UNIT – 2 GENERAL PRINCIPLES, SIMULATION SOFTWARE: Concepts in Discrete-Event Simulation: The Event-Scheduling / Time-Advance Algorithm, World Views, Manual simulation Using Event scheduling; List processing. Simulation in Java; Simulation in GPSS. 6 Hours

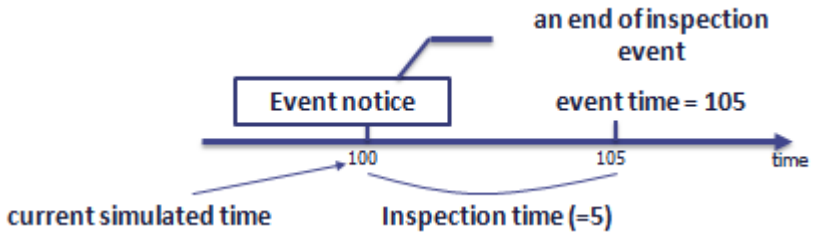
Discrete-event simulation

- The basic building blocks of all discrete-event simulation models: entities and attributes, activities and events.
- A system is modeled in terms of
  - Its state at each point in time
  - The entities that pass through the system and the entities that represent system resources
  - The activities and events that cause system state to change.
- Discrete-event models are appropriate for those systems for which changes in system state occur only at discrete points in time.
- This chapter deals exclusively with dynamic, stochastic systems (i.e., involving time and containing random elements) which change in a discrete manner.

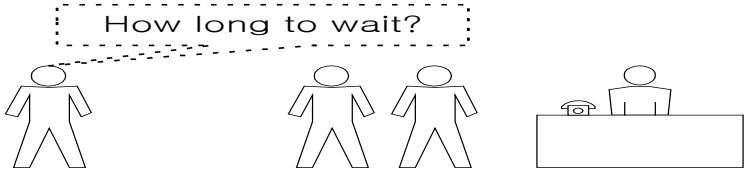
Concepts in Discrete-Event Simulation

1. **System:** A collection of entities (e.g., people and machines) that together over time to accomplish one or more goals.
2. **Model:** An abstract representation of a system, usually containing structural, logical, or mathematical relationships which describe a system in terms of state, entities and their attributes, sets, processes, events, activities, and delays.
3. **System state:** A collection of variables that contain all the information necessary to describe the system at any time.
4. **Entity:** Any object or component in the system which requires explicit representation in the model (e.g., a server, a customer, a machine).
5. **Attributes:** The properties of a given entity (e.g., the priority of a v customer, the routing of a job through a job shop).
6. **List:** A collection of (permanently or temporarily) associated entities ordered in some logical fashion (such as all customers currently in a waiting line, ordered by first come, first served, or by priority).
7. **Event:** An instantaneous occurrence that changes the state of a system as an arrival of a new customer).
8. **Event notice:** A record of an event to occur at the current or some future time, along with any associated data necessary to execute the event; at a minimum, the record includes the event type and the event time.
9. **Event list:** A list of event notices for future events, ordered by time of occurrence; also known as the future event list (FEL).
10. **Activity:** A duration of time of specified length (e.g., a service time or arrival time), which is known when it begins (although it may be defined in terms of a statistical distribution).
11. **Delay:** A duration of time of unspecified indefinite length, which is not known until it ends (e.g., a customer's delay in a last-in, first-out waiting line which, when it begins, depends on future arrivals).
12. **Clock:** A variable representing simulated time.

- The future event list is ranked by the event time recorded in the event notice.
- An **activity** typically represents a service time, an interarrival time, or any other processing time whose duration has been characterized and defined by the modeler.
- **An activity's duration may be specified in a number of ways:**
  1. **Deterministic**-for example, always exactly 5 minutes;
  2. **Statistical**-for example, as a random draw from among 2, 5, 7 with equal probabilities;
  3. A **function** depending on system variables and/or entity attributes-for example, loading time for an iron ore ship as a function of the ship's allowed cargo weight and the loading rate in tons per hour.
- **The duration of an activity** is computable from its specification at the instant it begins.
- To keep track of activities and their expected completion time, at the simulated instant that activity duration begins, an event notice is created having an **event time equal to the activity's completion time.**



- **A delay's duration**
  - Not specified by the modeler ahead of time, but rather determined by system conditions.
  - Quite often, a delay's duration is measured and is one of the desired outputs of a model run.
- A customer's delay in a waiting line may be dependent on the number and duration of service of other customers ahead in line as well as the availability of servers and equipment.



	Delay	Activity
What so called	a conditional wait	an unconditional wait
A completion	a secondary event	a primary event
A management	by placing an event notice on the FEL	by placing the associated entity on another list, not the FEL, perhaps representing a waiting line.

- System state, entity attributes and the number of active entities, the contents of sets, and the activities and delays currently in progress are all functions of time and are constantly changing over time.
- Time itself is represented by a variable called **CLOCK**.

EXAMPLE (Able and Baker, Revisited)

Consider the Able-Baker carhop system. A discrete- event model has the following components:

System state

1. **LQ(t), the number of cars waiting to be served at time t**
2. **LA(t), 0 or 1 to indicate Able being idle or busy at time t**
3. **LB (t), 0 or 1 to indicate Baker being idle or busy at time t**

Entities

Neither the customers (i.e., cars) nor the servers need to be explicitly represented, except in terms of the state variables, unless certain customer averages are desired.

Events

1. Arrival event
2. Service completion by Able
3. Service completion by Baker

Activities

1. Interarrival time, defined in Table 2.11
2. Service time by Able, defined in Table 2.12
3. Service time by Baker, defined in Table 2.13

Delay

A customer's wait in queue until Able or Baker becomes free.

- The definition of the model components provides a static description of the model.
- A description of the dynamic relationships and interactions between the components is also needed.
- A discrete-event simulation: the modeling over time of a system all of whose state changes occur at discrete points in time-those points when an event occurs.
- A discrete-event simulation proceeds by producing a sequence of system snapshots (or system images) which represent the evolution of the system through time.

Clock	*System State*	*Entities and Attributes*	*Set 1*	*Set 2*	...	*Future Event List, FEL*	*Cumulative Statistics and Counters*
t	$(x, y, z, \ldots)$					$(3, t_1)$— Type 3 event to occur at time t_1 $(1, t_2)$— Type 1 event to occur at time t_2	

Figure 2.1 Prototype system snapshot at simulation time t

The Event-Scheduling/Time-Advance Algorithm

- **The mechanism for advancing simulation time and guaranteeing that all events occur in correct chronological order is based on the future event list (FEL).**
- **Future Event List (FEL)**
  - **To contain all event notices for events that have been scheduled to occur at a future time.**
  - **To be ordered by event time,** meaning that the events are arranged chronologically; that is, the event times satisfy.

$$t < t_1 \le t_2 \le \quad \cdots \quad \le t_n$$

current value of simulated time

Imminent event

- **Scheduling a future event** means that at the instant an activity begins, its duration is computed or drawn as a sample from a statistical distribution and the end-activity event, together with its event time, is placed on the future event list.

The sequence of actions which a simulator must perform to advance the clock system snapshot is called the event-scheduling/time-advance algorithm.

Old system snapshot at time t

CLOCK	*System State*	...	*Future Event List*	...
t	(5, 1, 6)		$(3, t_1)$ — Type 3 event to occur at time t_1 $(1, t_2)$ — Type 1 event to occur at time t_2 $(1, t_3)$ — Type 1 event to occur at time t_3 . . . . $(2, t_n)$ — Type 2 event to occur at time t_n	

Event-scheduling/time-advance algorithm

Step 1. Remove the event notice for the imminent event (event 3, time t_1) from FEL

Step 2. Advance CLOCK to imminent event time (i.e., advance CLOCK from t to t_1).

Step 3. Execute imminent event: update system state, change entity attributes, and set membership as needed.

Step 4. Generate future events (if necessary) and place their event notices on FEL ranked by event time. (*Example:* Event 4 to occur at time t^*, where $t_2 < t^* < t_3$.)

Step 5. Update cumulative statistics and counters.

New system snapshot at time t_1

CLOCK	*System State*	...	*Future Event List*	...
t_1	(5, 1, 5)		$(1, t_2)$ — Type 1 event to occur at time t_2 $(4, t^*)$ — Type 4 event to occur at time t^* $(1, t_3)$ — Type 1 event to occur at time t_3 . . . . $(2, t_n)$ — Type 2 event to occur at time t_n	

Figure 3.2 Advancing simulation time and updating system image.

- **List processing: the management of a list.**
  - **The removal of the imminent event:** As the imminent event is usually at the top of the list, its removal is as efficient as possible.
  - **The addition of a new event to the list, and occasionally removal of some event (called cancellation of an event):** Addition of a new event (and cancellation of an old event) requires a search of the list.
- The **efficiency** of this search depends on the logical organization of the list and on how the search is conducted.
- The **removal and addition of events from the FEL** is illustrated in Figure 3.2.
- The system snapshot at time 0 is defined by the initial conditions and the generation of the so-called exogenous events.
- **An exogenous event:** a happening "outside the system" which impinges on the system.
- The specified initial conditions define the system state at time 0.
- In Figure 3.2, if t = 0, then the state (5, 1, 6) might represent the initial number of customers at three different points in the system.
  - How future events are generated?
  - to generate an arrival to a queueing system
  - by a service-completion event in a queueing simulation
  - to generate runtimes and downtimes for a machine subject to breakdowns

- **To generate an arrival to a queueing system**
  - The end of an interarrival interval is an example of a primary event.

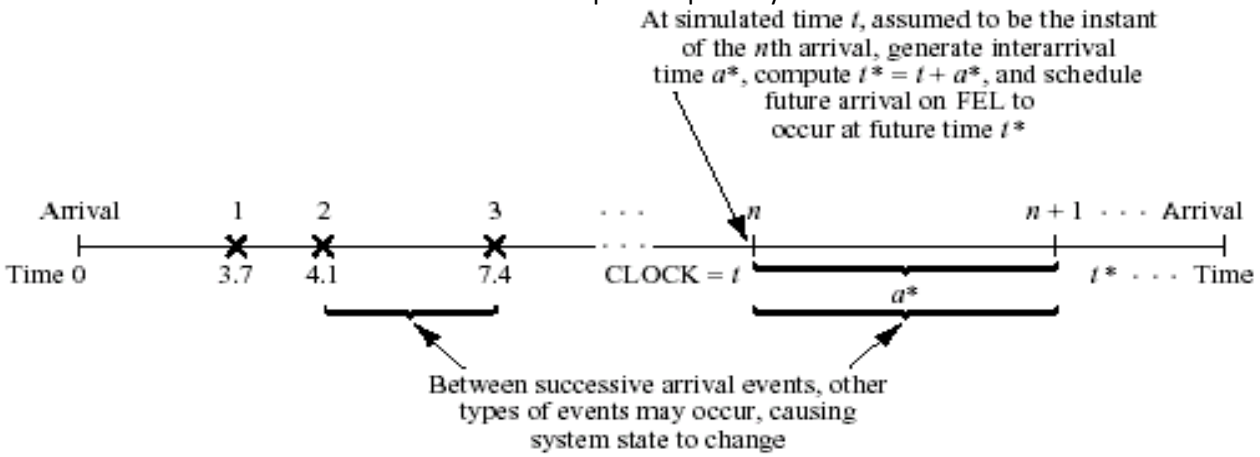


Figure 3.3 Generation of an external arrival stream by bootstrapping.

- By a service-completion event in a queueing simulation
  - A new service time, s^*, will be generated for the next customer.
    - When one customer completes service, at current time CLOCK = t
    - If the next customer is present
      - The next service-completion event will be scheduled to occur at future time $t^* = t + s^*$ by placing onto the FEL a new event notice of type service completion.
- A service-completion event will be generated and scheduled at the time of an arrival event, provided that, upon arrival, there is at least one idle server in the server group.
- **Beginning service:** a conditional event triggered only on the condition a customer is present and a server is free.
- **Service completion:** a primary event.
- **Service time:** an activity
- **By a service-completion event in a queueing simulation.**
  - A **conditional event** is triggered by a primary event occurring.
  - Only **primary events** appear on the FEL.
- **To generate runtimes and downtimes for a machine subject to breakdowns**
  - At time 0, the first runtime will be generated and an end-of-runtime event scheduled.
  - Whenever an end-of-runtime event occurs, a downtime will be generated and an end-of downtime event scheduled on the FEL.
  - When the CLOCK is eventually advanced to the time of this end-of-downtime event, a runtime is generated and an end-of-runtime event scheduled on the FEL.
  - **An end of runtime and an end of downtime:** primary events.
  - **A runtime and a downtime:** activities
- Every simulation must have **a stopping event, called E,** which defines how long the simulation will run.
- There are generally two ways to stop a simulation:
  1. At time 0, schedule a stop simulation event at a specified future time T_E.
     Ex) Simulate a job shop for T_E = 40 hours, that is, over the time interval [0, 40].
  2. Run length T_E is determined by the simulation itself. Generally, T_E is the time of occurrence of some specified event E.
     Ex) the time of the 100^{th} service completion at a certain service center.
     the time of breakdown of a complex system.
     the time of disengagement or total kill in a combat simulation.
     the time at which a distribution center ships the last carton in a day's orders.
- In case 2, T_E is not known ahead of time. Indeed, it may be one of the statistics of primary interest to be produced by the simulation.

World Views

- During simulation, a modeler adopts a **world view or orientation for developing a model**.
- Those most prevalent are **the event scheduling world view, the process-interaction worldview, and the activity-scanning world view.**
  1. **The process-interaction approach**, a simulation analyst thinks in terms of processes.
     - The **process-interaction approach** is popular because of its intuitive appeal, and because the simulation packages that implement it allow an analyst to describe the process flow in terms of high-level block or network constructs.
     - Figure 3.4 shows the interaction between two customer processes as customer n+1 is delayed until the previous customer's "end-service event" occurs.

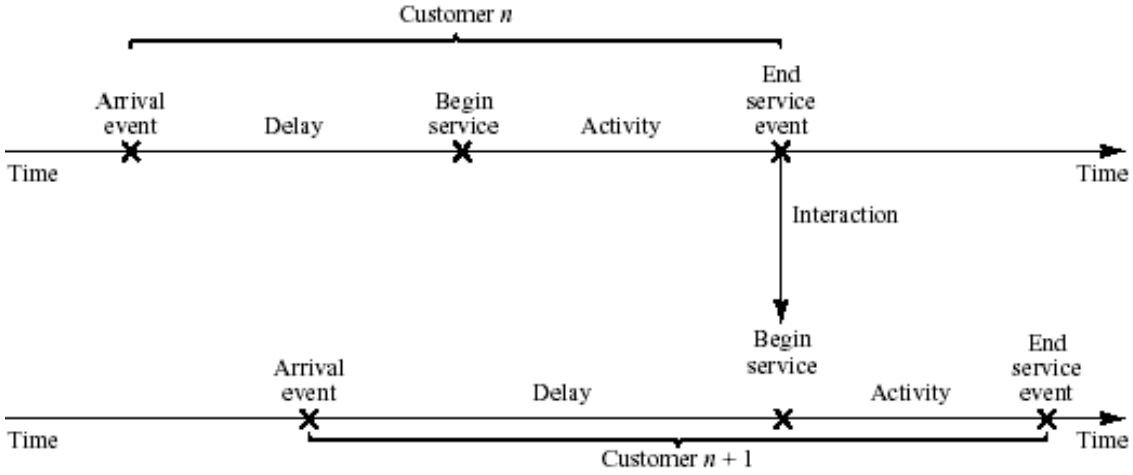


Figure 3.4 Two interacting customer processes in a single-server queue.

  2. When using the **event-scheduling approach**, a simulation **analyst concentrates on events** and their **effect on system state.**
     - Both the event-scheduling and the process-interaction approaches use a variable time advance.
  3. **The activity-scanning approach** uses a fixed time increment and a rule-based approach to decide whether any activities can begin at each point in simulated time.
- **The pure activity scanning approach** has been modified by what is called the **three-phase approach.**
- In the three-phase approach, events are considered to be activity duration-zero time units. With this definition, activities are divided into two categories called B and C.
- **B activities:** Activities bound to occur; all primary events and unconditional activities.
- **C activities:** Activities or events those are conditional upon certain conditions being true.

With the three-phase approach the simulation proceeds with repeated execution of the three phases until it is completed:

1. **Phase A:** Remove the imminent event from the FEL and advance the clock to its event time. Remove any other events from the FEL that have the event time.
2. **Phase B:** Execute all B-type events that were removed from the FEL.
3. **Phase C:** Scan the conditions that trigger each C-type activity and activate any whose conditions are met. Rescan until no additional C-type activities can begin or events occur.

Manual Simulation Using Event Scheduling

In an event-scheduling simulation, a simulation table is used to record the successive system snapshots as time advances.

Let us consider the example of a grocery shop which has only one checkout counter. **(Single-Channel Queue)**

The system consists of those customers in the waiting line plus the one (if any) checking out.

The model has the following components:

System state (LQ (t), LS (t)), where LQ (t) is the number of customers in the waiting line, and LS (t) is the number being served (0 or 1) at time t.

Entities: The server and customers are not explicitly modeled, except in terms of the state variables above.

Events

Arrival (A)

Departure (D)

Stopping event (E), scheduled to occur at time 60.

Event notices

(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

Interarrival time, denned in Table 2.6

Service time, defined in Table 2.7

Delay

Customer time spent in waiting line.

In this model, the FEL will always contain either two or three event notices.

The effect of the arrival and departure events was first shown in Figures 2.2 and 2.3 and is shown in more detail in Figures 3.5 and 3.6.

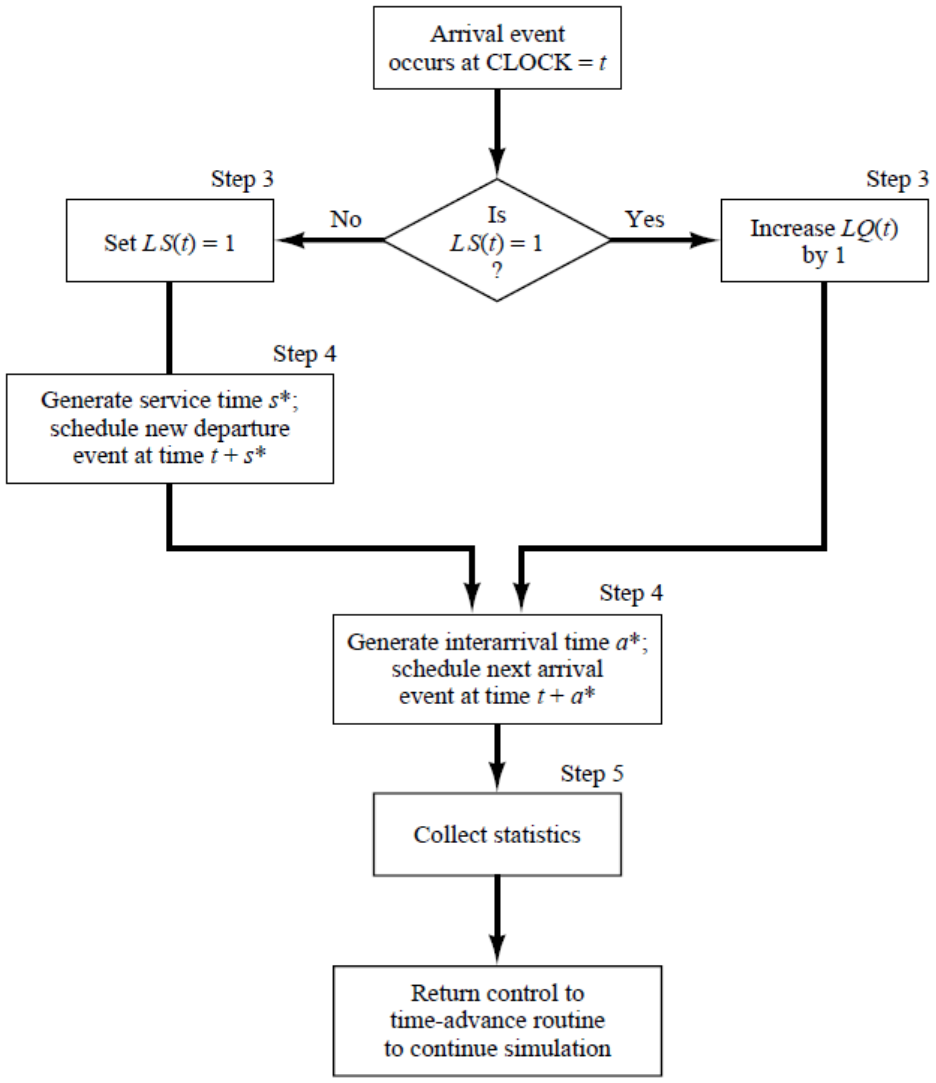


Figure 3.5 Execution of the arrival event.

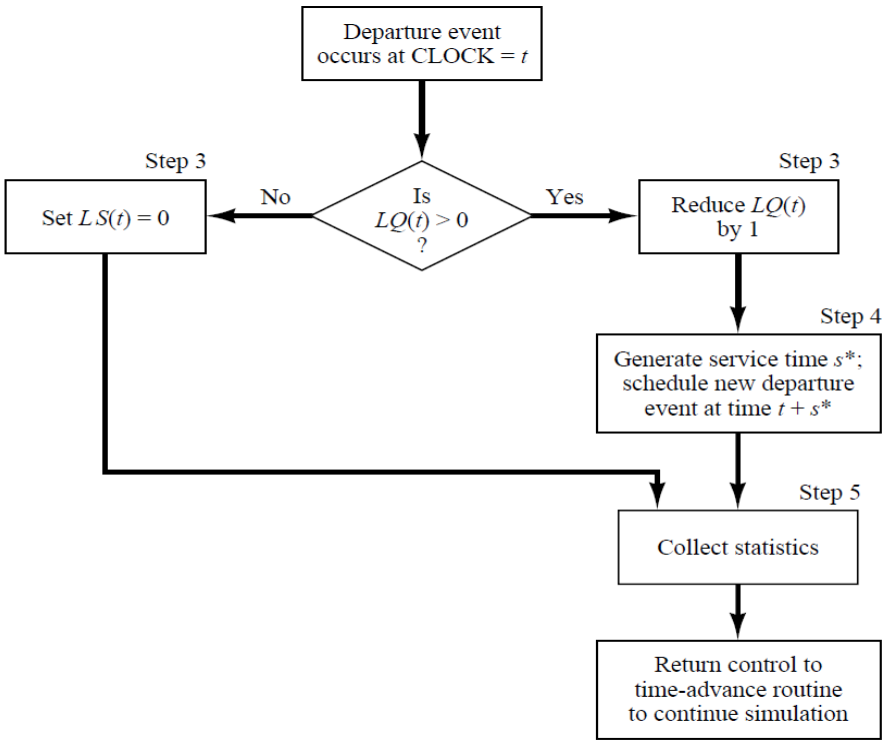


Figure 3.6 Execution of the departure event.

- Example 3.3
  - The interarrival times and service times will be identical to those used in Table 2.10.

Interarrival Times	8	6	1	8	3	8	...
Service Times	4	1	4	3	2	4	...

  - **Initial conditions**
    - the system snapshot at time zero (CLOCK = 0)
    - LQ(0) = 0, LS(0) = 1
    - both a departure event and arrival event on the FEL.
  - The simulation is scheduled to stop at time 60.
  - **Server utilization: total server busy time (B) / total time (T_E).**
  - a^* : the generated interarrival time
  - s^* : the generated service times
  - The simulation in Table 3.1 covers the time interval [0, 21].

Table 3.1 Simulation Table for Checkout Counter (Example 3.3)

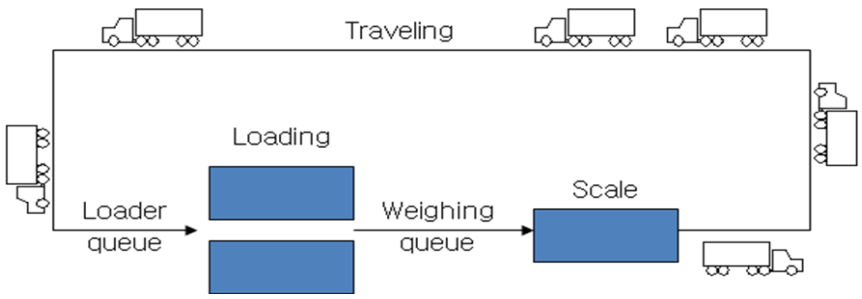
	System State				*Cumulative Statistics*	
Clock	$LQ(t)$	$LS(t)$	*Future Event List*	*Comment*	B	MQ
0	0	1	$(D,4)$ $(A,8)$ $(E,60)$	First A occurs $(a^* = 8)$ Schedule next A $(s^* = 4)$ Schedule first D	0	0
4	0	0	$(A,8)$ $(E,60)$	First D occurs: $(D,4)$	4	0
8	0	1	$(D,9)$ $(A,14)$ $(E,60)$	Second A occurs: $(A,8)$ $(a^* = 6)$ Schedule next A $(s^* = 1)$ Schedule next D	4	0
9	0	0	$(A,14)$ $(E,60)$	Second D occurs: $(D,9)$	5	0
14	0	1	$(A,15)$ $(D,18)$ $(E,60)$	Third A occurs: $(A,14)$ $(s^* = 4)$ Schedule next D	5	0
15	1	1	$(D,18)$ $(A,23)$ $(E,60)$	Fourth A occurs: $(A,15)$ (Customer delayed)	6	1
18	0	1	$(D,21)$ $(A,23)$ $(E,60)$	Third D occurs: $(D,18)$ $(s^* = 3)$ Schedule next D	9	1
21	0	0	$(A,23)$ $(E,60)$	Fourth D occurs: $(D,21)$	12	1

- In Example 3.3, to estimate :
- **Mean response time:** the average length of time a customer spends in the system
- Mean proportion of customers who spend 4 or more minutes in the system.
- **Entities (Ci, t):** representing customer Ci who arrived at time t
- **Event notices:**
  - **(A, t, Ci)**, the arrival of customer Ci at future time t
  - **(D, t, Cj)**, the departure of customer Cj at future time t
- Set : **"CHECKOUTLINE"** the set of all customers currently at the checkout counter (being served or waiting to be served), ordered by time of arrival
- A customer entity with arrival time as an attribute is added in order to estimate mean response time.
- Three new cumulative statistics will be collected :
  - S : the sum of customer response times for all customers who have departed by the current time
  - F : the total number of customers who spend 4 or more minutes at the checkout counter
  - N_D: the total number of departures up to the current simulation time.
- These three cumulative statistics will be updated whenever the departure event occurs.
- The simulation table for Example 3.4 is shown in Table 3.2.
- The response time for customer is computed by
- Response time = CLOCK TIME - attribute "time of arrival"
- For a simulation run length of 21 minutes
  - the average response time was $S/N_D = 15/4 = 3.75$ minutes
  - the observed proportion of customers who spent 4 or more minutes in the system was $F/N_D = 0.75$.

Table 3.2 Simulation Table for Example 3.4

	System State		*List*	*Future Event*	*Cumulative Statistics*		
Clock	$LQ(t)$	$LS(t)$	*"CHECKOUT LINE"*	*List*	S	N_D	F
0	0	1	$(C1, 0)$	$(D, 4, C1)$ $(A, 8, C2)$ $(E, 60)$	0	0	0
4	0	0		$(A, 8, C2)$ $(E, 60)$	4	1	1
8	0	1	$(C2, 8)$	$(D, 9, C2)$ $(A, 14, C3)$ $(E, 60)$	4	1	1
9	0	0		$(A, 14, C3)$ $(E, 60)$	5	2	1
14	0	1	$(C3, 14)$	$(A, 15, C4)$ $(D, 18, C3)$ $(E, 60)$	5	2	1
15	1	1	$(C3, 14)$ $(C4, 15)$	$(D, 18, C3)$ $(A, 23, C5)$ $(E, 60)$	5	2	1
18	0	1	$(C4, 15)$	$(D, 21, C4)$ $(A, 23, C5)$ $(E, 60)$	9	3	2
21	0	0		$(A, 23, C5)$ $(E, 60)$	15	4	3

Example 3.5 (The Dump Truck Problem, Figure 3.7)



- The distributions of loading time, weighing time, and travel time are given in Tables 3.3, 3.4, and 3.5, respectively, from Table A.1.
- The purpose of the simulation is to estimate the loader and scale utilizations (percentage of time busy).

Table 3.3 Distribution of Loading Time for the Dump Trucks

Loading Time	*Probability*	*Cumulative Probability*	*Random-Digit Assignment*
5	0.30	0.30	1–3
10	0.50	0.80	4–8
15	0.20	1.00	9–0

Table 3.4 Distribution of Weighing Time for the Dump Trucks

Weighing Time	*Probability*	*Cumulative Probability*	*Random-Digit Assignment*
12	0.70	0.70	1–7
16	0.30	1.00	8–0

Table 3.5 Distribution of Travel Time for the Dump Trucks

Travel Time	*Probability*	*Cumulative Probability*	*Random-Digit Assignment*
40	0.40	0.40	1–4
60	0.30	0.70	5–7
80	0.20	0.90	8–9
100	0.10	1.00	0

The activity times are taken from the following list as needed:

Loading Time	10	5	5	10	15	10	10
Weighing Time	12	12	12	16	12	16	
Travel Time	60	100	40	40	80		

- **System state [LQ(t), L(t), WQ(t), W(t)]**
  - LQ(t) = number of trucks in loader queue
  - L(t) = number of trucks (0, 1, or 2) being loaded
  - WQ(t) = number of trucks in weigh queue
  - W(t) = number of trucks (0 or 1) being weighed, all at simulation time t
- **Event notices :**
  - (ALQ, t, DT_i), dump truck i arrives at loader queue (ALQ) at time t
  - (EL, t, DT_i), dump truck i ends loading (EL) at time t
  - (EW, t, DT_i), dump truck i ends weighing (EW) at time t
- **Entities :** The six dump trucks (DT 1, … , DT 6)
- **Lists :**
  - Loader queue : all trucks waiting to begin loading, ordered on a first come, first served basis
  - Weigh queue : all trucks waiting to be weighed, ordered on a first come, first served basis
- **Activities :** Loading time, weighing time, and travel time
- **Delays :** Delay at loader queue, and delay at scale
- It has been assumed that five of the trucks are at the loaders and one is at the scale at time 0.
- The simulation table is given in Table 3.6.

Table 3.6 Simulation Table for Dump Truck Operation (Example 3.5)

Clock	System State				Lists			Cumulative Statistics	
					Loader	Weigh	Future Event		
t	$LQ(t)$	$L(t)$	$WQ(t)$	$W(t)$	Queue	Queue	List	B_L	B_S
0	3	2	0	1	$DT4$		$(EL, 5, DT3)$	0	0
					$DT5$		$(EL, 10, DT2)$		
					$DT6$		$(EW, 12, DT1)$		
5	2	2	1	1	$DT5$	$DT3$	$(EL, 10, DT2)$	10	5
					$DT6$		$(EL, 5+5, DT4)$		
							$(EW, 12, DT1)$		
10	1	2	2	1	$DT6$	$DT3$	$(EL, 10, DT4)$	20	10
						$DT2$	$(EW, 12, DT1)$		
							$(EL, 10+10, DT5)$		
10	0	2	3	1		$DT3$	$(EW, 12, DT1)$	20	10
						$DT2$	$(EL, 20, DT5)$		
						$DT4$	$(EL, 10+15, DT6)$		
12	0	2	2	1		$DT2$	$(EL, 20, DT5)$	24	12
						$DT4$	$(EW, 12+12, DT3)$		
							$(EL, 25, DT6)$		
							$(ALQ, 12+60, DT1)$		
20	0	1	3	1		$DT2$	$(EW, 24, DT3)$	40	20
						$DT4$	$(EL, 25, DT6)$		
						$DT5$	$(ALQ, 72, DT1)$		
24	0	1	2	1		$DT4$	$(EL, 25, DT6)$	44	24
						$DT5$	$(EW, 24+12, DT2)$		
							$(ALQ, 72, DT1)$		
							$(ALQ, 24+100, DT3)$		
25	0	0	3	1		$DT4$	(EW, 36, $DT2$)	45	25
						$DT5$	$(ALQ, 72, DT1)$		
						$DT6$	$(ALQ, 124, DT3)$		
36	0	0	2	1		$DT5$	$(EW, 36+16, DT4)$	45	36
						$DT6$	$(ALQ, 72, DT1)$		
							$(ALQ, 36+40, DT2)$		
							$(ALQ, 124, DT3)$		
52	0	0	1	1		$DT6$	$(EW, 52+12, DT5)$	45	52
							$(ALQ, 72, DT1)$		
							$(ALQ, 76, DT2)$		
							$(ALQ, 52+40, DT4)$		
							$(ALQ, 124, DT3)$		

Table 3.6 Continued

Clock	System State				Lists		Cumulative Statistics	
					Loader	Weigh	Future Event		
t	$LQ(t)$	$L(t)$	$WQ(t)$	$W(t)$	Queue	Queue	List	B_L	B_S
64	0	0	0	1			$(ALQ, 72, DT1)$ $(ALQ, 76, DT2)$ $(EW, 64+16, DT6)$ $(ALQ, 92, DT4)$ $(ALQ, 124, DT3)$ $(ALQ, 64+80, DT5)$	45	64
72	0	1	0	1			$(ALQ, 76, DT2)$ $(EW, 80, DT6)$ $(EL, 72+10, DT1)$ $(ALQ, 92, DT4)$ $(ALQ, 124, DT3)$ $(ALQ, 144, DT5)$	45	72
76	0	2	0	1			$(EW, 80, DT6)$ $(EL, 82, DT1)$ $(EL, 76+10, DT2)$ $(ALQ, 92, DT4)$ $(ALQ, 124, DT3)$ $(ALQ, 144, DT5)$	49	76

- This logic for the occurrence of the end-loading event
  - **When an end-loading (EL) event** occurs, say for truck j at time t , other events may be triggered.
  - If the scale is idle [W (t) =0], truck j begins weighing and an end-weighing event (EW) is scheduled on the FEL.
  - Otherwise, truck **j** joins the weigh queue.
  - If at this time there is another truck waiting for a loader, it will be removed from the loader queue and will begin loading by the scheduling of an end-loading event (EL) on the FEL.
- In order to estimate the loader and scale utilizations, two cumulative statistics are maintained:
  - B_L = total busy time of both loaders from time 0 to time t
  - B_S = total busy time of the scale from time 0 to time t
- The utilizations are estimated as follows:

Average loader utilization $= \dfrac{49/2}{76} = 0.32$

Average scale utilization $= \dfrac{76}{76} = 1.00$

- The events and activities were identified in Example 3.5.

Using the activity scanning approach

Activity	Condition
Loading time	Truck is at front of loader queue, and at least one loader is idle.
Weighing time	Truck is at front of weigh queue and weigh scale is idle.
Travel time	Truck has just completed weighing.

Using the process-interaction approach

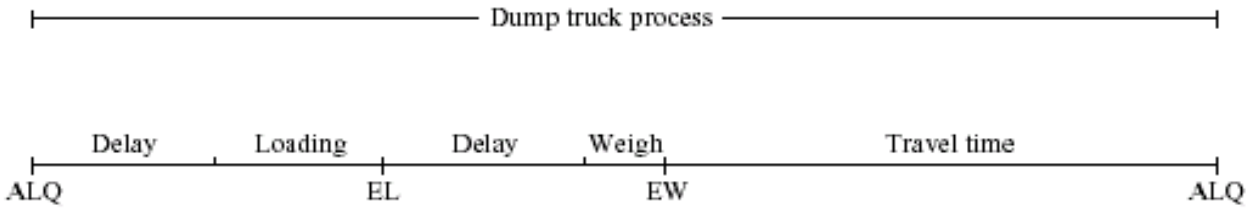


Figure 3.8 The dump truck process.

List Processing

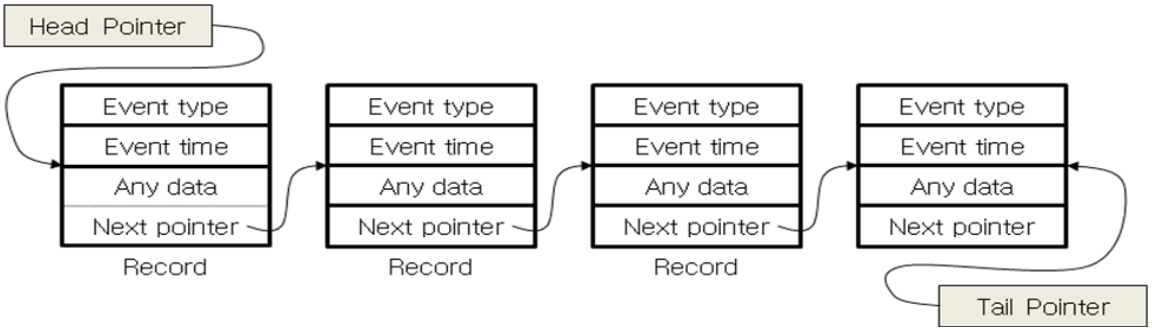
List processing deals with methods for handling lists of entities and the future event list

Basic properties and Operations

1. They have a head pointer/ top pointer
2. Some even have tail pointer

Operations

1. Removing a record from the top of the list
2. Removing a record from any location on the list
3. Adding an entity record to the top or bottom of the list
4. Adding a record to an arbitrary position on the list, determined by the ranking rule.



- List: a set of ordered or ranked records.
- Record: one entity or one event notice.
- Field: an entity identifier and its attributes
  : The event type, event time, and any other event related data

- **How to store record in a physical location in computer memory**
  - in arrays: successive records in contiguous locations by pointers to a **record: structures in C, classes in C++**

The main operations on a list :

1. **Removing a record from the top of the list.**
   - When time is advanced and the imminent event is due to be executed.
   - By adjusting the head pointer on the FEL ⇔ by removing the event at the top of the FEL.
2. **Removing a record from any location on the list.**
   - If an arbitrary event is being canceled, or an entity is removed from a list based on some of its attributes (say, for example, its priority and due date) to begin an activity.
   - By making a partial search through the list.
3. **Adding an entity record to the top or bottom of the list.**
   - When an entity joins the back of a first-in first-out queue.
   - by adjusting the tail pointer on the FEL ⇔ by adding an entity to the bottom of the FEL
4. **Adding a record to an arbitrary position on the list, determined by the ranking rule.**
   - If a queue has a ranking rule of earliest due date first (EDF).
   - By making a partial search through the list.

The goal of list-processing techniques: to make second and fourth operations efficient

- **The notation R(i) :** the i^{th} record in the array
- **Advantage:** Any specified record, say the i^{th}, can be retrieved quickly without searching, merely by referencing R (i).
- **Disadvantage:** When items are added to the middle of a list or the list must be rearranged.
  - Arrays typically have a fixed size, determined at compile time or upon initial allocation when a program first begins to execute.
  - In simulation, the maximum number of records for any list may be difficult or impossible to determine ahead of time, while the current number in a list may vary widely over the course of the simulation run.