The three-phase approach improves the execution efficiency of the activity-scanning method. In addition, proponents claim that the activity-scanning and three-phase approaches are particularly good at handling complex resource problems in which various combinations of resources are needed to accomplish different tasks. These approaches guarantee that all resources being freed at a given simulated time will be freed before any available resources are reallocated to new tasks.

Example 2: Call Center, Back Again

For the Call Center in Example 1, if we were to adopt the three-phase approach to the activity-scanning world view, the conditions for beginning each activity in Phase C would be as follows:

Activity	Condition
Service time by Able	A caller is in queue and Able is idle
Service time by Baker	A caller is in queue, Baker is idle and Able is busy

If it were to take the process-interaction world view, we would view the model from the viewpoint of a caller and its "life cycle." Considering a life cycle as beginning upon arrival, a customer process is pictured in Figure 4.

In summary, the process-interaction approach has been adopted by the simulation packages most popular in the USA. On the other hand, a number of activity-scanning packages are popular in the UK and Europe. Some of the packages allow portions of a model to be event-scheduling-based, if that orientation is convenient, mixed with the process-interaction approach. Finally, some of the packages are based on a flow chart, block diagram, or network structure, which upon closer examination turns out to be a specific implementation of the process-interaction concept.

1.3 Manual Simulation Using Event Scheduling

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

Example 3: Single-Channel Queue

Consider the grocery store with one checkout counter that is simulated in Example 5 from the Simulation Examples in a Spreadsheet chapter by an ad hoc method. The system consists of those customers in the waiting line plus the one (if any) checking out. A stopping time of 60 minutes is set for this example. 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.

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, defined in Table 9 from the Simulation Examples in a Spreadsheet chapter;
Service time, defined in Table 10 from the Simulation Examples in a Spreadsheet chapter.

Delay Customer time spent in waiting line.

The event notices are written as (event type, event time). In this model, the FEL will always contain either two or three event notices. The effect of the arrival and departure events is shown in detail in Figures 5 and 6.

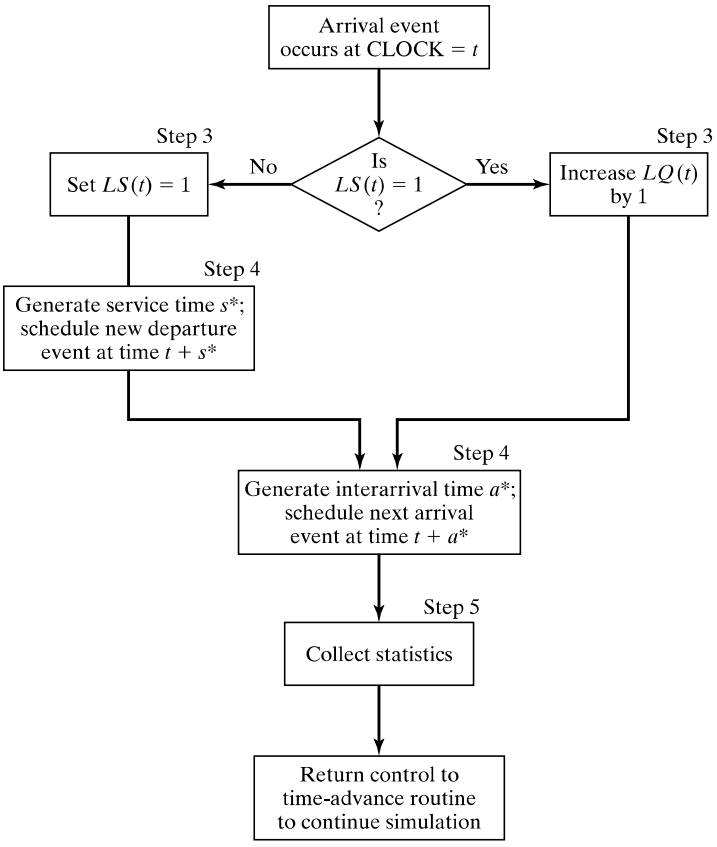

Figure 5 Execution of the arrival event.

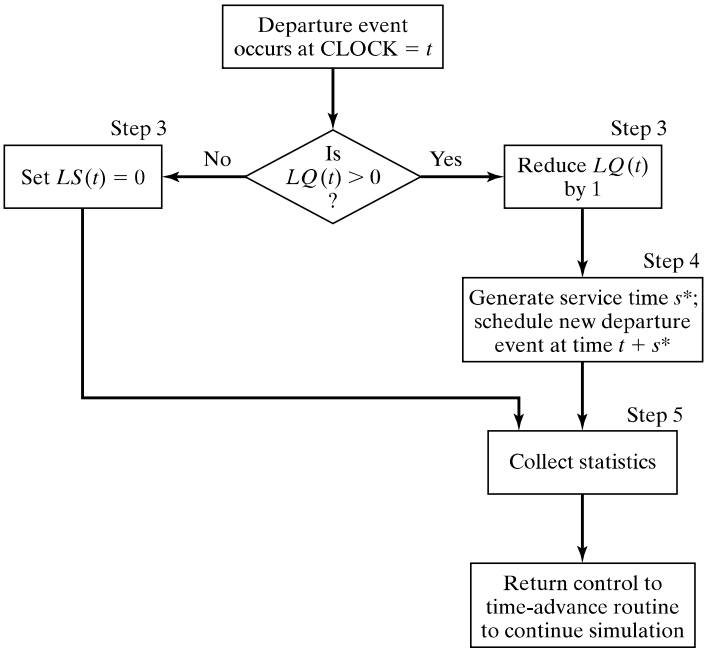

Figure 6 Execution of the departure event.

The simulation table for the checkout counter is given in Table 1. The reader should cover all system snapshots except one, starting with the first, and attempt to construct the next snapshot from the previous one and the event logic in Figures 5 and 6.

Interarrival Times	1	1	6	3	7	5	2	4	1 · · ·
Service Times	4	2	5	4	1	5	4	1	4 · · ·

Initial conditions are that the first customer arrive at time 0 and begin service. This is reflected in Table 1 by the system snapshot at time zero (CLOCK $= 0$), with $LQ(0) = 0, LS(0) = 1$, and both a departure event and arrival event on the FEL. Also, the simulation is scheduled to stop at time 60. Only two statistics, server utilization and maximum queue length, will be collected. Server utilization is defined by total server busy time (B) divided by total time (T_E). Total busy time, B, and maximum queue length, MQ, will be accumulated as the simulation progresses. A column headed "Comments" is included to aid the reader (a^* and s^* are the generated interarrival and service times, respectively).

As soon as the system snapshot at time CLOCK $= 0$ is complete, the simulation begins. At time 0, the imminent event is (A,1). The CLOCK is advanced to time 1, and (A,1) is removed from the FEL. Because $LS(t) = 1$ for $0 \leq t \leq 1$ (i.e., the server was busy for 1 minute), the cumulative busy time is increased from $B = 0$ to $B = 1$. By the event logic in Figure 6, set $LS(1) = 1$ (the

Table 1 Simulation Table for Checkout Counter

	System state				Cumulative statistics	
CLOCK	$LQ(t)$	$LS(t)$	Future event list	Comment	B	MQ
0	0	1	(A, 1) (D, 4) (E, 60)	First A occurs ($a^* = 1$) Schedule next A ($s^* = 4$) Schedule first D	0	0
1	1	1	(A, 2) (D, 4) (E, 60)	Second A occurs: (A, 1) ($a^* = 1$) Schedule next A (Customer delayed)	1	1
2	2	1	(D, 4) (A, 8) (E, 60)	Third A occurs: (A, 2) ($a^* = 6$) Schedule next A (Two customers delayed)	2	2
4	1	1	(D, 6) (A, 8) (E, 60)	First D occurs: (D, 4) ($s^* = 2$) Schedule next D (Customer delayed)	4	2
6	0	1	(A, 8) (D, 11) (E, 60)	Second D occurs: (D, 6) ($s^* = 5$) Schedule next D	6	2
8	1	1	(D, 11) (A, 11) (E, 60)	Fourth A occurs: (A, 8) ($a^* = 3$) Schedule next A (Customer delayed)	8	2
11	1	1	(D, 15) (A, 18) (E, 60)	Fifth A occurs: (A, 11) ($a^* = 7$) Schedule next A Third D occurs: (D, 11) ($s^* = 4$) Schedule next D (Customer delayed)	11	2
15	0	1	(D, 16) (A, 18) (E, 60)	Fourth D occurs: (D, 15) ($s^* = 1$) Schedule next D	15	2
16	0	0	(A, 18) (E, 60)	Fifth D occurs: (D, 16)	16	2
18	0	1	(D, 23) (A, 23) (E, 60)	Sixth A occurs ($a^* = 5$) Schedule next A ($s^* = 5$) Schedule next D	16	2
23	0	1	(A, 25) (D, 27) (E, 60)	Seventh A occurs: (A, 23) ($a^* = 2$) Schedule next Arrival Sixth D occurs: (D, 23)	21	2

server becomes busy). The FEL is left with three future events, (A, 2), (D, 4), and (E, 60). The simulation CLOCK is next advanced to time 2, and an arrival event is executed. The interpretation of the remainder of Table 1 is left to the reader.

The simulation in Table 1 covers the time interval [0,23]. At simulated time 23, the system empties, but the next arrival also occurs at time 23. The server was busy for 21 of the 23 time units simulated, and the maximum queue length was two. This simulation is, of course, too short to draw any reliable conclusions. Note that the simulation table gives the system state at all times, not just the listed times. For example, from time 11 to time 15, there is one customer in service and one in the waiting line.

When an event-scheduling algorithm is implemented in simulation software, the software maintains just one set of system states and other attribute values, that is, just one snapshot (the current one or partially updated one). With the idea of implementing event scheduling in Java or some other general-purpose language, the following rule should be followed. A new snapshot can be derived only from the previous snapshot, newly generated random variables, and the event logic (Figures 5 and 6). Past snapshots should be ignored for advancing of the clock. The current snapshot must contain all information necessary to continue the simulation.

Example 4: The Checkout-Counter Simulation, Continued

Suppose that, in the simulation of the checkout counter in Example 3, the simulation analyst desires to estimate mean response time and mean proportion of customers who spend 5 or more minutes in the system. A response time is the length of time a customer spends in the system. In order to estimate these customer averages, it is necessary to expand the model in Example 3 to represent the individual customers explicitly. In addition, to be able to compute an individual customer's response time when that customer departs, it will be necessary to know that customer's arrival time. Therefore, a customer entity with arrival time as an attribute will be added to the list of model components in Example 3. These customer entities will be stored in a list to be called "CHECKOUT LINE"; they will be called $C1, C2, C3, \ldots$ Finally, the event notices on the FEL will be expanded to indicate which customer is affected. For example, (D,4,$C1$) means that customer $C1$ will depart at time 4. The additional model components are the following:

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

"CHECKOUT LINE," the set of all customers currently at the checkout counter (being served or waiting to be served), ordered by time of arrival.

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 5 or more minutes at the checkout counter; and N_D, the total number of departures up to the current simulation time. These three cumulative statistics will be updated whenever the departure event