2

Simulation Examples

This chapter presents several examples of simulations that can be performed by devising a simulation table either manually or with a spreadsheet. The simulation table provides a systematic method for tracking system state over time. These examples provide insight into the methodology of discrete system simulation and the descriptive statistics used for predicting system performance.

The simulations in this chapter entail three steps:

1. Determine the characteristics of each of the inputs to the simulation. Quite often, these may be modeled as probability distributions, either continuous or discrete.
2. Construct a simulation table. Each simulation table is different, for each is developed for the problem at hand. An example of a simulation table is shown in Table 2.1. In this example there are p inputs, x_{ij}, $j = 1, 2, \ldots, p$, and one response, y_i, for each of repetitions $i = 1, 2, \ldots, n$. Initialize the table by filling in the data for repetition 1.
3. For each repetition i, generate a value for each of the p inputs, and evaluate the function, calculating a value of the response y_i. The input values may be computed by sampling values from the distributions determined in step 1. A response typically depends on the inputs and one or more previous responses.

This chapter gives a number of simulation examples in queueing, inventory, and reliability. The two queueing examples provide a single-server and

Table 2.1. Simulation Table

	Inputs						*Response*
Repetitions	x_{i1}	x_{i2}	$\cdots$	x_{ij}	$\cdots$	x_{ip}	y_i
1							
2							
3							
.							
.							
.							
n							

two-server system, respectively. (Chapter 6 provides more insight into queueing models.) The first inventory example involves a problem that has a closed-form solution; thus the simulation solution can be compared to the mathematical solution. The second inventory example pertains to the classic order-level model.

Finally, there is an example that introduces the concept of random normal numbers and a model for the determinination of lead-time demand.

2.1 Simulation of Queueing Systems

A queueing system is described by its calling population, the nature of the arrivals, the service mechanism, the system capacity, and the queueing discipline. These attributes of a queueing system are described in detail in Chapter 6. A simple single-channel queueing system is portrayed in Figure 2.1.

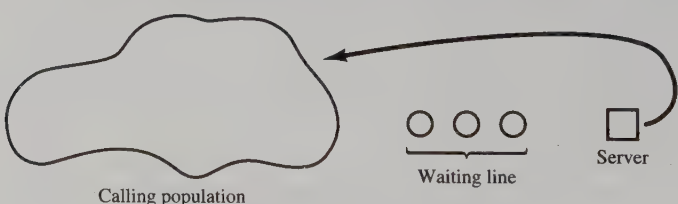

Figure 2.1. Queueing system.

In the single-channel queue, the calling population is infinite; that is, if a unit leaves the calling population and joins the waiting line or enters service, there is no change in the arrival rate of other units that may need service. Arrivals for service occur one at a time in a random fashion; once they join the waiting line, they are eventually served. In addition, service times are of some random length according to a probability distribution which does not change over time. The system capacity has no limit, meaning that any number of units can wait in line. Finally, units are served in the order of their arrival (often called FIFO: first in, first out) by a single server or channel.

Arrivals and services are defined by the distribution of the time between arrivals and the distribution of service times, respectively. For any simple single- or multi-channel queue, the overall effective arrival rate must be less than the total service rate, or the waiting line will grow without bound. When queues grow without bound, they are termed "explosive" or unstable. (In some reentrant queueing networks in which units return a number of times to the same server before finally exiting the system, the condition about arrival rate being less than service rate may not guarantee stability. See Harrison and Nguyen [1995] for more explanation. Interestingly, this type of instability was noticed first, not in theory, but in actual manufacturing in semiconductor plants.) More complex situations may occur—for example, arrival rates that are greater than service rates for short periods of time, or networks of queues with routing. However, this chapter sticks to the simplest, more basic queues.

Prior to introducing several simulations of queueing systems, it is necessary to understand the concepts of system state, events, and simulation clock. (These concepts are studied systematically in Chapter 3.) The *state* of the system is the number of units in the system and the status of the server, busy or idle. An *event* is a set of circumstances that cause an instantaneous change in the state of the system. In a single-channel queueing system there are only two possible events that can affect the state of the system. They are the entry of a unit into the system (the arrival event) or the completion of service on a unit (the departure event). The queueing system includes the server, the unit being serviced (if one is being serviced), and units in the queue (if any are waiting). The *simulation clock* is used to track simulated time.

If a unit has just completed service, the simulation proceeds in the manner shown in the flow diagram of Figure 2.2. Note that the server has only two possible states: it is either busy or idle.

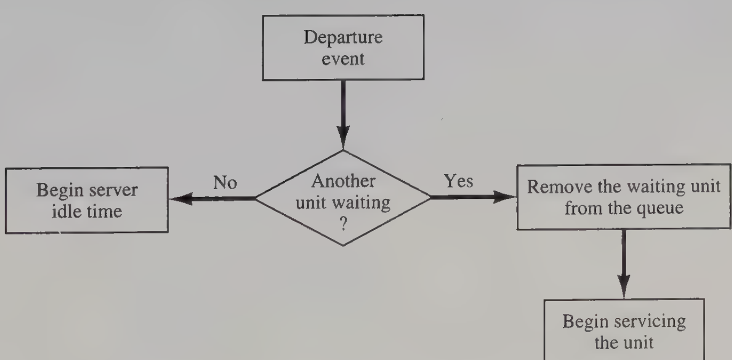

Figure 2.2. Service-just-completed flow diagram.

The arrival event occurs when a unit enters the system. The flow diagram for the arrival event is shown in Figure 2.3. The unit may find the server either idle or busy; therefore, either the unit begins service immediately, or it enters the queue for the server. The unit follows the course of action shown in Figure 2.4.

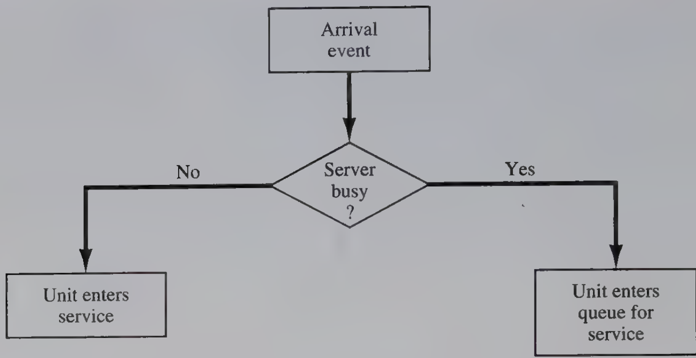

Figure 2.3. Unit-entering-system flow diagram.

If the server is busy, the unit enters the queue. If the server is idle and the queue is empty, the unit begins service. It is not possible for the server to be idle and the queue to be nonempty.

		Queue status	
		Not empty	Empty
Server status	Busy	Enter queue	Enter queue
	Idle	Impossible	Enter service

Figure 2.4. Potential unit actions upon arrival.

After the completion of a service the server may become idle or remain busy with the next unit. The relationship of these two outcomes to the status of the queue is shown in Figure 2.5. If the queue is not empty, another unit will enter the server and it will be busy. If the queue is empty, the server will be idle after a service is completed. These two possibilities are shown as the shaded portions of Figure 2.5. It is impossible for the server to become busy if the queue is empty when a service is completed. Similarly, it is impossible for the server to be idle after a service is completed when the queue is not empty.

		Queue status	
		Not empty	Empty
Server outcomes	Busy		Impossible
	Idle	Impossible	

Figure 2.5. Server outcomes after service completion.

Now, how can the events described above occur in simulated time? Simulations of queueing systems generally require the maintenance of an event list for determining what happens next. The event list tracks the future times at which the different types of events occur. Simulations using event lists are described in Chapter 3. This chapter simplifies the simulation by tracking each unit explicitly. Simulation clock times for arrivals and departures are computed in a simulation table customized for each problem. In simulation, events usually occur at random times, the randomness imitating uncertainty in real life. For example, it is not known with certainty when the next customer will arrive at a grocery checkout counter, or how long the bank teller will take to complete a transaction. In these cases, a statistical model of the data is developed from either data collected and analyzed, or subjective estimates and assumptions.

The randomness needed to imitate real life is made possible through the use of "random numbers." Random numbers are distributed uniformly and independently on the interval $(0, 1)$. Random digits are uniformly distributed on the set $\{0, 1, 2, \ldots, 9\}$. Random digits can be used to form random numbers by selecting the proper number of digits for each random number and placing a decimal point to the left of the value selected. The proper number of digits is dictated by the accuracy of the data being used for input purposes. If the input distribution has values with two decimal places, two digits are taken from a random-digits table (such as Table A.1) and the decimal point is placed to the left to form a random number.

Random numbers can also be generated in simulation packages and in spreadsheets such as Excel®. For example, Excel has a macro function called RAND() that returns a "random" number between 0 and 1. When numbers are generated using a procedure, they are often referred to as pseudo-random numbers. Since the method is known, it is always possible to know the sequence of numbers that will be generated prior to the simulation. The most commonly used methods for generating random numbers are discussed in Chapter 7.

In a single-channel queueing system, interarrival times and service times are generated from the distributions of these random variables. The examples that follow show how such times are generated. For simplicity, assume that the times between arrivals were generated by rolling a die five times and recording the up face. Table 2.2 contains a set of five interarrival times generated in this manner. These five interarrival times are used to compute the arrival times of six customers at the queueing system.

Table 2.2. Interarrival and Clock Times

Customer	*Interarrival Time*	*Arrival Time on Clock*
1	—	0
2	2	2
3	4	6
4	1	7
5	2	9
6	6	15

Table 2.3. Service Times

Customer	*Service Time*
1	2
2	1
3	3
4	2
5	1
6	4

The first customer is assumed to arrive at clock time 0. This starts the clock in operation. The second customer arrives two time units later, at a clock time of 2. The third customer arrives four time units later, at a clock time of 6; and so on.

The second time of interest is the service time. Table 2.3 contains service times generated at random from a distribution of service times. The only possible service times are one, two, three, and four time units. Assuming that all four values are equally likely to occur, these values could have been generated by placing the numbers one through four on chips and drawing the chips from a hat with replacement, being sure to record the numbers selected. Now, the interarrival times and service times must be meshed to simulate the single-channel queueing system. As shown in Table 2.4, the first customer arrives at clock time 0 and immediately begins service, which requires two minutes. Service is completed at clock time 2. The second customer arrives at clock time 2 and is finished at clock time 3. Note that the fourth customer arrived at clock time 7, but service could not begin until clock time 9. This occurred because customer 3 did not finish service until clock time 9.

Table 2.4 was designed specifically for a single-channel queue which serves customers on a first-in, first-out (FIFO) basis. It keeps track of the clock time

Table 2.4. Simulation Table Emphasizing Clock Times

A	B	C	D	E
Customer Number	*Arrival Time (Clock)*	*Time Service Begins (Clock)*	*Service Time (Duration)*	*Time Service Ends (Clock)*
1	0	0	2	2
2	2	2	1	3
3	6	6	3	9
4	7	9	2	11
5	9	11	1	12
6	15	15	4	19

Table 2.5. Chronological Ordering of Events

Event Type	*Customer Number*	*Clock Time*
Arrival	1	0
Departure	1	2
Arrival	2	2
Departure	2	3
Arrival	3	6
Arrival	4	7
Departure	3	9
Arrival	5	9
Departure	4	11
Departure	5	12
Arrival	6	15
Departure	6	19

at which each event occurs. The second column of Table 2.4 records the clock time of each arrival event, while the last column records the clock time of each departure event. The occurrence of the two types of events in chronological order is shown in Table 2.5 and Figure 2.6.

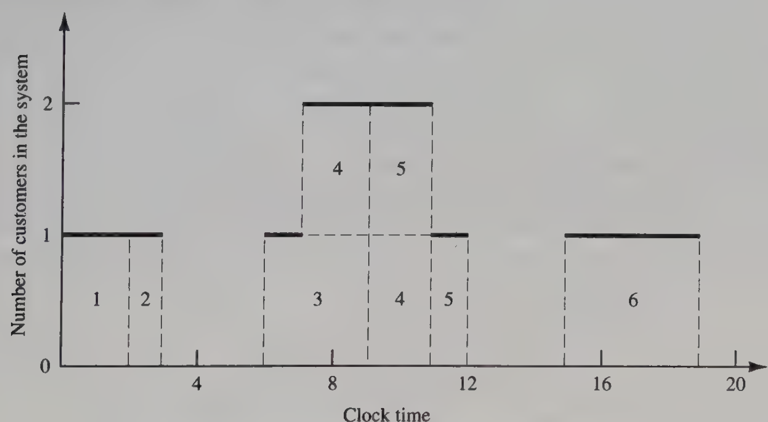

Figure 2.6. Number of customers in the system.

It should be noted that Table 2.5 is ordered by clock time, in which case the events may or may not be ordered by customer number. The chronological ordering of events is the basis of the approach to discrete-event simulation described in Chapter 3.

Figure 2.6 depicts the number of customers in the system at the various clock times. It is a visual image of the event listing of Table 2.5. Customer 1

Table 2.6. Distribution of Time Between Arrivals

Time between Arrivals (Minutes)	*Probability*	*Cumulative Probability*	*Random-Digit Assignment*
1	0.125	0.125	001–125
2	0.125	0.250	126–250
3	0.125	0.375	251–375
4	0.125	0.500	376–500
5	0.125	0.625	501–625
6	0.125	0.750	626–750
7	0.125	0.875	751–875
8	0.125	1.000	876–000

is in the system from clock time 0 to clock time 2. Customer 2 arrives at clock time 2 and departs at clock time 3. No customers are in the system from clock time 3 to clock time 6. During some time periods two customers are in the system, such as at clock time 8, when both customers 3 and 4 are in the system. Also, there are times when events occur simultaneously, such as at clock time 9, when customer 5 arrives and customer 3 departs.

Example 2.1 follows the logic described above while keeping track of a number of attributes of the system. Example 2.2 is concerned with a two-channel queueing system. The flow diagrams for a multichannel queueing system are slightly different from those for a single-channel system. The development and interpretation of these flow diagrams is left as an exercise for the reader.

Example 2.1 Single-Channel Queue

A small grocery store has only one checkout counter. Customers arrive at this checkout counter at random from 1 to 8 minutes apart. Each possible value of interarrival time has the same probability of occurrence, as shown in Table 2.6. The service times vary from 1 to 6 minutes with the probabilities shown in Table 2.7. The problem is to analyze the system by simulating the arrival and service of 20 customers.

Table 2.7. Service-Time Distribution

Service Time (Minutes)	*Probability*	*Cumulative Probability*	*Random-Digit Assignment*
1	0.10	0.10	01–10
2	0.20	0.30	11–30
3	0.30	0.60	31–60
4	0.25	0.85	61–85
5	0.10	0.95	86–95
6	0.05	1.00	96–00

In actuality, 20 customers is too small a sample size to allow drawing any reliable conclusions. The accuracy of the results is enhanced by increasing the sample size, as discussed in Chapter 11. However, the purpose of the exercise is to demonstrate how simple simulations can be carried out in a table, either manually or with a spreadsheet, not to recommend changes in the grocery store. A second issue, discussed thoroughly in Chapter 11, is that of initial conditions. A simulation of a grocery store that starts with an empty system is not realistic unless the intention is to model the system from startup or to model until steady-state operation is reached. Here, to keep things simple, starting conditions and concerns are overlooked.

A set of uniformly distributed random numbers is needed to generate the arrivals at the checkout counter. Random numbers have the following properties:

1. The set of random numbers is uniformly distributed between 0 and 1.
2. Successive random numbers are independent.

With tabular simulations, random digits such as those found in Table A.1 in the Appendix can be converted to random numbers. If using a spreadsheet, most have a built-in random-number generator such as RAND() in Excel. The example in the text uses random digits from Table A.1; in some of the exercises the student is asked to use a spreadsheet.

Random digits are converted to random numbers by placing a decimal point appropriately. Since the probabilities in Table 2.6 are accurate to 3 significant digits, three-place random numbers will suffice. It is necessary to list only 19 random numbers to generate times between arrivals. Why only 19 numbers? The first arrival is assumed to occur at time 0, so only 19 more arrivals need to be generated to end up with 20 customers. Similarly, for Table 2.7, two-place random numbers will suffice.

The rightmost two columns of Tables 2.6 and 2.7 are used to generate random arrivals and random service times. The third column in each table contains the cumulative probability for the distribution. The rightmost column contains the random digit-assignment. In Table 2.6, the first random-digit assignment is 001–125. There are 1000 three-digit values possible (001 through 000). The probability of a time-between-arrivals of 1 minute is 0.125, and 125 of the 1000 random-digit values are assigned to such an occurrence. Times between arrivals for 19 customers are generated by listing 19 three-digit values from Table A.1 and comparing them to the random-digit assignment of Table 2.6.

For manual simulations, it is good practice to start at a random position in the random-digit table and proceed in a systematic direction, never re-using the same stream of digits in a given problem. If the same pattern is used repeatedly, bias could result, because the same event pattern would be generated. In Excel, each time the random function RAND() is evaluated, it returns a new random value.

The time-between-arrival determination is shown in Table 2.8. Note that the first random digits are 913. To obtain the corresponding time between

Table 2.8. Time-Between-Arrivals Determination

Customer	Random Digits	Time between Arrivals (Minutes)	Customer	Random Digits	Time between Arrivals (Minutes)
1	–	–	11	109	1
2	913	8	12	093	1
3	727	6	13	607	5
4	015	1	14	738	6
5	948	8	15	359	3
6	309	3	16	888	8
7	922	8	17	106	1
8	753	7	18	212	2
9	235	2	19	493	4
10	302	3	20	535	5

arrivals, enter the fourth column of Table 2.6 and read 8 minutes from the first column of the table. Alternatively, we see that 0.913 is between the cumulative probabilities 0.876 and 1.000, again resulting in 8 minutes as the generated time.

Service times for all 20 customers are shown in Table 2.9. These service times were generated based on the methodology described above, together with the aid of Table 2.7. The first customer's service time is 4 minutes because the random digits 84 fall in the bracket 61–85, or alternatively because the derived random number 0.84 falls between the cumulative probabilities 0.61 and 0.85.

Table 2.9. Service Times Generated

Customer	Random Digits	Service Time (Minutes)	Customer	Random Digits	Service Time (Minutes)
1	84	4	11	32	3
2	10	1	12	94	5
3	74	4	13	79	4
4	53	3	14	05	1
5	17	2	15	79	5
6	79	4	16	84	4
7	91	5	17	52	3
8	67	4	18	55	3
9	89	5	19	30	2
10	38	3	20	50	3

The essence of a manual simulation is the simulation table. These tables are designed for the problem at hand, with columns added to answer the questions posed. The simulation table for the single-channel queue, shown in Table 2.10, is an extension of the type of table already seen in Table 2.4. The first step is to initialize the table by filling in cells for the first customer. The first customer is assumed to arrive at time 0. Service begins immediately and finishes at time 4. The customer was in the system for 4 minutes. After the first customer, subsequent rows in the table are based on the random numbers for interarrival time and service time and the completion time of the previous customer. For example, the second customer arrives at time 8. Thus, the server (checkout person) was idle for 4 minutes. Skipping down to the fourth customer, it is seen that this customer arrived at time 15 but could not be served until time 18. This customer had to wait in the queue for 3 minutes. This process continues for all 20 customers. Extra columns have been added to collect statistical measures of performance such as each customer's time in the system and the server's idle time (if any) since the previous customer departed. In order to compute summary statistics, totals are formed as shown for service times, time customers spend in the system, idle time of the server, and time the customers wait in the queue.

In the exercises, the reader is asked to implement the simulation table for the single-channel queue, Table 2.10, in Excel or another spreadsheet. Here we give some hints when using Excel. The key column to compute is column E, the "Time Service Begins." (We leave for the reader the question of how to compute the random interarrival and service times, but suggest the RAND() random-number generator or other built-in distribution in Excel.) First, the reader may fill in row 1 for the first customer manually. The values for the remaining customers must use macro formulas (which begin with an equals sign in Excel). Note that a customer begins service at the later of its own arrival time (column C) or the completion time (column G) of the previous customer. Therefore, for customer 10, service begins at E10 = MAX(C10, G9), where MAX() is the Excel macro function that returns the maximum value in a range or list of cells. This easily generalizes to other customers. (The statistical measures in columns H and I are easily computed by simple subtractions–also left for the reader.) A final hint on how to verify your spreadsheet model: instead of using a random function for arrivals and service times, type in the actual values given in Table 2.10 in columns B and D. If your formulas are correct, the spreadsheet should duplicate Table 2.10 exactly. After verification, replace the numbers by an appropriate random function. Then on each recalculation of the spreadsheet (function key F9 in Excel), it will generate new random numbers and you will get a new "run" of the simulation.

Some of the findings from the simulation in Table 2.10 are as follows:

Table 2.10. Simulation Table for Queueing Problem

A *Customer*	B *Time Since Last Arrival (Minutes)*	C *Arrival Time*	D *Service Time (Minutes)*	E *Time Service Begins*	F *Time Customer Waits in Queue (Minutes)*	G *Time Service Ends*	H *Time Customer Spends in System (Minutes)*	I *Idle Time of Server (Minutes)*
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
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
			68		56		124	18

1. The average waiting time for a customer is 2.8 minutes. This is determined in the following manner:

$$\text{average waiting time (minutes)} = \frac{\text{total time customers wait in queue (minutes)}}{\text{total numbers of customers}}$$

$$= \frac{56}{20} = 2.8 \text{ minutes}$$

2. The probability that a customer has to wait in the queue is 0.65. This is determined in the following manner:

$$\text{probability (wait)} = \frac{\text{number of customers who wait}}{\text{total number of customers}}$$

$$= \frac{13}{20} = 0.65$$

3. The fraction of idle time of the server is 0.21. This is determined in the following manner:

$$\text{probability of idle server} = \frac{\text{total idle time of server (minutes)}}{\text{total run time of simulation (minutes)}}$$

$$= \frac{18}{86} = 0.21$$

The probability of the server being busy is the complement of 0.21, or 0.79.

4. The average service time is 3.4 minutes, determined as follows:

$$\text{average service time (minutes)} = \frac{\text{total service time (minutes)}}{\text{total number of customers}}$$

$$= \frac{68}{20} = 3.4 \text{ minutes}$$

This result can be compared with the expected service time by finding the mean of the service-time distribution using the equation

$$E(S) = \sum_{s=0}^{\infty} sp(s)$$

Applying the expected-value equation to the distribution in Table 2.7 gives an expected service time of:

$$= 1(0.10) + 2(0.20) + 3(0.30) + 4(0.25) + 5(0.10) + 6(0.05)$$

$$= 3.2 \text{ minutes}$$

The expected service time is slightly lower than the average service time in the simulation. The longer the simulation, the closer the average will be to $E(S)$.

5. The average time between arrivals is 4.3 minutes. This is determined in the following manner:

$$\begin{array}{c}\text{average time between}\\\text{arrivals (minutes)}\end{array} = \frac{\begin{array}{c}\text{sum of all times}\\\text{between arrivals (minutes)}\end{array}}{\text{number of arrivals} - 1}$$

$$= \frac{82}{19} = 4.3 \text{ minutes}$$

One is subtracted from the denominator because the first arrival is assumed to occur at time 0. This result can be compared to the expected time between arrivals by finding the mean of the discrete uniform distribution whose endpoints are $a = 1$ and $b = 8$. The mean is given by

$$E(A) = \frac{a + b}{2} = \frac{1 + 8}{2} = 4.5 \text{ minutes}$$

The expected time between arrivals is slightly higher than the average. However, as the simulation becomes longer, the average value of the time between arrivals will approach the theoretical mean, $E(A)$.

6. The average waiting time of those who wait is 4.3 minutes. This is determined in the following manner:

$$\begin{array}{c}\text{Average waiting time of}\\\text{those who wait (minutes)}\end{array} = \frac{\text{total time customers wait in queue (minutes)}}{\text{total number of customers who wait}}$$

$$= \frac{56}{13} = 4.3 \text{ minutes}$$

7. The average time a customer spends in the system is 6.2 minutes. This can be determined in two ways. First, the computation can be achieved by the following relationship:

$$\begin{array}{c}\text{average time customer}\\\text{spends in the system}\\\text{(minutes)}\end{array} = \frac{\begin{array}{c}\text{total time customers spend in the}\\\text{system (minutes)}\end{array}}{\text{total number of customers}}$$

$$= \frac{124}{20} = 6.2 \text{ minutes}$$

The second way of computing this same result is to realize that the following relationship must hold:

$$\begin{array}{c}\text{average time}\\\text{customer spends}\\\text{in the system}\\\text{(minutes)}\end{array} = \begin{array}{c}\text{average time}\\\text{customer spends}\\\text{waiting in the}\\\text{queue (minutes)}\end{array} + \begin{array}{c}\text{average time}\\\text{customer spends}\\\text{in service}\\\text{(minutes)}\end{array}$$

From findings 1 and 4 this results in:

average time customer spends in the system (minutes)

$$= 2.8 + 3.4 = 6.2 \text{ minutes}$$

A decision maker would be interested in results of this type, but a longer simulation would increase the accuracy of the findings. However, some subjective inferences can be drawn at this point. Most customers have to wait; however, the average waiting time is not excessive. The server does not have an undue amount of idle time. Objective statements about the results would depend on balancing the cost of waiting with the cost of additional servers. (Simulations requiring variations of the arrival and service distribution, as well as implementation in a spreadsheet, are presented as exercises for the reader.) ◀

Example 2.2 The Able Baker Carhop Problem

This example illustrates the simulation procedure when there is more than one service channel. Consider a drive-in restaurant where carhops take orders and bring food to the car. Cars arrive in the manner shown in Table 2.11. There are two carhops—Able and Baker. Able is better able to do the job and works a bit faster than Baker. The distribution of their service times is shown in Tables 2.12 and 2.13.

Table 2.11. Interarrival Distribution of Cars

Time between Arrivals (Minutes)	*Probability*	*Cumulative Probability*	*Random-Digit Assignment*
1	0.25	0.25	01–25
2	0.40	0.65	26–65
3	0.20	0.85	66–85
4	0.15	1.00	86–00

The simulation proceeds in a manner similar to Example 2.1, except that it is more complex because of the two servers. A simplifying rule is that Able gets the customer if both carhops are idle. Perhaps, Able has seniority. (The

Table 2.12. Service Distribution of Able

Service Time (Minutes)	*Probability*	*Cumulative Probability*	*Random-Digit Assignment*
2	0.30	0.30	01–30
3	0.28	0.58	31–58
4	0.25	0.83	59–83
5	0.17	1.00	84–00

Table 2.13. Service Distribution of Baker

Service Time (Minutes)	*Probability*	*Cumulative Probability*	*Random-Digit Assignment*
3	0.35	0.35	01–35
4	0.25	0.60	36–60
5	0.20	0.80	61–80
6	0.20	1.00	81–00

solution would be different if the decision were made at random or by any other rule.)

The problem is to find how well the current arrangement is working. To estimate the system measures of performance, a simulation of 1 hour of operation is made. A longer simulation would yield more reliable results, but for purposes of illustration a 1-hour period has been selected.

The simulation proceeds in a manner similar to Example 2.1. Here there are more events: a customer arrives, a customer begins service from Able, a customer completes service from Able, a customer begins service from Baker, and a customer completes service from Baker. The simulation table is shown in Table 2.14.

In later exercises, the reader is asked to implement the simulation table, Table 2.14, in a spreadsheet such as Excel. Here we provide a few hints (and rules!). The row for the first customer is filled in manually, with the random-number function RAND() or another random function replacing the random digits. After the first customer, the cells for the other customers must be based on logic and formulas. For example, the "Clock Time of Arrival" (column D) in the row for the second customer is computed as follows:

```
D2 = D1 + C2
```

using notation similar to that used by most spreadsheets. (C2 is the time between arrivals 1 and 2.) This formula is easily generalized for any customer.

The logic to compute who gets a given customer, and when that service begins, is more complex. Here we give a hint using the Excel macro function IF(), which returns one of two values depending on whether a condition is true or false. [The syntax is IF(condition, value_if_true, value_if_false).] The logic goes as follows when a customer arrives: If the customer finds Able idle, the customer begins service immediately with Able. If Able is not idle but Baker is, then the customer begins service immediately with Baker. If both are busy, the customer begins service with the first server to become free. The logic requires that we compute when Able and Baker will become free, for which we use the built-in Excel function for maximum over a range, MAX(). For example, for customer 10, Able will become free at MAX(H$1:H9), since service completion time is in column H and we need to look at customers 1–9. (Using H$1 instead of H1 works better with Excel when formulas are copied. The dollar sign indicates an absolute reference versus a relative reference to a cell.) The resulting formula to compute whether and when Able serves customer 10 is as follows:

```
F10 = IF(D10>MAX(H$1:H9),D10, IF(D10>MAX(K$1:K9),"",
   MIN(MAX(H$1:H9),MAX(K$1:K9))))
```

In this formula, note that if the first condition (Able idle when customer 10 arrives) is true, then the customer begins immediately at the arrival time in D10. Otherwise, a second IF() function is evaluated, which says if Baker is idle, put nothing ("") in the cell. Otherwise, the function returns the time that Able or Baker becomes idle, whichever is first [the minimum or MIN() of their respective completion times]. A similar formula applies to cell I10 for "Time Service Begins" for Baker. For service times for Able, you could use another IF() function to make the cell blank or have a value:

```
G10 = IF(F10 > 0,new_service_time, "")
H10 = IF(F10 > 0, F10+G10, "")
```

and similarly for Baker. With these hints, we leave the formula for `new_service_time` as well as the remainder of the solution to the reader.

The analysis of Table 2.14 results in the following:

1. Over the 62-minute period Able was busy 90% of the time.
2. Baker was busy only 69% of the time. The seniority rule keeps Baker less busy (and gives Able more tips).
3. Nine of the 26 arrivals (about 35%) had to wait. The average waiting time for all customers was only about 0.42 minute (25 seconds), which is very small.
4. Those nine who did have to wait only waited an average of 1.22 minutes, which is quite low.
5. In summary, this system seems well balanced. One server cannot handle all the diners, and three servers would probably be too many. Adding an additional server would surely reduce the waiting time to nearly zero. However, the cost of waiting would have to be quite high to justify an additional server. ◀

Table 2.14. Simulation Table for Carhop Example

A	B	C	D	E	F	G	H	I	J	K	L
						Able			*Baker*		
Customer No.	*Random Digits for Arrival*	*Time between Arrivals*	*Clock Time of Arrival*	*Random Digits for Service*	*Time Service Begins*	*Service Time*	*Time Service Ends*	*Time Service Begins*	*Service Time*	*Time Service Ends*	*Time in Queue*
1	–	–	0	95	0	5	5				0
2	26	2	2	21				2	3	5	0
3	98	4	6	51	6	3	9				0
4	90	4	10	92	10	5	15				0
5	26	2	12	89				12	6	18	0
6	42	2	14	38	15	3	18				1
7	74	3	17	13	18	2	20				1
8	80	3	20	61	20	4	24				0
9	68	3	23	50				23	4	27	0
10	22	1	24	49	24	3	27				0
11	48	2	26	39	27	3	30				1
12	34	2	28	53				28	4	32	0
13	45	2	30	88	30	5	35				0
14	24	1	31	01				32	3	35	1
15	34	2	33	81	35	4	39				2
16	63	2	35	53				35	4	39	0
17	38	2	37	81	39	4	43				2
18	80	3	40	64				40	5	45	0
19	42	2	42	01	43	2	45				1
20	56	2	44	67	45	4	49				1
21	89	4	48	01				48	3	51	0
22	18	1	49	47	49	3	52				0
23	51	2	51	75				51	5	56	0
24	71	3	54	57	54	3	57				0
25	16	1	55	87				56	6	62	1
26	92	4	59	47	59	3	62				0
						56			43		11

2.2 Simulation of Inventory Systems

An important class of simulation problems involves inventory systems. A simple inventory system is shown in Figure 2.7. This inventory system has a periodic review of length N, at which time the inventory level is checked. An order is made to bring the inventory up to the level M. At the end of the first review period, an order quantity, Q_1, is placed. In this inventory system the lead time (i.e., the length of time between the placement and receipt of an order) is zero. Since demands are not usually known with certainty, the order quantities are probabilistic. Demand is shown as being uniform over the time period in Figure 2.7. In actuality, demands are not usually uniform and do fluctuate over time. One possibility is that demands all occur at the beginning of the cycle. Another is that the lead time is random of some positive length.

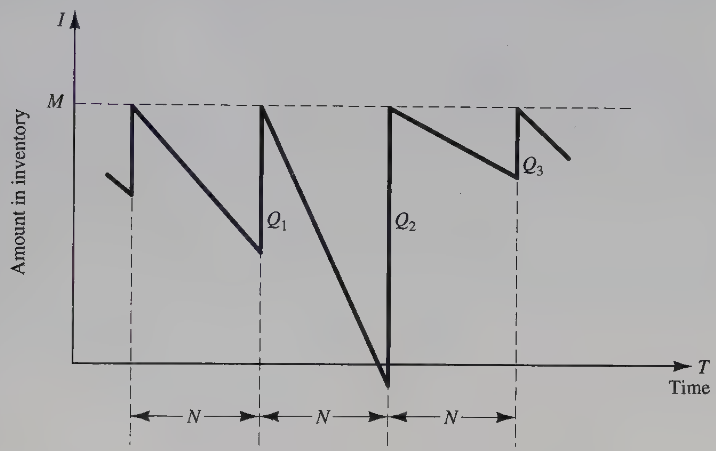

Figure 2.7. Probabilistic order-level inventory system.

Notice that in the second cycle, the amount in inventory drops below zero, indicating a shortage. In Figure 2.7, these units are backordered; when the order arrives, the demand for the backordered items is satisfied first. To avoid shortages, a buffer, or safety, stock would need to be carried.

Carrying stock in inventory has an associated cost attributed to the interest paid on the funds borrowed to buy the items (this also could be considered as the loss from not having the funds available for other investment purposes). Other costs can be placed in the carrying or holding cost column: renting of storage space, hiring guards, and so on. An alternative to carrying high inventory is to make more frequent reviews, and consequently, more frequent purchases or replenishments. This has an associated cost: the ordering cost. Also, there is a cost in being short. Customers may get angry, with a subsequent loss of good will. Larger inventories decrease the possibilities of shortages. These costs must be traded off in order to minimize the total cost of an inventory system.

The total cost (or total profit) of an inventory system is the measure of performance. This can be affected by the policy alternatives. For example, in Figure 2.7, the decision maker can control the maximum inventory level, M, and the length of the cycle, N. What effect does changing N have on the various costs?

In an (M, N) inventory system, the events that may occur are: the demand for items in the inventory, the review of the inventory position, and the receipt of an order at the end of each review period. When the lead time is zero, as in Figure 2.7, the last two events occur simultaneously.

In the following example for deciding how many newspapers to buy, only a single time period of specified length is relevant and only a single procurement is made. Inventory remaining at the end of the single time period is sold for scrap or discarded. A wide variety of real-world problems are of this form, including the stocking of spare parts, perishable items, style goods, and special seasonal items [Hadley and Whitin, 1963].

Example 2.3 The Newspaper Seller's Problem

A classical inventory problem concerns the purchase and sale of newspapers. The paper seller buys the papers for 33 cents each and sells them for 50 cents each. Newspapers not sold at the end of the day are sold as scrap for 5 cents each. Newspapers can be purchased in bundles of 10. Thus, the paper seller can buy 50, 60, and so on. There are three types of newsdays, "good," "fair," and "poor," with probabilities of 0.35, 0.45, and 0.20, respectively. The distribution of papers demanded on each of these days is given in Table 2.15. The problem is to determine the optimal number of papers the newspaper seller should purchase. This will be accomplished by simulating demands for 20 days and recording profits from sales each day.

The profits are given by the following relationship:

$$\text{Profit} = \left[\begin{pmatrix}\text{revenue}\\ \text{from sales}\end{pmatrix} - \begin{pmatrix}\text{cost of}\\ \text{newspapers}\end{pmatrix} - \begin{pmatrix}\text{lost profit from}\\ \text{excess demand}\end{pmatrix} + \begin{pmatrix}\text{salvage from sale}\\ \text{of scrap papers}\end{pmatrix}\right]$$

Table 2.15. Distribution of Newspapers Demanded

	Demand Probability Distribution		
Demand	*Good*	*Fair*	*Poor*
40	0.03	0.10	0.44
50	0.05	0.18	0.22
60	0.15	0.40	0.16
70	0.20	0.20	0.12
80	0.35	0.08	0.06
90	0.15	0.04	0.00
100	0.07	0.00	0.00

Table 2.16. Random-Digit Assignment for Type of Newsday

Type of Newsday	*Probability*	*Cumulative Probability*	*Random-Digit Assignment*
Good	0.35	0.35	01–35
Fair	0.45	0.80	36–80
Poor	0.20	1.00	81–00

From the problem statement, the revenue from sales is 50 cents for each paper sold. The cost of newspapers is 33 cents for each paper purchased. The lost profit from excess demand is 17 cents for each paper demanded that could not be provided. Such a shortage cost is somewhat controversial but makes the problem much more interesting. The salvage value of scrap papers is 5 cents each.

Tables 2.16 and 2.17 provide the random-digit assignments for the types of newsdays and the demands for those newsdays. To solve this problem by simulation requires setting a policy of buying a certain number of papers each day, then simulating the demands for papers over the 20-day time period to determine the total profit. The policy (number of newspapers purchased) is changed to other values and the simulation repeated until the best value is found.

Table 2.17. Random-Digit Assignments for Newspapers Demanded

	Cumulative Distribution			*Random-Digit Assignment*		
Demand	*Good*	*Fair*	*Poor*	*Good*	*Fair*	*Poor*
40	0.03	0.10	0.44	01–03	01–10	01–44
50	0.08	0.28	0.66	04–08	11–28	45–66
60	0.23	0.68	0.82	09–23	29–68	67–82
70	0.43	0.88	0.94	24–43	69–88	83–94
80	0.78	0.96	1.00	44–78	89–96	95–00
90	0.93	1.00	1.00	79–93	97–00	
100	1.00	1.00	1.00	94–00		

The simulation table for the decision to purchase 70 newspapers is shown in Table 2.18.

On day 1 the demand is for 60 newspapers. The revenue from the sale of 60 newspapers is \$30.00. Ten newspapers are left over at the end of the day. The salvage value at 5 cents each is 50 cents. The profit for the first day is determined as follows:

$$\text{Profit} = \$30.00 - \$23.10 - 0 + \$.50 = \$7.40$$

Table 2.18. Simulation Table for Purchase of 70 Newspapers

Day	*Random Digits for Type of Newsday*	*Type of Newsday*	*Random Digits for Demand*	*Demand*	*Revenue from Sales*	*Lost Profit from Excess Demand*	*Salvage from Sale of Scrap*	*Daily Profit*
1	94	Poor	80	60	$30.00	–	$0.50	$7.40
2	77	Fair	20	50	25.00	–	1.00	2.90
3	49	Fair	15	50	25.00	–	1.00	2.90
4	45	Fair	88	70	35.00	–	–	11.90
5	43	Fair	98	90	35.00	$3.40	–	8.50
6	32	Good	65	80	35.00	1.70	–	10.20
7	49	Fair	86	70	35.00	–	–	11.90
8	00	Poor	73	60	30.00	–	0.50	7.40
9	16	Good	24	70	35.00	–	–	11.90
10	24	Good	60	80	35.00	1.70	–	10.20
11	31	Good	60	80	35.00	1.70	–	10.20
12	14	Good	29	70	35.00	–	–	11.90
13	41	Fair	18	50	25.00	–	1.00	2.90
14	61	Fair	90	80	35.00	1.70	–	10.20
15	85	Poor	93	70	35.00	–	–	11.90
16	08	Good	73	80	35.00	1.70	–	10.20
17	15	Good	21	60	30.00	–	0.50	7.40
18	97	Poor	45	50	25.00	–	1.00	2.90
19	52	Fair	76	70	35.00	–	–	11.90
20	78	Fair	96	80	35.00	1.70	–	10.20
					$645.00	$13.60	$5.50	$174.90

On the fifth day the demand is greater than the supply. The revenue from sales is \$35.00, since only 70 papers are available under this policy. An additional 20 papers could have been sold. Thus, a lost profit of \$3.40 (20 × 17 cents) is assessed. The daily profit is determined as follows:

$$\text{Profit} = \$35.00 - \$23.10 - \$3.40 + 0 = \$8.50$$

The profit for the 20-day period is the sum of the daily profits, \$174.90. It can also be computed from the totals for the 20 days of the simulation as follows:

$$\text{Total profit} = \$645.00 - \$462.00 - \$13.60 + \$5.50 = \$174.90$$

In general, since the results of one day are independent of those of previous days, inventory problems of this type are easier than queueing problems when solved in a spreadsheet such as Excel. The determination of the optimal number of newspapers to purchase is left as an exercise for the reader. ◀

EXAMPLE 2.4 Simulation of an (M, N) Inventory System

This example follows the pattern of the probabilistic order-level inventory system shown in Figure 2.7. Suppose that the maximum inventory level, M, is 11 units and the review period, N, is 5 days. The problem is to estimate, by simulation, the average ending units in inventory and the number of days when a shortage condition occurs. The distribution of the number of units demanded per day is shown in Table 2.19. In this example, lead time is a random variable, as shown in Table 2.20. Assume that orders are placed at the close of business and are received for inventory at the beginning of business as determined by the lead time.

Table 2.19. Random-Digit Assignments for Daily Demand

Demand	*Probability*	*Cumulative Probability*	*Random-Digit Assignment*
0	0.10	0.10	01–10
1	0.25	0.35	11–35
2	0.35	0.70	36–70
3	0.21	0.91	71–91
4	0.09	1.00	92–00

Table 2.20. Random-Digit Assignments for Lead Time

Lead Time (Days)	*Probability*	*Cumulative Probability*	*Random-Digit Assignment*
1	0.6	0.6	1–6
2	0.3	0.9	7–9
3	0.1	1.0	0

Table 2.21. Simulation Tables for (M, N) Inventory System

Cycle	*Day*	*Beginning Inventory*	*Random Digits for Demand*	*Demand*	*Ending Inventory*	*Shortage Quantity*	*Order Quantity*	*Random Digits for Lead Time*	*Days until Order Arrives*
1	1	3	24	1	2	0	—	—	1
	2	2	35	1	1	0	—	—	0
	3	9	65	2	7	0	—	—	—
	4	7	81	3	4	0	—	—	—
	5	4	54	2	2	0	9	5	1
2	1	2	03	0	2	0	—	—	0
	2	11	87	3	8	0	—	—	—
	3	8	27	1	7	0	—	—	—
	4	7	73	3	4	0	—	—	—
	5	4	70	2	2	0	9	0	3
3	1	2	47	2	0	0	—	—	2
	2	0	45	2	0	2	—	—	1
	3	0	48	2	0	4	—	—	0
	4	9	17	1	4	0	—	—	—
	5	4	09	0	4	0	7	3	1
4	1	4	42	2	2	0	—	—	0
	2	9	87	3	6	0	—	—	—
	3	6	26	1	5	0	—	—	—
	4	5	36	2	3	0	—	—	—
	5	3	40	2	1	0	10	4	1
5	1	1	07	0	1	0	—	—	0
	2	11	63	2	9	0	—	—	—
	3	9	19	1	8	0	—	—	—
	4	8	88	3	5	0	—	—	—
	5	5	94	4	1	0	10	8	2
					88				

To make an estimate of the mean units in ending inventory, many cycles would have to be simulated. For purposes of this example, only five cycles will be shown. The reader is asked to continue the example as an exercise at the end of the chapter.

The random-digit assignments for daily demand and lead time are shown in the rightmost columns of Tables 2.19 and 2.20. The resulting simulation table is shown in Table 2.21. The simulation has been started with the inventory level at 3 units and an order of 8 units scheduled to arrive in 2 days' time.

Following the simulation table for several selected days indicates how the process operates. The order for 8 units is available on the morning of the third day of the first cycle, raising the inventory level from 1 unit to 9 units. Demands during the remainder of the first cycle reduced the ending inventory level to 2 units on the fifth day. Thus, an order for 9 units was placed. The lead time for this order was 1 day. The order of 9 units was added to inventory on the morning of day 2 of cycle 2.

Notice that the beginning inventory on the second day of the third cycle was zero. An order for 2 units on that day led to a shortage condition. The units were backordered on that day and the next day also. On the morning of day 4 of cycle 3 there was a beginning inventory of 9 units. The 4 units that were backordered and the 1 unit demanded that day reduced the ending inventory to 4 units.

Based on five cycles of simulation, the average ending inventory is approximately 3.5 (88 ÷ 25) units. On 2 of 25 days a shortage condition existed. ◀

2.3 Other Examples of Simulation

This section includes examples of the simulation of a reliability problem, a bombing mission, and the generation of the lead-time demand distribution given the distributions of demand and lead time.

Example 2.5 A Reliability Problem

A large milling machine has three different bearings that fail in service. The cumulative distribution function of the life of each bearing is identical, as shown in Table 2.22. When a bearing fails, the mill stops, a repairperson is called, and a new bearing is installed. The delay time of the repairperson's arriving at the milling machine is also a random variable, with the distribution given in Table 2.23. Downtime for the mill is estimated at \$5 per minute. The direct on-site cost of the repairperson is \$15 per hour. It takes 20 minutes to change one bearing, 30 minutes to change two bearings, and 40 minutes to change three bearings. The bearings cost \$16 each. A proposal has been made to replace all three bearings whenever a bearing fails. Management needs an evaluation of this proposal.

Table 2.24 represents a simulation of 20,000 hours of operation under the current method of operation. Note that there are instances where more than one bearing fails at the same time. This is unlikely to occur in practice and is

Table 2.22. Bearing-Life Distribution

Bearing Life (Hours)	*Probability*	*Cumulative Probability*	*Random-Digit Assignment*
1000	0.10	0.10	01–10
1100	0.13	0.23	11–23
1200	0.25	0.48	24–48
1300	0.13	0.61	49–61
1400	0.09	0.70	62–70
1500	0.12	0.82	71–82
1600	0.02	0.84	83–84
1700	0.06	0.90	85–90
1800	0.05	0.95	91–95
1900	0.05	1.00	96–00

due to using a rather coarse grid of 100 hours. It will be assumed in this example that the times are never exactly the same, and thus no more than one bearing is changed at any breakdown. Sixteen bearing changes were made for bearings 1 and 2, but only 14 bearing changes were required for bearing 3. The cost of the current system is estimated as follows:

Cost of bearings = 46 bearings × \$16/bearing = \$736

Cost of delay time = (110 + 125 + 95) minutes × \$5/minute = \$1650

Cost of downtime during repair =

46 bearings × 20 minutes/bearing × \$5/minute = \$4600

Cost of repairpersons =

46 bearings × 20 minutes/bearing × \$15/60 minutes= \$230

Total cost = \$736 + \$1650 + \$4600 + \$230 = \$7216

Table 2.23. Delay-Time Distribution

Delay Time (Minutes)	*Probability*	*Cumulative Probability*	*Random-Digit Assignment*
5	0.6	0.6	1–6
10	0.3	0.9	7–9
15	0.1	1.0	0

Table 2.25 is a simulation using the proposed method. Notice that bearing life is taken from Table 2.24, so that for as many bearings as were used in the current method, the bearing life is identical for both methods. It is assumed

Table 2.24. Bearing Replacement Using Current Method

	Bearing 1					Bearing 2					Bearing 3				
	RD[a]	Life (Hours)	Accumulated Life (Hours)	RD	Delay (Minutes)	RD	Life (Hours)	Accumulated Life (Hours)	RD	Delay (Minutes)	RD	Life (Hours)	Accumulated Life (Hours)	RD	Delay (Minutes)
1	67	1,400	1,400	2	5	70	1,500	1,500	0	15	76	1,500	1,500	0	15
2	08	1,000	2,400	3	5	43	1,200	2,700	7	10	65	1,400	2,900	2	5
3	49	1,300	3,700	1	5	86	1,700	4,400	3	5	61	1,400	4,300	7	10
4	84	1,600	5,300	7	10	93	1,800	6,200	1	5	96	1,900	6,200	1	5
5	44	1,200	6,500	8	10	81	1,600	7,800	2	5	65	1,400	7,600	3	5
6	30	1,200	7,700	1	5	44	1,200	9,000	8	10	56	1,300	8,900	3	5
7	10	1,000	8,700	2	5	19	1,100	10,100	1	5	11	1,100	10,000	6	5
8	63	1,400	10,100	8	10	51	1,300	11,400	1	5	86	1,700	11,700	3	5
9	02	1,000	11,100	3	5	45	1,300	12,700	7	10	57	1,300	13,000	1	5
10	02	1,000	12,100	8	10	12	1,100	13,800	8	5	49	1,300	14,300	4	5
11	77	1,500	13,600	7	10	48	1,300	15,100	0	15	36	1,200	15,500	8	10
12	59	1,300	14,900	5	5	09	1,000	16,100	8	10	44	1,200	16,700	2	5
13	23	1,100	16,000	5	5	44	1,200	17,300	1	5	94	1,800	18,500	1	5
14	53	1,300	17,300	9	10	46	1,200	18,500	2	5	78	1,500	20,000	7	10
15	85	1,700	19,000	6	5	40	1,200	19,700	8	10					
16	75	1,500	20,500	4	5	52	1,300	21,000	5	5					
					110					125					95

[a] RD, random digits.

Table 2.25. Bearing Replacement Using Proposed Method

	Bearing 1 Life (Hours)	*Bearing 2 Life (Hours)*	*Bearing 3 Life (Hours)*	*First Failure (Hours)*	*Accumulated Life (Hours)*	*RD*	*Delay (Minutes)*
1	1,400	1,500	1,500	1,400	1,400	3	5
2	1,000	1,200	1,400	1,000	2,400	7	10
3	1,300	1,700	1,400	1,300	3,700	5	5
4	1,600	1,800	1,900	1,600	5,300	1	5
5	1,200	1,600	1,400	1,200	6,500	4	5
6	1,200	1,200	1,300	1,200	7,700	3	5
7	1,000	1,100	1,100	1,000	8,700	7	10
8	1,400	1,300	1,700	1,300	10,000	8	10
9	1,000	1,300	1,300	1,000	11,000	8	10
10	1,000	1,100	1,300	1,000	12,000	3	5
11	1,500	1,300	1,200	1,200	13,200	2	5
12	1,300	1,000	1,200	1,000	14,200	4	5
13	1,100	1,200	1,800	1,100	15,300	1	5
14	1,300	1,200	1,500	1,200	16,500	6	5
15	1,700	1,200	63/1,400	1,200	17,700	2	5
16	1,500	1,300	21/1,100	1,100	18,800	7	10
17	85/1,700	53/1,300	23/1,100	1,100	19,900	0	15
18	05/1,000	29/1,200	51/1,300	1,000	20,900	5	5
							125

that the bearings are in order on a shelf and they are taken sequentially and placed on the mill. Since the proposed method uses more bearings than the current method, the second simulation uses new random digits for generating the additional lifetimes. (When comparing two scenarios, the effect of using different random numbers versus common random numbers is discussed in Chapter 12.) The random digits that lead to the lives of the additional bearings are shown above the slashed line beginning with the 15th replacement of bearing 3. When the new policy is used, some 18 sets of bearings were required. In the two simulations, repairperson delays were not duplicated but were generated independently using different random digits. The total cost of the new policy is computed as follows:

Cost of bearings = 54 bearings × \$16/bearing = \$864

Cost of delay time = 125 minutes × \$5/minute = \$625

Cost of downtime during repairs =

18 sets × 40 minutes/set × \$5/minute = \$3600

Cost of repairpersons =

18 sets × 40 minutes/set × \$15/60 minutes = \$180

Total cost = \$864 + \$625 + \$3600 + \$180 = \$5269

The new policy generates a savings of \$1947 over a 20,000-hour simulation. If the machine runs continuously, the simulated time is about $2\frac{1}{4}$ years. Thus, the savings are about \$865 per year. ◀

EXAMPLE 2.6 Random Normal Numbers

A classic simulation problem is that of a squadron of bombers attempting to destroy an ammunition depot shaped as shown in Figure 2.8. If a bomb lands anywhere on the depot, a hit is scored. Otherwise, the bomb is a miss. The aircraft fly in the horizontal direction. Ten bombers are in each squadron. The aiming point is the dot located in the heart of the ammunition dump. The point of impact is assumed to be normally distributed around the aiming point with a standard deviation of 600 meters in the horizontal direction and 300 meters in the vertical direction. The problem is to simulate the operation and make statements about the number of bombs on target.

Recall that the standardized normal variate, Z, with mean 0 and standard deviation 1, is distributed as

$$Z = \frac{X - \mu}{\sigma}$$

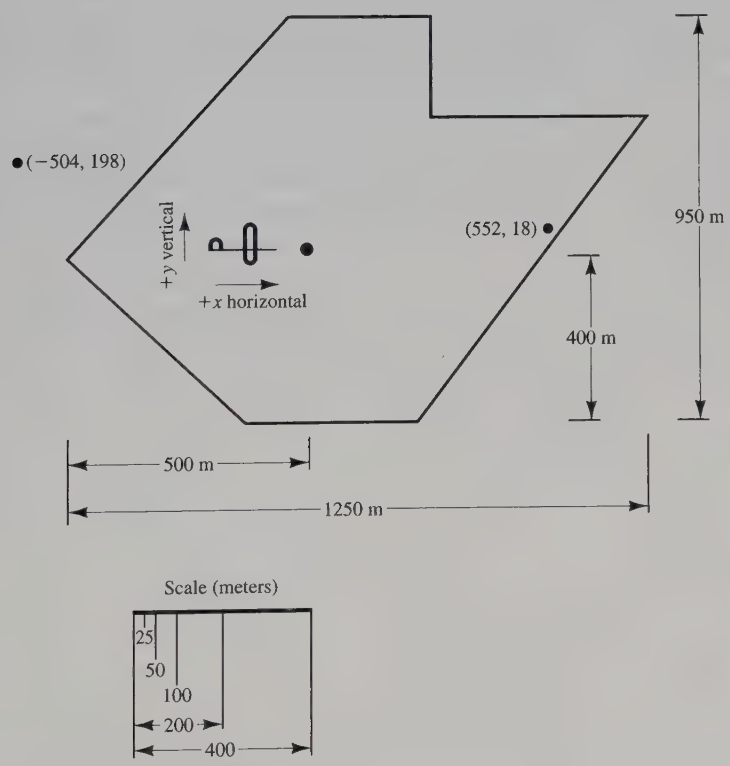

Figure 2.8. Ammunition depot.