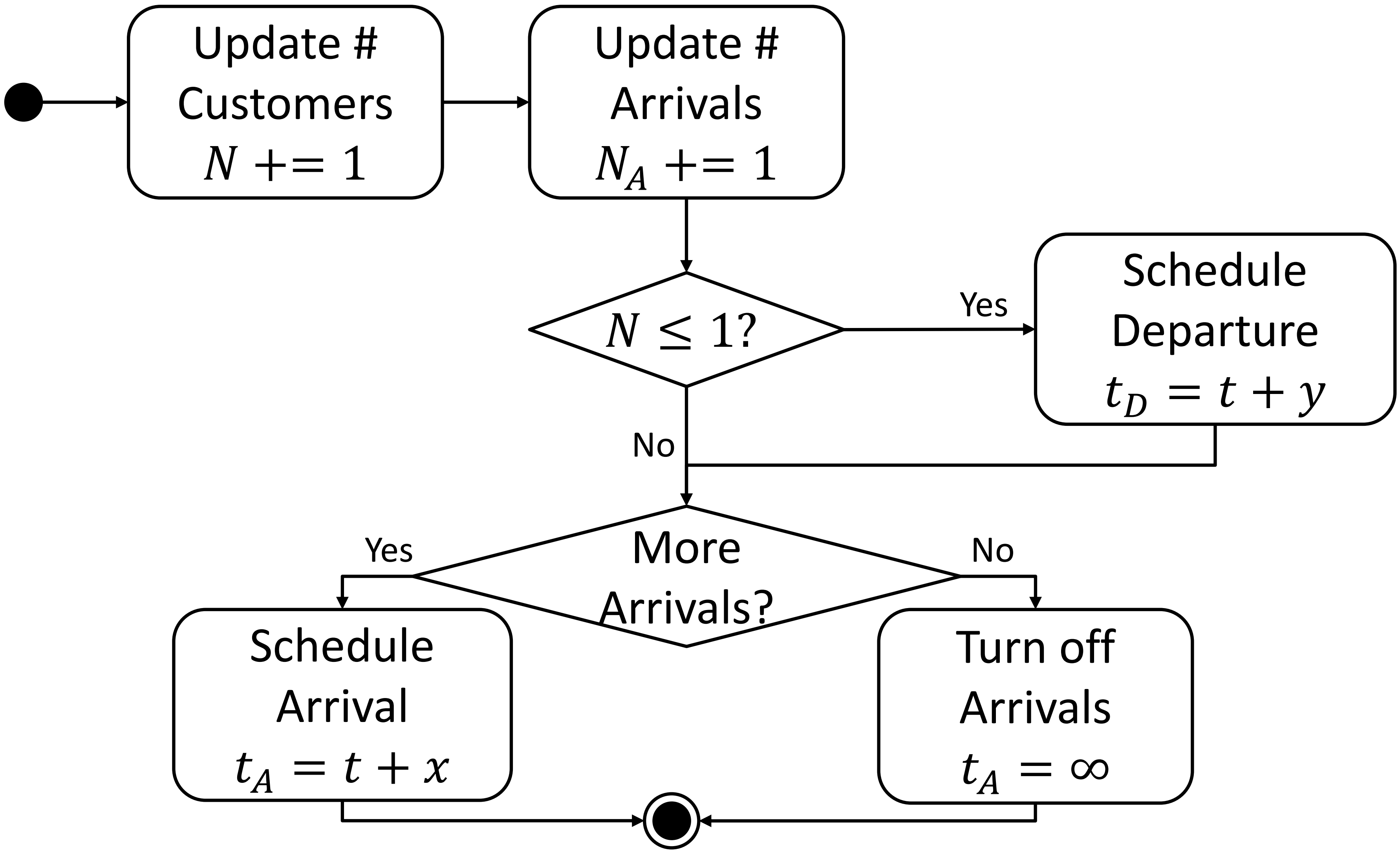


Based on A.M. Law (2013), pp. 12-27.

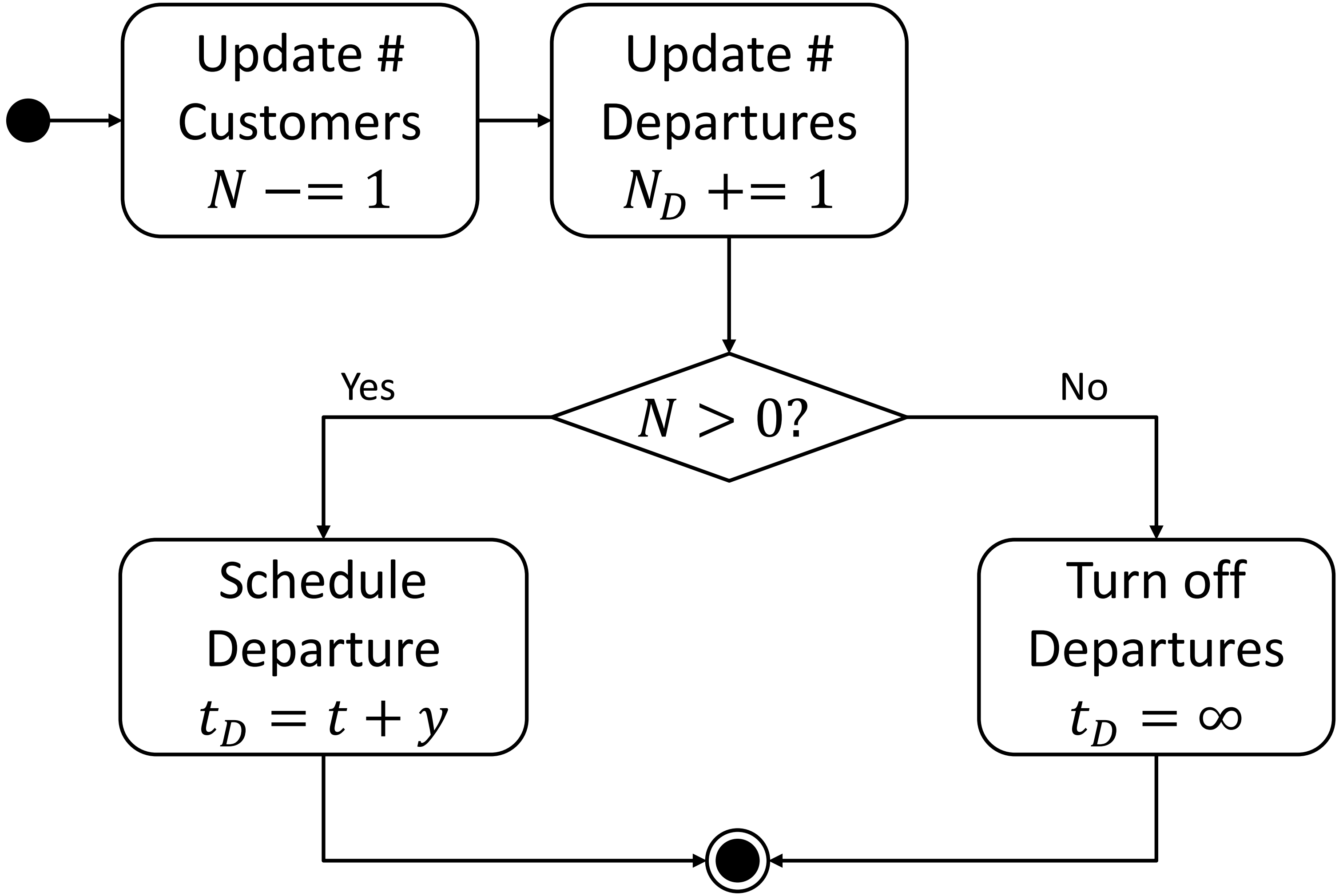
Advance Time / Handle Event



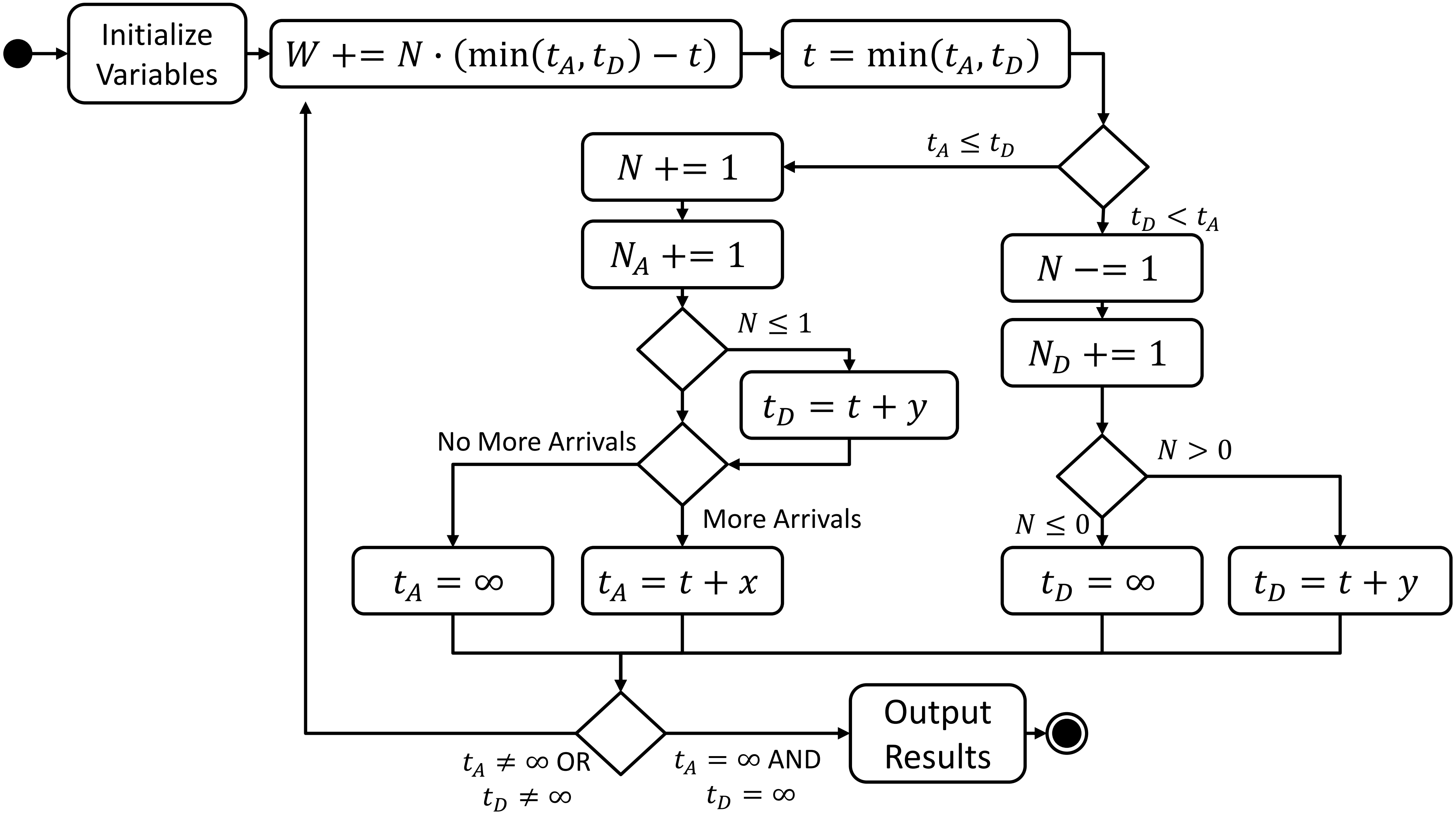
Handle Arrival Event



Handle Departure Event



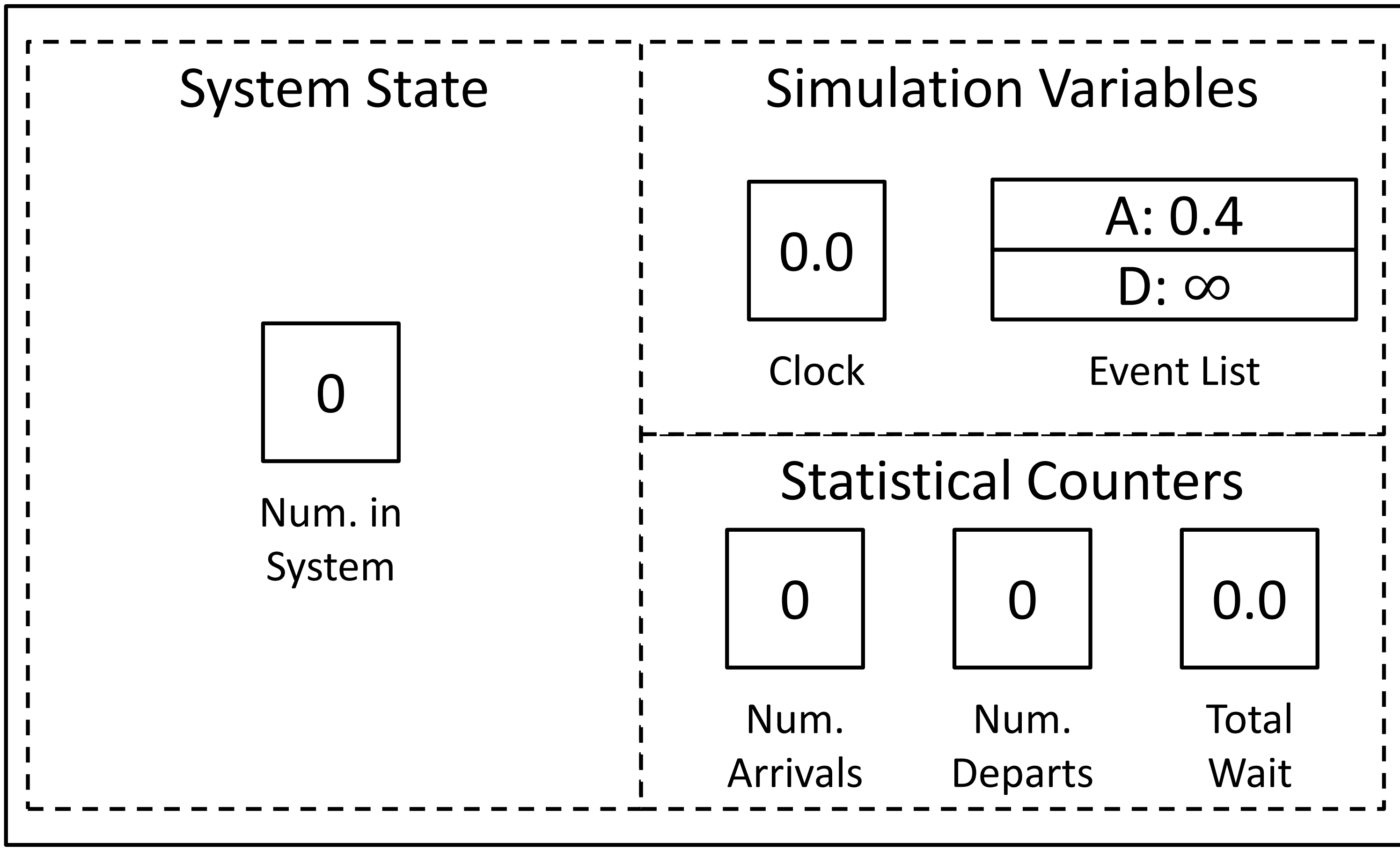
Complete Activity Diagram



Initialize Simulation

Inter-arrival times: 0.4, 1.2, 0.5, 1.7

Service times: 2.0, 0.7, 0.2, 1.1



Based on A.M. Law (2013), pp. 12-27.

Arrival Event @ $t = 0.4$

Inter-arrival times: ~~0.4~~, 1.2, 0.5, 1.7

Service times: 2.0, 0.7, 0.2, 1.1

System State

~~0~~
1

Num. in System

Simulation Variables

~~0.0~~
0.4

Clock

A: ~~0.4~~ 1.6
D: ~~∞~~ 2.4

Event List

Statistical Counters

Num. Arrivals	Num. Departs	Total Wait
~~0~~ 1	0	0.0

Based on A.M. Law (2013), pp. 12-27.

Arrival Event @ $t = 1.6$

Inter-arrival times: ~~0.4, 1.2~~, 0.5, 1.7

Service times: ~~2.0~~, 0.7, 0.2, 1.1

System State

Num. in System: ~~1~~ 2

Simulation Variables

Clock: ~~0.4~~ 1.6

Event List:
- A: ~~1.6~~ 2.1
- D: 2.4

Statistical Counters

Num. Arrivals	Num. Departs	Total Wait
~~1~~ 2	0	~~0.0~~ 1.2

Based on A.M. Law (2013), pp. 12-27.

Arrival Event @ $t = 2.1$

Inter-arrival times: ~~0.4, 1.2, 0.5~~, 1.7 Service times: ~~2.0~~, 0.7, 0.2, 1.1

System State

~~2~~
3

Num. in System

Simulation Variables

~~1.6~~
2.1

Clock

A: ~~2.1~~ 3.8
D: 2.4

Event List

Statistical Counters

Num. Arrivals	Num. Departs	Total Wait
~~2~~ 3	0	~~1.2~~ 2.2

Based on A.M. Law (2013), pp. 12-27.

Departure Event @ $t = 2.4$

Inter-arrival times: ~~0.4, 1.2, 0.5, 1.7~~ Service times: ~~2.0~~, 0.7, 0.2, 1.1

System State

~~3~~
2

Num. in System

Simulation Variables

~~2.1~~
2.4

Clock

A: 3.8
D: ~~2.4~~ 3.1

Event List

Statistical Counters

3

Num. Arrivals

~~0~~
1

Num. Departs

~~2.2~~
3.1

Total Wait

Based on A.M. Law (2013), pp. 12-27.

Departure Event @ $t = 3.1$

Inter-arrival times: ~~0.4, 1.2, 0.5, 1.7~~

Service times: ~~2.0, 0.7~~, 0.2, 1.1

System State

~~2~~ 1

Num. in System

Simulation Variables

~~2.4~~ 3.1

Clock

A: 3.8

D: ~~3.1~~ 3.3

Event List

Statistical Counters

3

Num. Arrivals

~~1~~ 2

Num. Departs

~~3.1~~ 4.5

Total Wait

Based on A.M. Law (2013), pp. 12-27.

Departure Event @ $t = 3.3$

Inter-arrival times: ~~0.4, 1.2, 0.5, 1.7~~

Service times: ~~2.0, 0.7, 0.2~~, 1.1

System State

~~1~~
0

Num. in System

Simulation Variables

~~3.1~~
3.3

Clock

A: 3.8
D: ~~3.3~~ ∞

Event List

Statistical Counters

3

Num. Arrivals

~~2~~
3

Num. Departs

~~4.5~~
4.7

Total Wait

Based on A.M. Law (2013), pp. 12-27.

Arrival Event @ $t = 3.8$

Inter-arrival times: ~~0.4, 1.2, 0.5, 1.7~~ Service times: ~~2.0, 0.7, 0.2, 1.1~~

System State

~~0~~ 1

Num. in System

Simulation Variables

~~3.3~~ 3.8

Clock

A: ~~3.8~~ ∞

D: ~~∞~~ 4.9

Event List

Statistical Counters

Num. Arrivals	Num. Departs	Total Wait
~~3~~ 4	3	4.7

Based on A.M. Law (2013), pp. 12-27.

Departure Event @ $t = 4.9$

Inter-arrival times: ~~0.4, 1.2, 0.5, 1.7~~   Service times: ~~2.0, 0.7, 0.2, 1.1~~

System State

~~1~~
0

Num. in System

Simulation Variables

~~3.8~~
4.9

Clock

A: ∞
D: ~~4.9~~ ∞

Event List

Statistical Counters

4

Num. Arrivals

~~3~~
4

Num. Departs

~~4.7~~
5.8

Total Wait

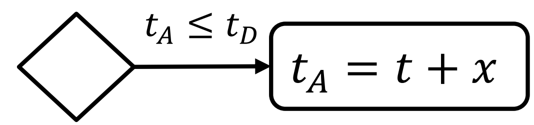
Based on A.M. Law (2013), pp. 12-27.

Event-based Sim (Excel)

	A	B	C	D	E	F	G	H
1	Event	t	t_A	t_D	N	N_A	N_D	W
2	0	0	0.33	9999.00	0	0	0	0.00
3	1	=MIN(C2:D2)		0.80	1	1	0	0.00

$$t_e = \min(t_A, t_D)$$

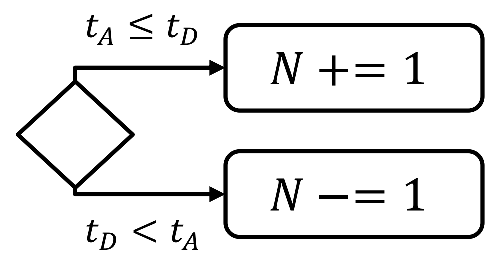
	A	B	C	D	E	F	G	H
1	Event	t	t_A	t_D	N	N_A	N_D	W
2	0	0	0.77	9999.00	0	0	0	0.00
3	1	0.77	=IF(C2<=D2,B3-1.5*LN(1-RAND()),C2)				0	0.00



	A	B	C	D	E	F	G	H	I	J	K
1	Event	t	t_A	t_D	N	N_A	N_D	W		W_bar	
2	0	0	0.04	9999.00	0	0	0	0.00		1.84	
3	1	0.04	0.88	=IF(OR(AND(C2<=D2,E2+1<=1),AND(D2<C2,E2-1>0)),B3-0.75*LN(1-RAND()),IF(AND(D2<							
4	2	0.88	7.96	C2,E2-1<=0),9999,D2))							

Complicated!!

	A	B	C	D	E	F	G	H
1	Event	t	t_A	t_D	N	N_A	N_D	W
2	0	0	2.88	9999.00	0	0	0	0.00
3	1	2.88	3.00	4.63	=IF(C2<=D2,E2+1,E2-1)			0.00
4	2	3.00	4.62	4.63	2	2	0	0.12

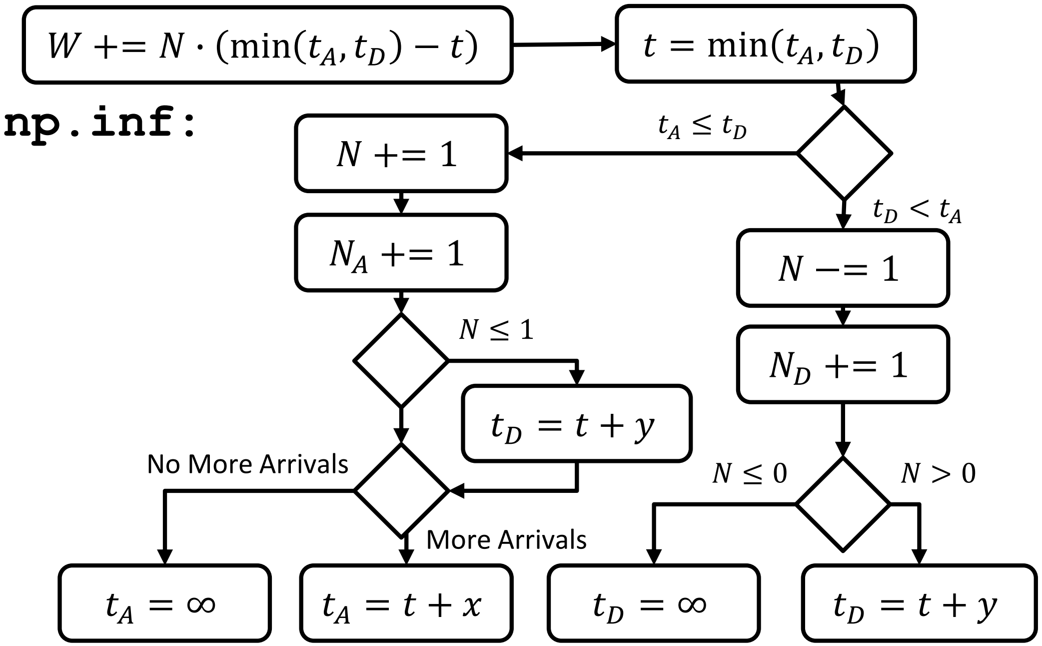


	A	B	C	D	E	F	G	H	I
1	Event	t	t_A	t_D	N	N_A	N_D	W	
2	0	0	2.56	9999.00	0	0	0	0.00	
3	1	2.56	4.36	3.48	1	1	0	=H2+E2*(B3-B2)	
4	2	3.48	4.36	9999.00	0	1	1	0.93	

$$W \mathrel{+}= N \cdot (\min(t_A, t_D) - t)$$

Event-based Sim (Python)

```
while t_A < np.inf or t_D < np.inf:
  W += N*(min(t_A, t_D) - t)
  t  = min(t_A, t_D)
  if t_A <= t_D:
    N += 1
    N_A += 1
    if N <= 1:
      t_D = t + generate_y()
    t_A = t + generate_x() if t < 1000 else np.inf
  else:
    N -= 1
    N_D += 1
    t_D = t + generate_y() if N > 0 else np.inf
```



Discrete Event Simulation

Strengths

- Generate random variables exactly when needed
  - Function of time/state
- Scales well to many types of events
- Only need state from previous time step

Limitations

- Difficult to calculate time-average results (need counter variables)
- State updates can be complex for atomic functions (Excel)

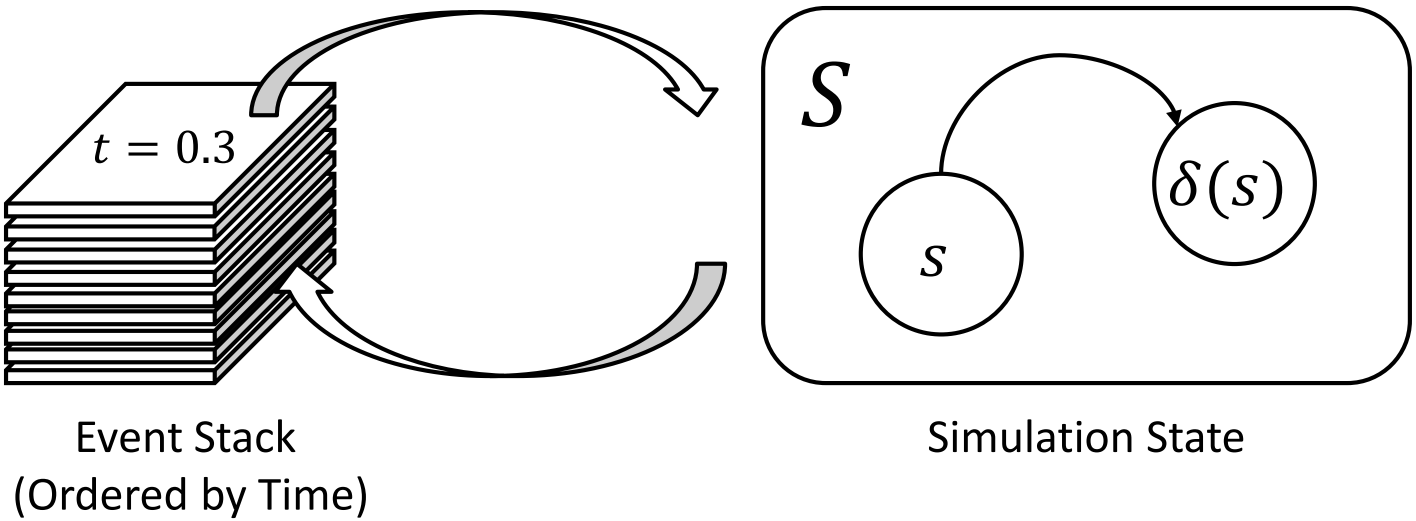
DES World Views

DES World Views

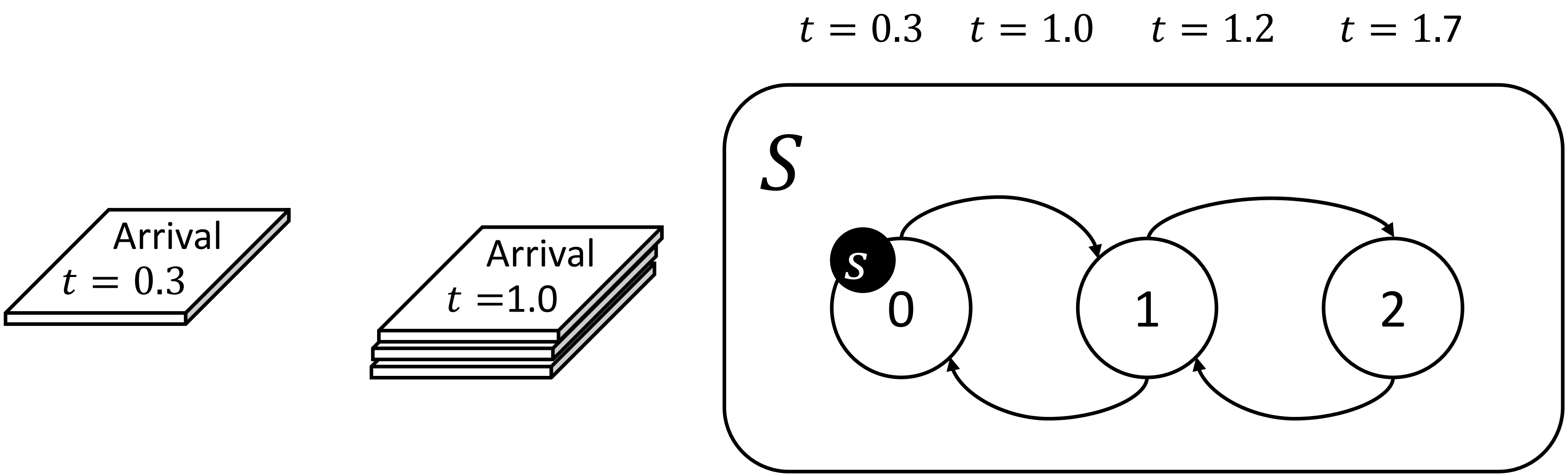
- Three common types of simulation strategies in discrete event simulation (DES):
  - Event-Scheduling
  - Activity Scanning
  - Process Interaction
- Each matches a particular view for how the world works (i.e. world view)

Event-Scheduling World View

- **Events** are building blocks
- Schedule all events in an event stack
- Event execution performs state transition



Event-Scheduling Example



Activity Scanning World View

- **Activities** are building blocks
- Define a sequence of activities waiting to be executed with required conditions
- Scan for activities at each time step

Activity 1
- Precondition 1a
- Precondition 1b
- Duration
- Postcondition 1a
- Postcondition 1b

Activity 2
- Precondition 2a
- Precondition 2b
- Duration
- Postcondition 2a
- Postcondition 2b

Activity Scanning Example

Inter-arrival 1

- Duration: 0.3
- Post: $\delta(s) = s + 1$

Inter-arrival 2

- Pre: Inter-arrival 1 complete
- Duration: 0.7
- Post: $\delta(s) = s + 1$

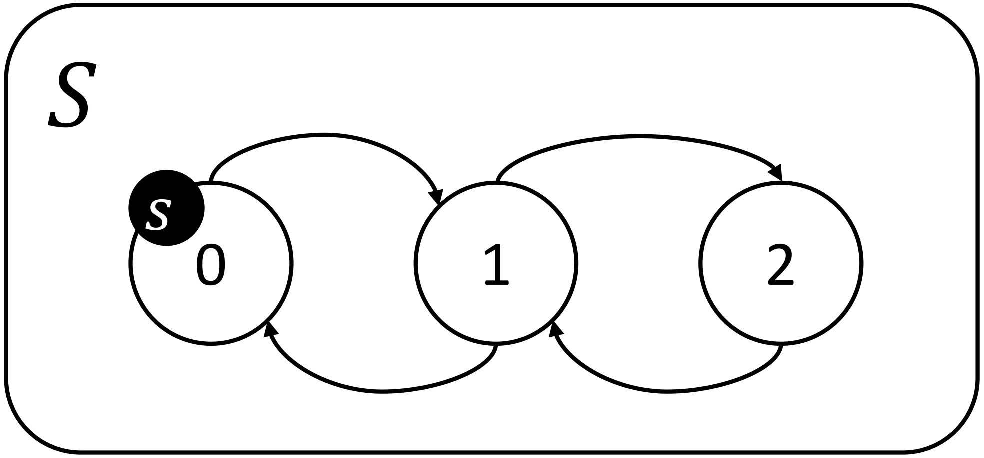
Service 1

- Pre: Inter-arrival 1 complete
- Duration: 0.9
- Post: $\delta(s) = s - 1$

Service 2

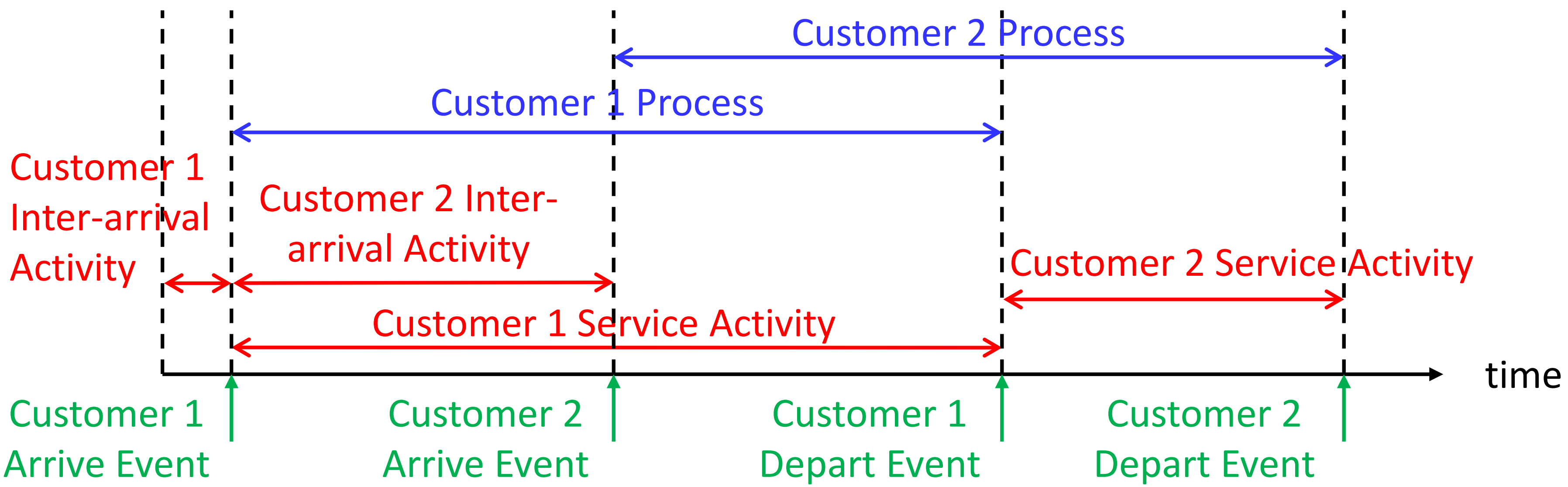
- Pre: Service 1 complete
- Pre: Inter-arrival 2 complete
- Duration: 0.5
- Post: $\delta(s) = s - 1$

$t = 0.3$ $t = 1.0$ $t = 1.2$ $t = 1.7$



Process Interaction World View

- **Processes** are building blocks
- Analogous to processes in an operating system
- Define a sequence of actions for the life-cycle of each model component



Process Interaction Example

Customer 1
- Arrive at 0.3
- Request server resource
- Hold resource for 0.9
- Release server resource
- Depart

Customer 2
- Arrive at 1.0
- Request server resource
- Hold resource for 0.5
- Release server resource
- Depart

$t = 0.3$ $t = 1.0$ $t = 1.2$ $t = 1.7$