Discrete-Event System Simulation

Fourth Edition

Jerry Banks
Independent Consultant

John S. Carson II
Brooks Automation

Barry L. Nelson
Northwestern University

David M. Nicol
University of Illinois, Urbana-Champaign

Prentice Hall
is an imprint of

PEARSON

Mathematica and the Mathematica logo are registered trademarks of Wolfram Research Inc., 100 Trade Center Drive, Champaign, IL. 61820-7237

The author and publisher of this book have used their best efforts in preparing this book. These efforts include the development, research, and testing of the theories and programs to determine their effectiveness. The author and publisher make no warranty of any kind, expressed or implied, with regard to these programs or the documentation contained in this book. The author and publisher shall not be liable in any event for incidental and consequential damages in connection with, or arising out of, the furnishing, performance, or use of these programs.

Authorized adaptation from the United States edition, entitled *Discrete-Event System Simulation, 4th Edition*, ISBN: 0131446797 by Banks, Jerry; Carson, John S., II; Nelson, Barry L.; Nicol, David M., published by Pearson Education, Inc., Copyright © 2005.

Indian Subcontinent Adaptation
Copyright © 2007 Dorling Kindersley (India) Pvt. Ltd

This book is sold subject to the condition that it shall not, by way of trade or otherwise, be lent, resold, hired out or otherwise circulated without the publisher's prior written consent in any form of binding or cover other than that in which it is published without a similar condition including this condition being imposed on subsequent purchaser and without limiting the rights under copyright reserved above, no part of this publication may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording or otherwise), without the prior written permission of both the copyright owner and above-mentioned publisher of this book.

Pearson Prentice Hall™ is a trademark of Pearson Education, Inc.
Pearson® is a registered trademark of Pearson Plc.
Prentice Hall® is a registered trademark of Pearson Education, Inc.

S I T Library
Valachil, Mangalore
Accn No: 017209

Srinivas Institute of Technology
Acc. No.: 17209
Call No.:

ISBN 978-81-7758-591-9

10 9 8 7 6 5 4

This edition is manufactured in India and is authorized for sale only in India, Bangladesh, Bhutan, Pakistan, Nepal, Sri Lanka and the Maldives.

Published by Dorling Kindersley (India) Pvt. Ltd., licensees of Pearson Education in South Asia.

Head Office: 7th Floor, Knowledge Boulevard, A-8(A), Sector-62, Noida 201309, UP, India.
Registered Office: 14 Local Shopping Centre, Panchsheel Park, New Delhi 110 017, India.

Printed in India at Baba Barkha Nath Printers.

To
Susie, Jay, Danny, and Zack
Jonna, Jennifer and Jonathan
Sharon and LeRoy
Elizabeth, Caitrin, Thomas, Galen, and John Patrick

Contents

Preface

The objective of this text is to provide a basic treatment of all of the important aspects of discrete-event simulation, with particular emphasis on applications in manufacturing, services, and computing. The fourth edition, like earlier editions, is meant for an upper-level-undergraduate or master's-level introduction to simulation or for a second course with applications. We have updated the material extensively, revised some chapters completely, and added a new chapter on the simulation of computer networks. The associated Web site www.bcnn.net is now closely integrated with the text, and all students and instructors should visit the site.

Chapter 1, Introduction to Simulation, has been generally updated, and every example in Chapter 2, Simulation Examples, has an Excel spreadsheet solution on the Web site. Exercises have been prepared that require downloading these spreadsheet solutions and using them. To reflect the continuing evolution of simulation software, Chapter 3, General Principles, has been modernized to include properties and operations of current simulation languages; in Chapter 4, Simulation Software, simulation in Java replaces C++. We also have maintained an up-to-date discussion of the features of currently available simulation software. Simulation software changes so rapidly, however, that we point to the Web sites of all of the software vendors mentioned in the text.

Chapter 5, Statistical Models in Simulation, incorporates some additional models: the beta and negative binomial distributions and the nonstationary Poisson process. These are backed up by new material on simulating (Chapter 8, Random-Variate Generation) and fitting (Chapter 9, Input Modeling) the models. For clarity, Chapter 8 has been substantially reorganized. In Chapter 7, Random-Number Generation, we have deemphasized statistical testing of random-number generators since the period length of modern generators has become so long that sampling-based tests are no longer feasible.

Chapter 10, Verification and Validation, replaces hypothesis testing by a confidence-interval approach for input-output validation.

The core chapters on the analysis of simulation output are Chapter 11, Output Analysis for a Single Model, and Chapter 12, Comparison and Evaluation of Alternative System Designs. Chapter 11 has been significantly reorganized, and there is new material on prediction intervals and on estimating probabilities and quantiles from only summary statistics. Chapter 12 contains a new procedure for screening a large number of system designs to extract a smaller group of the best.

Chapter 13, Simulation of Manufacturing and Material-Handling Systems, adds an extended example and analysis of a small manufacturing system.

Chapter 14, Simulation of Computer Systems, replaces the discussion of C++ simulation tools for computer simulation with one focused on Java in general and on the SSFNet simulator in particular. Chapter 15, Simulation of Computer Networks, is new. The Web site has examples (in Java) of simulations discussed in this Chapter and provides extensive links to supporting material.

Discrete-Event System Simulation can serve as a textbook in the following types of courses:

An introductory simulation course in engineering, computer science, or management (Chapters 1–9 and selected parts of Chapters 10–12 when no companion language text is used; if a companion language text is used, skip Chapter 4, and use the application Chapters 13, 14, and 15, as appropriate);

A second course in simulation (all of Chapters 10–12, a companion language text, and an outside project; add Chapter 13, 14, or 15, as appropriate)

We gratefully acknowledge the cheerful aid of Gamze Tokol and Dave Goldsman in converting some of the chapters to LaTeX for those coauthors who do not speak the language, and the assistance of Feng Yang, who checked the references and exercises in many chapters.

JERRY BANKS
JOHN S. CARSON II
BARRY L. NELSON
DAVID M. NICOL

About the Authors

Jerry Banks retired in 1999 as a professor in the School of Industrial and Systems Engineering, Georgia Institute of Technology, after which he worked as senior simulation technology advisor for Brooks Automation; he is currently an independent consultant. He is the author, coauthor, editor, or coeditor of eleven books, one set of proceedings, several chapters in texts, and numerous technical papers. He is the editor of the *Handbook of Simulation*, published in 1998 by John Wiley, which won the award for Excellence in Engineering Handbooks from the Professional Scholarly Publishing Division of the Association of American Publishers, Inc. He is also author or coauthor of *Getting Started with AutoMod*, Second Edition, *Introduction to SIMAN V and CINEMA V*, *Getting Started with GPSS/H*, Second Edition, *Forecasting and Management of Technology* and *Principles of Quality Control.* He was a founding partner in the simulation-consulting firm Carson/Banks & Associates, Inc., which was purchased by AutoSimulations, Inc. (now part of Brooks Automation). He is a full member of many technical societies, among them the Institute of Industrial Engineers (IIE); he served eight years as that organization's representative to the Board of the Winter Simulation Conference, including two years as board chair. He is the recipient of the INFORMS College on Simulation Distinguished Service Award for 1999 and was named a fellow of IIE in 2002.

John S. Carson II is the consulting technical manager for the AutoMod Group at Brooks Automation. He has over 28 years experience in simulation in a wide range of application areas, including manufacturing, distribution, warehousing and material handling, transportation and rapid transit systems, port operations (container terminals and bulk handling), and health-care systems. Currently, he is involved in the design of next-generation simulation products and in the development of tools to speed up model development for semi-conductor manufacturing, distribution centers, container terminals and other areas of special interest. He co-founded and managed an independent simulation services company for 8 years, has been an independent simulation consultant, and has taught at the Georgia Institute of Technology, the University of Florida, and the University of Wisconsin-Madison.

Barry L. Nelson is Professor in the Department of Industrial Engineering and Management Sciences at Northwestern University and is director of the Master of Engineering Management Program there. His research centers on the design and analysis of computer-simulation experiments on models of stochastic systems,

concentrating on multivariate input modeling and output analysis and on optimization via simulation. He has published numerous papers and two books. He has served as the simulation area editor of *Operations Research* and as president of the INFORMS (then TIMS) College on Simulation, and he has held many positions for the annual Winter Simulation Conference, including program chair in 1997 and board member currently.

David M. Nicol is professor of electrical and computer engineering at the University of Illinois at Urbana-Champaign. He is a long-time contributor in the field of parallel and distributed discrete-event simulations, having written one of the early Ph.D. dissertations on the topic. He has also worked in parallel algorithms, algorithms for mapping workload in parallel architectures, performance analysis, and reliability modeling and analysis. His research contributions extend to 150 articles in leading computer-science journals and conferences. His research is driven largely by problems encountered in industry and government—he has worked closely with researchers at NASA, IBM, AT&T, Bellcore, Motorola, and the Los Alamos, Sandia, and Oak Ridge National Laboratories. His current interests lie in modeling and simulation of very large systems, particularly communications and other infrastructure, with applications in evaluating system security. From 1997 to 2003 he was the editor-in-chief of the *ACM Transactions on Modeling and Computer Simulation.* Professor Nicol is a Fellow of the IEEE.

Part I

Introduction to Discrete-Event System Simulation

1

Introduction to Simulation

A *simulation* is the imitation of the operation of a real-world process or system over time. Whether done by hand or on a computer, simulation involves the generation of an artificial history of a system and the observation of that artificial history to draw inferences concerning the operating characteristics of the real system.

The behavior of a system as it evolves over time is studied by developing a simulation *model*. This model usually takes the form of a set of assumptions concerning the operation of the system. These assumptions are expressed in mathematical, logical, and symbolic relationships between the *entities*, or objects of interest, of the system. Once developed and validated, a model can be used to investigate a wide variety of "what if" questions about the real-world system. Potential changes to the system can first be simulated, in order to predict their impact on system performance. Simulation can also be used to study systems in the design stage, before such systems are built. Thus, simulation modeling can be used both as an analysis tool for predicting the effect of changes to existing systems and as a design tool to predict the performance of new systems under varying sets of circumstances.

In some instances, a model can be developed which is simple enough to be "solved" by mathematical methods. Such solutions might be found by the use of differential calculus, probability theory, algebraic methods, or other mathematical techniques. The solution usually consists of one or more numerical parameters, which are called measures of performance of the system. However, many real-world systems are so complex that models of these systems are virtually impossible to solve mathematically. In these instances, numerical, computer-based simulation can be used to imitate the behavior of the system over time. From the simulation, data are collected as if a real system were being observed. This simulation-generated data is used to estimate the measures of performance of the system.

This book provides an introductory treatment of the concepts and methods of one form of simulation modeling—discrete-event simulation modeling. The first chapter initially discusses when to use simulation, its advantages and disadvantages, and actual areas of its application. Then the concepts of system and model are explored. Finally, an outline is given of the steps in building and using a simulation model of a system.

1.1 WHEN SIMULATION IS THE APPROPRIATE TOOL

The availability of special-purpose simulation languages, of massive computing capabilities at a decreasing cost per operation, and of advances in simulation methodologies have made simulation one of the most widely used and accepted tools in operations research and systems analysis. Circumstances under which simulation is the appropriate tool to use have been discussed by many authors, from Naylor *et al.* [1966] to Shannon [1998]. Simulation can be used for the following purposes:

1. Simulation enables the study of, and experimentation with, the internal interactions of a complex system or of a subsystem within a complex system.
2. Informational, organizational, and environmental changes can be simulated, and the effect of these alterations on the model's behavior can be observed.
3. The knowledge gained during the designing of a simulation model could be of great value toward suggesting improvement in the system under investigation.
4. Changing simulation inputs and observing the resulting outputs can produce valuable insight into which variables are the most important and into how variables interact.
5. Simulation can be used as a pedagogical device to reinforce analytic solution methodologies.
6. Simulation can be used to experiment with new designs or policies before implementation, so as to prepare for what might happen.
7. Simulation can be used to verify analytic solutions.
8. Simulating different capabilities for a machine can help determine the requirements on it.
9. Simulation models designed for training make learning possible without the cost and disruption of on-the-job instruction.
10. Animation shows a system in simulated operation so that the plan can be visualized.
11. The modern system (factory, wafer fabrication plant, service organization, etc.) is so complex that its internal interactions can be treated only through simulation.

1.2 WHEN SIMULATION IS NOT APPROPRIATE

This section is based on an article by Banks and Gibson [1997], who gave ten rules for evaluating when simulation is not appropriate. The first rule indicates that simulation should not be used when the problem can be solved by common sense. An example is given of an automobile tag facility serving customers who arrive randomly at an average rate of 100/hour and are served at a mean rate of 12/hour. To determine the minimum number of servers needed, simulation is not necessary. Just compute 100/12 = 8.33 indicating that nine or more servers are needed.

The second rule says that simulation should not be used if the problem can be solved analytically. For example, under certain conditions, the average waiting time in the example above can be found from curves that were developed by Hillier and Lieberman [2002].

The next rule says that simulation should not be used if it is easier to perform direct experiments. An example of a fast-food drive-in restaurant is given where it was less expensive to stage a person taking orders using a hand-held terminal and voice communication to determine the effect of adding another order station on customer waiting time.

The fourth rule says not to use simulation if the costs exceed the savings. There are many steps in completing a simulation, as will be discussed in Section 1.11, and these must be done thoroughly. If a simulation study costs $20,000 and the savings might be $10,000, simulation would not be appropriate.

Rules five and six indicate that simulation should not be performed if the resources or time are not available. If the simulation is estimated to cost $20,000 and there is only $10,000 available, the suggestion is not to

venture into a simulation study. Similarly, if a decision in needed in two weeks and a simulation will take a month, the simulation study is not advised.

Simulation takes data, sometimes lots of data. If no data is available, not even estimates, simulation is not advised. The next rule concerns the ability to verify and validate the model. If there is not enough time or if the personnel are not available, simulation is not appropriate.

If managers have unreasonable expectations, if they ask for too much too soon, or if the power of simulation is overestimated, simulation might not be appropriate.

Last, if system behavior is too complex or can't be defined, simulation is not appropriate. Human behavior is sometimes extremely complex to model.

1.3 ADVANTAGES AND DISADVANTAGES OF SIMULATION

Simulation is intuitively appealing to a client because it mimics what happens in a real system or what is perceived for a system that is in the design stage. The output data from a simulation should directly correspond to the outputs that could be recorded from the real system. Additionally, it is possible to develop a simulation model of a system without dubious assumptions (such as the same statistical distribution for every random variable) of mathematically solvable models. For these and other reasons, simulation is frequently the technique of choice in problem solving.

In contrast to optimization models, simulation models are "run" rather than solved. Given a particular set of input and model characteristics, the model is run and the simulated behavior is observed. This process of changing inputs and model characteristics results in a set of scenarios that are evaluated. A good solution, either in the analysis of an existing system or in the design of a new system, is then recommended for implementation.

Simulation has many advantages, but some disadvantages. These are listed by Pegden, Shannon, and Sadowski [1995]. Some advantages are these:

1. New policies, operating procedures, decision rules, information flows, organizational procedures, and so on can be explored without disrupting ongoing operations of the real system.
2. New hardware designs, physical layouts, transportation systems, and so on can be tested without committing resources for their acquisition.
3. Hypotheses about how or why certain phenomena occur can be tested for feasibility.
4. Time can be compressed or expanded to allow for a speed-up or slow-down of the phenomena under investigation.
5. Insight can be obtained about the interaction of variables.
6. Insight can be obtained about the importance of variables to the performance of the system.
7. Bottleneck analysis can be performed to discover where work in process, information, materials, and so on are being delayed excessively.
8. A simulation study can help in understanding how the system operates rather than how individuals think the system operates.
9. "What if" questions can be answered. This is particularly useful in the design of new systems.

Some disadvantages are these:

1. Model building requires special training. It is an art that is learned over time and through experience. Furthermore, if two models are constructed by different competent individuals, they might have similarities, but it is highly unlikely that they will be the same.

2. Simulation results can be difficult to interpret. Most simulation outputs are essentially random variables (they are usually based on random inputs), so it can be hard to distinguish whether an observation is a result of system interrelationships or of randomness.
3. Simulation modeling and analysis can be time consuming and expensive. Skimping on resources for modeling and analysis could result in a simulation model or analysis that is not sufficient to the task.
4. Simulation is used in some cases when an analytical solution is possible, or even preferable, as was discussed in Section 1.2. This might be particularly true in the simulation of some waiting lines where closed-form queueing models are available.

In defense of simulation, these four disadvantages, respectively, can be offset as follows:

1. Vendors of simulation software have been actively developing packages that contain models that need only input data for their operation. Such models have the generic tag "simulator" or "template."
2. Many simulation software vendors have developed output-analysis capabilities within their packages for performing very thorough analysis.
3. Simulation can be performed faster today than yesterday and will be even faster tomorrow, because of advances in hardware that permit rapid running of scenarios and because of advances in many simulation packages. For example, some simulation software contains constructs for modeling material handling that uses such transporters as fork-lift trucks, conveyors, and automated guided vehicles.
4. Closed-form models are not able to analyze most of the complex systems that are encountered in practice. In many years of consulting practice by two of the authors, not one problem was encountered that could have been solved by a closed-form solution.

1.4 AREAS OF APPLICATION

The applications of simulation are vast. The Winter Simulation Conference (WSC) is an excellent way to learn more about the latest in simulation applications and theory. There are also numerous tutorials at both the beginning and the advanced levels. WSC is sponsored by six technical societies and the National Institute of Standards and Technology (NIST). The technical societies are American Statistical Association (ASA), Association for Computing Machinery/Special Interest Group on Simulation (ACM/SIGSIM), Institute of Electrical and Electronics Engineers: Computer Society (IEEE/CS), Institute of Electrical and Electronics Engineers: Systems, Man and Cybernetics Society (IEEE/SMCS), Institute of Industrial Engineers (IIE), Institute for Operations Research and the Management Sciences: College on Simulation (INFORMS/CS) and The Society for Computer Simulation (SCS). Note that IEEE is represented by two bodies. Information about the upcoming WSC can be obtained from `www.wintersim.org`. WSC programs with full papers are available from `www.informs-cs.org/wscpapers.html`. Some presentations, by area, from a recent WSC are listed next:

Manufacturing Applications

Dynamic modeling of continuous manufacturing systems, using analogies to electrical systems
Benchmarking of a stochastic production planning model in a simulation test bed
Paint line color change reduction in automobile assembly
Modeling for quality and productivity in steel cord manufacturing
Shared resource capacity analysis in biotech manufacturing
Neutral information model for simulating machine shop operations

Semiconductor Manufacturing

Constant time interval production planning with application to work-in-process control
Accelerating products under due-date oriented dispatching rules
Design framework for automated material handling systems in 300-mm wafer fabrication factories

Making optimal design decisions for next-generation dispensing tools
Application of cluster tool modeling in a 300-mm wafer fabrication factory
Resident-entity based simulation of batch chamber tools in 300-mm semiconductor manufacturing

Construction Engineering and Project Management

Impact of multitasking and merge bias on procurement of complex equipment
Application of lean concepts and simulation for drainage operations maintenance crews
Building a virtual shop model for steel fabrication
Simulation of the residential lumber supply chain

Military Applications

Frequency-based design for terminating simulations: A peace-enforcement example
A multibased framework for supporting military-based interactive simulations in 3D environments
Specifying the behavior of computer-generated forces without programming
Fidelity and validity: Issues of human behavioral representation
Assessing technology effects on human performance through trade-space development and evaluation
Impact of an automatic logistics system on the sortie-generation process
Research plan development for modeling and simulation of military operations in urban terrain

Logistics, Supply Chain, and Distribution Applications

Inventory analysis in a server–computer manufacturing environment
Comparison of bottleneck detection methods for AGV systems
Semiconductor supply-network simulation
Analysis of international departure passenger flows in an airport terminal
Application of discrete simulation techniques to liquid natural gas supply chains
Online simulation of pedestrian flow in public buildings

Transportation Modes and Traffic

Simulating aircraft-delay absorption
Runway schedule determination by simulation optimization
Simulation of freeway merging and diverging behavior
Modeling ambulance service of the Austrian Red Cross
Simulation modeling in support of emergency firefighting in Norfolk
Modeling ship arrivals in ports
Optimization of a barge transportation system for petroleum delivery
Iterative optimization and simulation of barge traffic on an inland waterway

Business Process Simulation

Agent-based modeling and simulation of store performance for personalized pricing
Visualization of probabilistic business models
Modeling and simulation of a telephone call center
Using simulation to approximate subgradients of convex performance measures in service systems
Simulation's role in baggage screening at airports
Human-fatigue risk simulations in continuous operations
Optimization of a telecommunications billing system
Segmenting the customer base for maximum returns

Health Care

Modeling front office and patient care in ambulatory health care practices
Evaluation of hospital operations between the emergency department and a medical telemetry unit
Estimating maximum capacity in an emergency room
Reducing the length of stay in an emergency department
Simulating six-sigma improvement ideas for a hospital emergency department
A simulation–integer-linear-programming-based tool for scheduling emergency room staff

Some general trends in simulation applications are as follows: At present, simulation for risk analysis is growing, including in such areas as insurance, options pricing, and portfolio analysis. Another growing area is call-center analysis, which is not amenable to queuing models because of its complexity. Simulation of large-scale systems such as the internet backbone, wireless networks, and supply chains are growing as hardware and software increase their capability to handle extremely large numbers of entities in a reasonable time.

Lastly, simulation models of automated material handling systems (AMHS) are being used as test beds for the development and functional testing of control-system software. Called an emulation, the simulation model is connected in real time to the control-system software or to a software emulator; it is used to provide the same responses to a control system as the real AMHS does (for example, a box blocking or clearing a photo eye, or a command to start picking an order). Software development can begin much earlier during AMHS installation and commissioning, to reduce the time spent in the field on trying to debug control software while attempting to ramp up a new system or continue running an existing one. Models have been driven by control systems at various levels—from high-level supervisory systems, such as warehouse management systems (WMS) or AGV dispatching systems, to such low-level control as programmable logic controllers (PLCs) controlling merges on a conveyor system.

1.5 SYSTEMS AND SYSTEM ENVIRONMENT

To model a system, it is necessary to understand the concept of a system and the system boundary. A *system* is defined as a group of objects that are joined together in some regular interaction or interdependence toward the accomplishment of some purpose. An example is a production system manufacturing automobiles. The machines, component parts, and workers operate jointly along an assembly line to produce a high-quality vehicle.

A system is often affected by changes occurring outside the system. Such changes are said to occur in the *system environment* [Gordon, 1978]. In modeling systems, it is necessary to decide on the *boundary* between the system and its environment. This decision may depend on the purpose of the study.

In the case of the factory system, for example, the factors controlling the arrival of orders may be considered to be outside the influence of the factory and therefore part of the environment. However, if the effect of supply on demand is to be considered, there will be a relationship between factory output and arrival of orders, and this relationship must be considered an activity of the system. Similarly, in the case of a bank system, there could be a limit on the maximum interest rate that can be paid. For the study of a single bank, this would be regarded as a constraint imposed by the environment. In a study of the effects of monetary laws on the banking industry, however, the setting of the limit would be an activity of the system. [Gordon, 1978]

1.6 COMPONENTS OF A SYSTEM

In order to understand and analyze a system, a number of terms need to be defined. An *entity* is an object of interest in the system. An *attribute* is a property of an entity. An *activity* represents a time period of specified length. If a bank is being studied, customers might be one of the entities, the balance in their checking accounts might be an attribute, and making deposits might be an activity.

The collection of entities that compose a system for one study might only be a subset of the overall system for another study [Law and Kelton, 2000]. For example, if the aforementioned bank is being studied to determine the number of tellers needed to provide for paying and receiving, the system can be defined as that portion of the bank consisting of the regular tellers and the customers waiting in line. If the purpose of the study is expanded to determine the number of special tellers needed (to prepare cashier's checks, to sell traveler's checks, etc.), the definition of the system must be expanded.

The *state* of a system is defined to be that collection of variables necessary to describe the system at any time, relative to the objectives of the study. In the study of a bank, possible state variables are the number of busy tellers, the number of customers waiting in line or being served, and the arrival time of the next customer. An *event* is defined as an instantaneous occurrence that might change the state of the system. The term *endogenous* is used to describe activities and events occurring within a system, and the term *exogenous* is used to describe activities and events in the environment that affect the system. In the bank study, the arrival of a customer is an exogenous event, and the completion of service of a customer is an endogenous event.

Table 1.1 lists examples of entities, attributes, activities, events, and state variables for several systems. Only a partial listing of the system components is shown. A complete list cannot be developed unless the purpose of the study is known. Depending on the purpose, various aspects of the system will be of interest, and then the listing of components can be completed.

1.7 DISCRETE AND CONTINUOUS SYSTEMS

Systems can be categorized as discrete or continuous. "Few systems in practice are wholly discrete or continuous, but since one type of change predominates for most systems, it will usually be possible to classify a system as being either discrete or continuous" [Law and Kelton, 2000]. A *discrete system* is one in which the state variable(s) change only at a discrete set of points in time. The bank is an example of a discrete system: The state variable, the number of customers in the bank, changes only when a customer arrives or when the service provided a customer is completed. Figure 1.1 shows how the number of customers changes only at discrete points in time.

A *continuous system* is one in which the state variable(s) change continuously over time. An example is the head of water behind a dam. During and for some time after a rain storm, water flows into the lake behind the dam. Water is drawn from the dam for flood control and to make electricity. Evaporation also decreases the water level. Figure 1.2 shows how the state variable *head of water behind the dam* changes for this continuous system.

1.8 MODEL OF A SYSTEM

Sometimes it is of interest to study a system to understand the relationships between its components or to predict how the system will operate under a new policy. To study the system, it is sometimes possible to experiment with the system itself. However, this is not always possible. A new system might not yet exist; it could be in only hypothetical form or at the design stage. Even if the system exists, it might be impractical to experiment with it. For example, it might not be wise or possible to double the unemployment rate to discover the effect of employment on inflation. In the case of a bank, reducing the numbers of tellers to study the effect on the length of waiting lines might infuriate the customers so greatly that they move their accounts to a competitor. Consequently, studies of systems are often accomplished with a model of a system.

We had a consulting job for the simulation of a redesigned port in western Australia. At $200 millions for a loading/unloading berth, it's not advisable to invest that amount only to find that the berth is inadequate for the task.

A *model* is defined as a representation of a system for the purpose of studying the system. For most studies, it is only necessary to consider those aspects of the system that affect the problem under investigation. These aspects are represented in a model of the system; the model, by definition, is a simplification of the system. On the other hand, the model should be sufficiently detailed to permit valid conclusions to be drawn about the real system. Different models of the same system could be required as the purpose of investigation changes.

Table 1.1 Examples of Systems and Components

System	*Entities*	*Attributes*	*Activities*	*Events*	*State Variables*
Banking	Customers	Checking-account balance	Making deposits	Arrival; departure	Number of busy tellers; number of customers waiting
Rapid rail	Riders	Origination; destination	Traveling	Arrival at station; arrival at destination	Number of riders waiting at each station; number of riders in transit
Production	Machines	Speed; capacity; breakdown rate	Welding; stamping	Breakdown	Status of machines (busy, idle, or down)
Communications	Messages	Length; destination	Transmitting	Arrival at destination	Number waiting to be transmitted
Inventory	Warehouse	Capacity	Withdrawing	Demand	Levels of inventory; backlogged demands

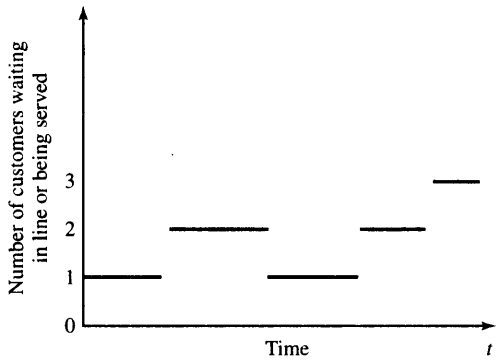

Figure 1.1 Discrete-system state variable.

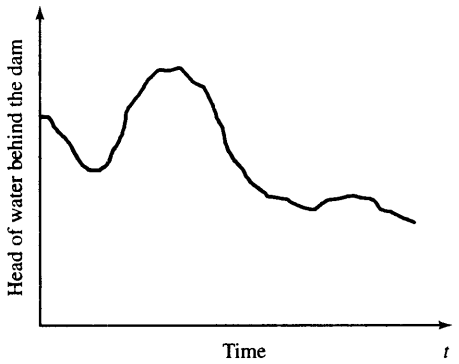

Figure 1.2 Continuous-system state variable.

Just as the components of a system were entities, attributes, and activities, models are represented similarly. However, the model contains only those components that are relevant to the study. The components of a model are discussed more extensively in Chapter 3.

1.9 TYPES OF MODELS

Models can be classified as being mathematical or physical. A mathematical model uses symbolic notation and mathematical equations to represent a system. A simulation model is a particular type of mathematical model of a system.

Simulation models may be further classified as being static or dynamic, deterministic or stochastic, and discrete or continuous. A *static* simulation model, sometimes called a Monte Carlo simulation, represents a system at a particular point in time. *Dynamic* simulation models represent systems as they change over time. The simulation of a bank from 9:00 A.M. to 4:00 P.M. is an example of a dynamic simulation.

Simulation models that contain no random variables are classified as *deterministic*. Deterministic models have a known set of inputs, which will result in a unique set of outputs. Deterministic arrivals would occur at a dentist's office if all patients arrived at the scheduled appointment time. A *stochastic* simulation model has one or more random variables as inputs. Random inputs lead to random outputs. Since the outputs are

random, they can be considered only as estimates of the true characteristics of a model. The simulation of a bank would usually involve random interarrival times and random service times. Thus, in a stochastic simulation, the output measures—the average number of people waiting, the average waiting time of a customer—must be treated as statistical estimates of the true characteristics of the system.

Discrete and continuous systems were defined in Section 1.7. Discrete and continuous models are defined in an analogous manner. However, a discrete simulation model is not always used to model a discrete system, nor is a continuous simulation model always used to model a continuous system. Tanks and pipes are modeled discretely by some software vendors, even though we know that fluid flow is continuous. In addition, simulation models may be mixed, both discrete and continuous. The choice of whether to use a discrete or continuous (or both discrete and continuous) simulation model is a function of the characteristics of the system and the objective of the study. Thus, a communication channel could be modeled discretely if the characteristics and movement of each message were deemed important. Conversely, if the flow of messages in aggregate over the channel were of importance, modeling the system via continuous simulation could be more appropriate. The models considered in this text are discrete, dynamic, and stochastic.

1.10 DISCRETE-EVENT SYSTEM SIMULATION

This is a textbook about discrete-event system simulation. Discrete-event systems simulation is the modeling of systems in which the state variable changes only at a discrete set of points in time. The simulation models are analyzed by numerical methods rather than by analytical methods. *Analytical* methods employ the deductive reasoning of mathematics to "solve" the model. For example, differential calculus can be used to compute the minimum-cost policy for some inventory models. *Numerical* methods employ computational procedures to "solve" mathematical models. In the case of simulation models, which employ numerical methods, models are "run" rather than solved—that is, an artificial history of the system is generated from the model assumptions, and observations are collected to be analyzed and to estimate the true system performance measures. Real-world simulation models are rather large, and the amount of data stored and manipulated is vast, so such runs are usually conducted with the aid of a computer. However, much insight can be obtained by simulating small models manually.

In summary, this textbook is about discrete-event system simulation in which the models of interest are analyzed numerically, usually with the aid of a computer.

1.11 STEPS IN A SIMULATION STUDY

Figure 1.3 shows a set of steps to guide a model builder in a thorough and sound simulation study. Similar figures and discussion of steps can be found in other sources [Shannon, 1975; Gordon, 1978; Law and Kelton, 2000]. The number beside each symbol in Figure 1.3 refers to the more detailed discussion in the text. The steps in a simulation study are as follows:

Problem formulation. Every study should begin with a statement of the problem. If the statement is provided by the policymakers, or those that have the problem, the analyst must ensure that the problem being described is clearly understood. If a problem statement is being developed by the analyst, it is important that the policymakers understand and agree with the formulation. Although not shown in Figure 1.3, there are occasions where the problem must be reformulated as the study progresses. In many instances, policymakers and analysts are aware that there is a problem long before the nature of the problem is known.

Setting of objectives and overall project plan. The objectives indicate the questions to be answered by simulation. At this point, a determination should be made concerning whether simulation is the

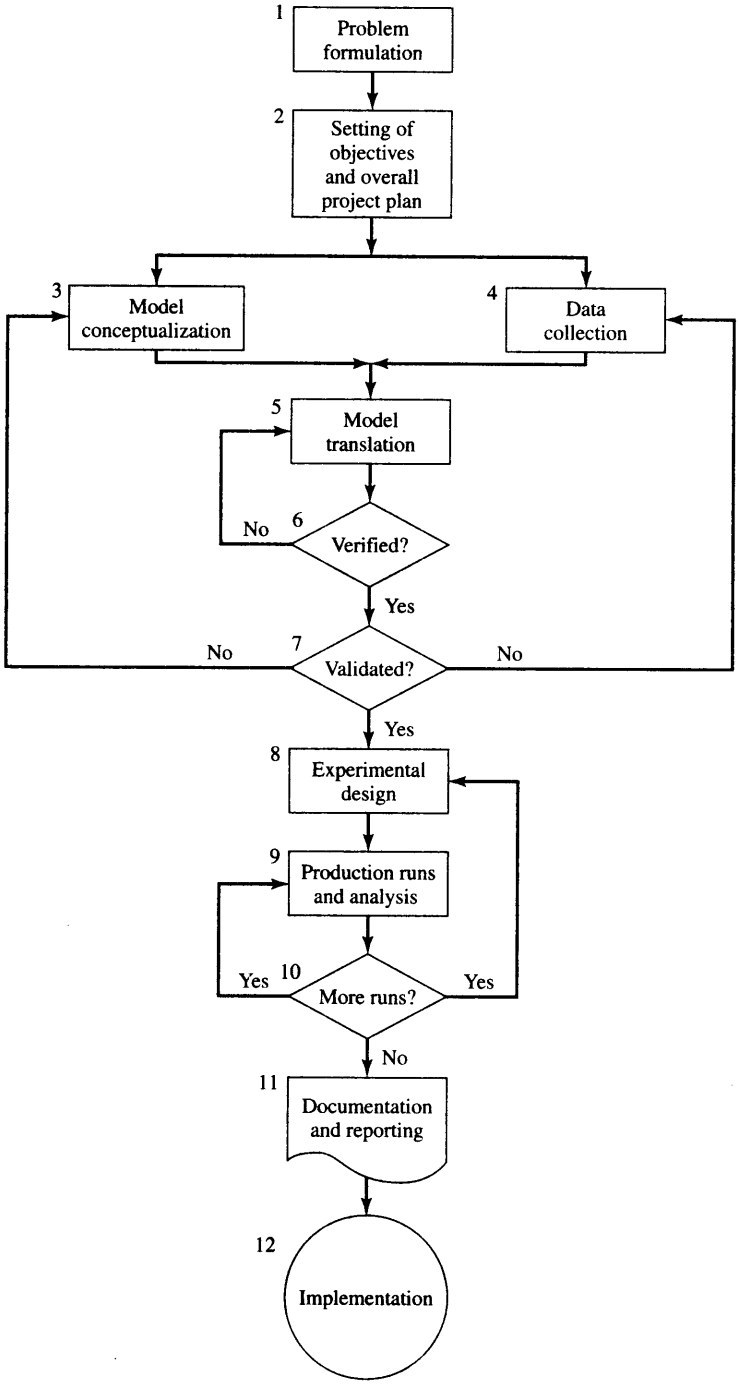

Figure 1.3 Steps in a simulation study.

appropriate methodology for the problem as formulated and objectives as stated. Assuming that it is decided that simulation is appropriate, the overall project plan should include a statement of the alternative systems to be considered and of a method for evaluating the effectiveness of these alternatives. It should also include the plans for the study in terms of the number of people involved, the cost of the study, and the number of days required to accomplish each phase of the work, along with the results expected at the end of each stage.

Model conceptualization. The construction of a model of a system is probably as much art as science. Pritsker [1998] provides a lengthy discussion of this step. "Although it is not possible to provide a set of instructions that will lead to building successful and appropriate models in every instance, there are some general guidelines that can be followed" [Morris, 1967]. The art of modeling is enhanced by an ability to abstract the essential features of a problem, to select and modify basic assumptions that characterize the system, and then to enrich and elaborate the model until a useful approximation results. Thus, it is best to start with a simple model and build toward greater complexity. However, the model complexity need not exceed that required to accomplish the purposes for which the model is intended. Violation of this principle will only add to model-building and computer expenses. It is not necessary to have a one-to-one mapping between the model and the real system. Only the essence of the real system is needed.

It is advisable to involve the model user in model conceptualization. Involving the model user will both enhance the quality of the resulting model and increase the confidence of the model user in the application of the model. (Chapter 2 describes a number of simulation models. Chapter 6 describes queueing models that can be solved analytically. However, only experience with real systems—versus textbook problems—can "teach" the art of model building.)

Data collection. There is a constant interplay between the construction of the model and the collection of the needed input data [Shannon, 1975]. As the complexity of the model changes, the required data elements can also change. Also, since data collection takes such a large portion of the total time required to perform a simulation, it is necessary to begin it as early as possible, usually together with the early stages of model building.

The objectives of the study dictate, in a large way, the kind of data to be collected. In the study of a bank, for example, if the desire is to learn about the length of waiting lines as the number of tellers change, the types of data needed would be the distributions of interarrival times (at different times of the day), the service-time distributions for the tellers, and historic distributions on the lengths of waiting lines under varying conditions. This last type of data will be used to validate the simulation model. (Chapter 9 discusses data collection and data analysis; Chapter 5 discusses statistical distributions that occur frequently in simulation modeling. See also an excellent discussion by Henderson [2003].)

Model translation. Most real-world systems result in models that require a great deal of information storage and computation, so the model must be entered into a computer-recognizable format. We use the term "program" even though it is possible to accomplish the desired result in many instances with little or no actual coding. The modeler must decide whether to program the model in a simulation language, such as GPSS/H (discussed in Chapter 4), or to use special-purpose simulation software. For manufacturing and material handling, Chapter 4 discusses Arena®, AutoMod™, Extend™, Flexsim, MicroSaint, ProModel®, Quest®, SIMUL8®, and WITNESS™. Simulation languages are powerful and flexible. However, if the problem is amenable to solution with the simulation software, the model development time is greatly reduced. Furthermore, most of the simulation-software packages have added features that enhance their flexibility, although the amount of flexibility varies greatly.

Verified? Verification pertains to the computer program prepared for the simulation model. Is the computer program performing properly? With complex models, it is difficult, if not impossible, to translate a model successfully in its entirety without a good deal of debugging; if the input parameters and logical structure of the model are correctly represented in the computer, verification has been completed. For the

most part, common sense is used in completing this step. (Chapter 10 discusses verification of simulation models, and Balci [2003] also discusses this topic.)

Validated? Validation usually is achieved through the calibration of the model, an iterative process of comparing the model against actual system behavior and using the discrepancies between the two, and the insights gained, to improve the model. This process is repeated until model accuracy is judged acceptable. In the example of a bank previously mentioned, data was collected concerning the length of waiting lines under current conditions. Does the simulation model replicate this system measure? This is one means of validation. (Chapter 10 discusses the validation of simulation models, and Balci [2003] also discusses this topic.)

Experimental design. The alternatives that are to be simulated must be determined. Often, the decision concerning which alternatives to simulate will be a function of runs that have been completed and analyzed. For each system design that is simulated, decisions need to be made concerning the length of the initialization period, the length of simulation runs, and the number of replications to be made of each run. (Chapters 11 and 12 discuss issues associated with the experimental design, and Kleijnen [1998] discusses this topic extensively.)

Production runs and analysis. Production runs, and their subsequent analysis, are used to estimate measures of performance for the system designs that are being simulated. (Chapters 11 and 12 discuss the analysis of simulation experiments, and Chapter 4 discusses software to aid in this step, including AutoStat (in AutoMod), OptQuest (in several pieces of simulation software), SimRunner (in ProModel), and WITNESS Optimizer (in WITNESS).

More Runs? Given the analysis of runs that have been completed, the analyst determines whether additional runs are needed and what design those additional experiments should follow.

Documentation and reporting. There are two types of documentation: program and progress. Program documentation is necessary for numerous reasons. If the program is going to be used again by the same or different analysts, it could be necessary to understand how the program operates. This will create confidence in the program, so that model users and policymakers can make decisions based on the analysis. Also, if the program is to be modified by the same or a different analyst, this step can be greatly facilitated by adequate documentation. One experience with an inadequately documented program is usually enough to convince an analyst of the necessity of this important step. Another reason for documenting a program is so that model users can change parameters at will in an effort to learn the relationships between input parameters and output measures of performance or to discover the input parameters that "optimize" some output measure of performance.

Musselman [1998] discusses progress reports that provide the important, written history of a simulation project. Project reports give a chronology of work done and decisions made. This can prove to be of great value in keeping the project on course.

Musselman suggests frequent reports (monthly, at least) so that even those not involved in the day-to-day operation can keep abreast. The awareness of these others can often enhance the successful completion of the project by surfacing misunderstandings early, when the problem can be solved easily. Musselman also suggests maintaining a project log providing a comprehensive record of accomplishments, change requests, key decisions, and other items of importance.

On the reporting side, Musselman suggests frequent deliverables. These may or may not be the results of major accomplishments. His maxim is that "it is better to work with many intermediate milestones than with one absolute deadline." Possibilities prior to the final report include a model specification, prototype demonstrations, animations, training results, intermediate analyses, program documentation, progress reports, and presentations. He suggests that these deliverables should be timed judiciously over the life of the project.

The result of all the analysis should be reported clearly and concisely in a final report. This will enable the model users (now, the decision makers) to review the final formulation, the alternative systems that were addressed, the criterion by which the alternatives were compared, the results of the experiments, and the recommended solution to the problem. Furthermore, if decisions have to be justified at a higher level, the final report should provide a vehicle of certification for the model user/decision maker and add to the credibility of the model and of the model-building process.

Implementation. The success of the implementation phase depends on how well the previous 11 steps have been performed. It is also contingent upon how thoroughly the analyst has involved the ultimate model user during the entire simulation process. If the model user has been thoroughly involved during the model-building process and if the model user understands the nature of the model and its outputs, the likelihood of a vigorous implementation is enhanced [Pritsker, 1995]. Conversely, if the model and its underlying assumptions have not been properly communicated, implementation will probably suffer, regardless of the simulation model's validity.

The simulation-model building process shown in Figure 1.3 can be broken down into four phases. The first phase, consisting of steps 1 (Problem Formulation) and 2 (Setting of Objective and Overall Design), is a period of discovery or orientation. The initial statement of the problem is usually quite "fuzzy," the initial objectives will usually have to be reset, and the original project plan will usually have to be fine-tuned. These recalibrations and clarifications could occur in this phase, or perhaps will occur after or during another phase (i.e., the analyst might have to restart the process).

The second phase is related to model building and data collection and includes steps 3 (Model Conceptualization), 4 (Data Collection), 5 (Model Translation), 6 (Verification), and 7 (Validation). A continuing interplay is required among the steps. Exclusion of the model user during this phase can have dire implications at the time of implementation.

The third phase concerns the running of the model. It involves steps 8 (Experimental Design), 9 (Production Runs and Analysis), and 10 (Additional Runs). This phase must have a thoroughly conceived plan for experimenting with the simulation model. A discrete-event stochastic simulation is in fact a statistical experiment. The output variables are estimates that contain random error, and therefore a proper statistical analysis is required. Such a philosophy is in contrast to that of the analyst who makes a single run and draws an inference from that single data point.

The fourth phase, implementation, involves steps 11 (Documentation and Reporting) and 12 (Implementation). Successful implementation depends on continual involvement of the model user and on the successful completion of every step in the process. Perhaps the most crucial point in the entire process is step 7 (Validation), because an invalid model is going to lead to erroneous results, which, if implemented, could be dangerous, costly, or both.

REFERENCES

BALCI, O. [2003], "Verification, Validation, and Certification of Modeling and Simulation Applications," in *Proceedings of the Winter Simulation Conference*, Ed., S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, New Orleans, LA, Dec. 7–10, pp. 150–158.

BANKS, J., AND R. R. GIBSON [1997], "Don't Simulate When: 10 Rules for Determining when Simulation Is Not Appropriate," *IIE Solutions*, September.

GORDON, G. [1978], *System Simulation*, 2d ed., Prentice-Hall, Englewood Cliffs, NJ.

HENDERSON, S. G. [2003], "Input Model Uncertainty: Why Do We Care and What Should We Do About It? " in *Proceedings of the Winter Simulation Conference*, Ed., S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, New Orleans, LA, Dec. 7–10, pp. 90–100.

HILLIER, F. S., AND G. J. LIEBERMAN [2002], *Introduction to Operations Research*, 7th ed., McGraw–Hill, New York.

KLEIJNEN, J. P. C. [1998], "Experimental Design for Sensitivity Analysis, Optimization, and Validation of Simulation Models," in *Handbook of Simulation*, Ed., Jerry Banks, John Wiley, New York.

LAW, A. M., AND W. D. KELTON [2000], *Simulation Modeling and Analysis*, 3d ed., McGraw–Hill, New York.

MORRIS, W. T. [1967], "On the Art of Modeling," *Management Science*, Vol. 13, No. 12.

MUSSELMAN, K. J. [1998], "Guidelines for Success," in *Handbook of Simulation*, Ed., Jerry Banks, John Wiley, New York.

NAYLOR, T. H., J. L. BALINTFY, D. S. BURDICK, AND K. CHU [1966], *Computer Simulation Techniques*, Wiley, New York.

PEGDEN, C. D., R. E. SHANNON, AND R. P. SADOWSKI [1995], *Introduction to Simulation Using SIMAN*, 2d ed., McGraw–Hill, New York.

PRITSKER, A. A. B. [1995], *Introduction to Simulation and SLAM II*, 4th ed., Wiley & Sons, New York.

PRITSKER, A. A. B. [1998], "Principles of Simulation Modeling," in *Handbook of Simulation*, Ed., Jerry Banks, John Wiley, New York.

SHANNON, R. E. [1975], *Systems Simulation: The Art and Science*, Prentice-Hall, Englewood Cliffs, NJ.

SHANNON, R. E. [1998], "Introduction to the Art and Science of Simulation," in *Proceedings of the Winter Simulation Conference*, Eds., D. J. Medeiros, E. F. Watson, J. S. Carson, and M. S. Manivannan, Washington, DC, Dec. 13–16, pp. 7–14.

EXERCISES

1. Name entities, attributes, activities, events, and state variables for the following systems:
   - **(a)** University library
   - **(b)** Bank
   - **(c)** Call center
   - **(d)** Hospital blood bank
   - **(e)** Departmental store
   - **(f)** Fire service station
   - **(g)** Airport
   - **(h)** Software organization
2. Consider the simulation process shown in Figure 1.3.
   - **(a)** Reduce the steps by at least two by combining similar activities. Give your rationale.
   - **(b)** Increase the steps by at least two by separating current steps or enlarging on existing steps. Give your rationale.
3. A simulation of a major traffic intersection is to be conducted, with the objective of improving the current traffic flow. Provide three iterations, in increasing order of complexity, of steps 1 and 2 in the simulation process of Figure 1.3.
4. A simulation is to be conducted of cooking a spaghetti dinner to discover at what time a person should start in order to have the meal on the table by 7:00 P.M. Read a recipe for preparing a spaghetti dinner (or ask a friend or relative for the recipe). As best you can, trace what you understand to be needed, in the data-collection phase of the simulation process of Figure 1.3, in order to perform a simulation in which the model includes each step in the recipe. What are the events, activities, and state variables in this system?
5. List down the events and activities applying for master's program in a university.
6. Read an article on the application of simulation related to your major area of study or interest, in the current WSC Proceedings, and prepare a report on how the author accomplishes the steps given in Figure 1.3.

7. Get a copy of a recent WSC Proceedings and report on the different applications discussed in an area of interest to you.
8. Get a copy of a recent WSC Proceedings and report on the most unusual application that you can find.
9. Go to the Winter Simulation Conference website at: `www.wintersim.org` and address the following:
   (a) What advanced tutorials were offered at the previous WSC or are planned at the next WSC?
   (b) Where and when will the next WSC be held?
10. Go to the Winter Simulation Conference website at `www.wintersim.org` and address the following:
   (a) When was the largest (in attendance) WSC, and how many attended?
   (b) In what calendar year, from the beginning of WSC, was there no Conference?
   (c) What was the largest expanse of time, from the beginning of WSC, between occurrences of the Conference?
   (d) Beginning with the 25th WSC, can you discern a pattern for the location of the Conference?
11. Search the web for "Applications of discrete simulation" and prepare a report based on the findings.
12. Search the web for "Manufacturing simulation" and prepare a report based on the findings.
13. Search the web for "Call center simulation" and prepare a report based on the findings.

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 are carried out by following three steps:

1. Determine the characteristics of each of the inputs to the simulation. Quite often, these are 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 (or, trials) $i = 1, 2,\ldots, n$. 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 chosen 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, reliability, and network analysis. The two queueing examples provide a single-server and two-server system, respectively. (Chapter 6 provides more insight into queueing models.) The first of the inventory examples 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.

Next, there is an example that introduces the concept of random normal numbers and a model for the simulation of lead-time demand. The examples conclude with the analysis of a network.

Table 2.1 Simulation Table

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

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.

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 could 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 multichannel 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 re-entrant queueing networks in which units return a number of times to the same server before finally exiting from the system, the condition that arrival rate be less than service rate might 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 manufacturing plants.) More complex situations can 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 most basic queues.

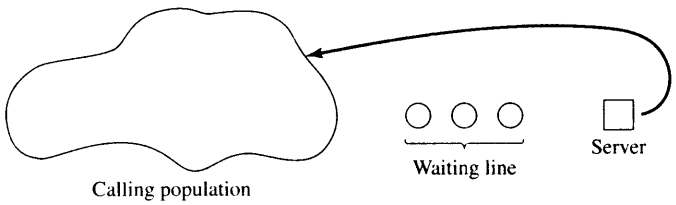

Figure 2.1 Queueing system.

Prior to our 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 causes 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) and 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 the 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.

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 will 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. 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 while the queue is nonempty.

After the completion of a service, the server either will become idle or will 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

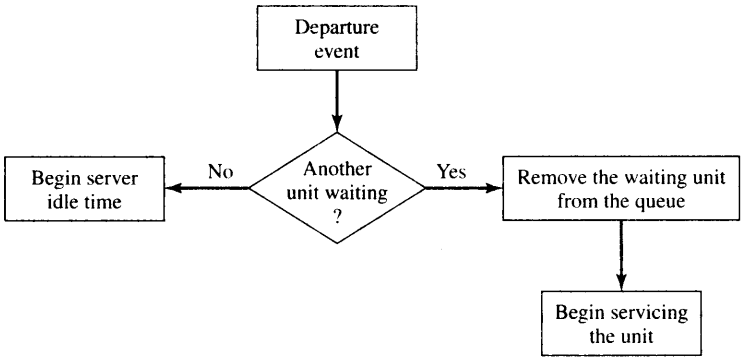

Figure 2.2 *Service just completed* flow diagram.

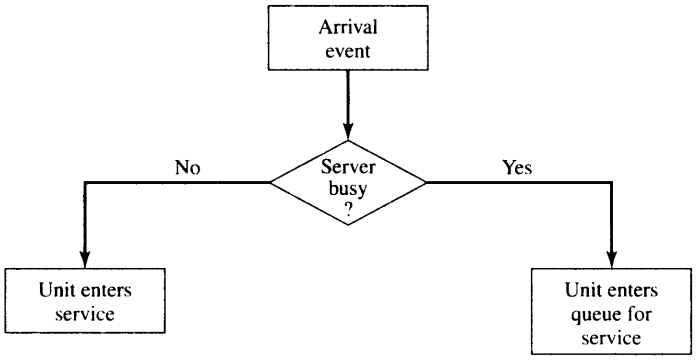

Figure 2.3 *Unit entering system* flow diagram.

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

Figure 2.4 Potential unit actions upon arrival.

		Queue status	
		Not empty	Empty
Server outcomes	Busy		Impossible
	Idle	Impossible	

Figure 2.5 Server outcomes after the completion of service.

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.

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 either from data collected and analyzed or from 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,..., 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 also can 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 by using a procedure, they are often referred to as pseudo-random numbers. Because the procedure is fully known, it is always possible to predict 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 simulation, 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

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 clock time 2. The third customer arrives four time units later, at clock time 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 is 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 that serves customers on a first-in–first-out (FIFO) basis. It keeps track of the clock time 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.

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 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

Table 2.3 Service Times

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

Table 2.4 Simulation Table Emphasizing Clock Times

A *Customer Number*	B *Arrival Time (Clock)*	C *Time Service Begins (Clock)*	D *Service Time (Duration)*	E *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

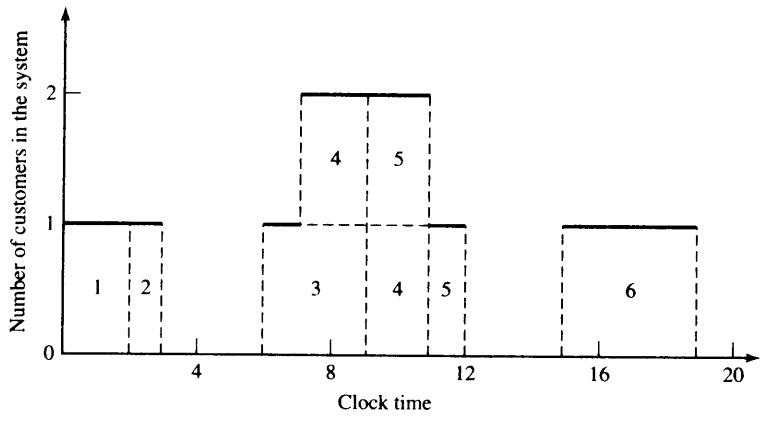

Figure 2.6 Number of customers in the system.

time 6. During some time periods, two customers are in the system, such as at clock time 8, when customers 3 and 4 are both 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 times that are 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 100 customers.

In actuality, 100 customers is too small a sample size to draw any reliable conclusions. The accuracy of the results is enhanced by increasing the sample size, as is 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 calculations simple, starting conditions and concerns are overlooked.

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

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

A set of uniformly distributed random numbers is needed to generate the arrivals at the checkout counter. Such 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. 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 99 random numbers to generate the times between arrivals. Why only 99 numbers? The first arrival is assumed to occur at time 0, so only 99 more arrivals need to be generated to end up with 100 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-arrival of 1 minute is 0.125, so 125 of the 1000 random digit values are assigned to such an occurrence. Times between arrival for 99 customers are generated by listing 99 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 from the same pattern's being generated.

The time-between-arrival determination is shown in Table 2.8. Note that the first random digits are 064. To obtain the corresponding time between arrivals, enter the fourth column of Table 2.6 and read 1 minute from the first column of the table. Alternatively, we see that 0.064 is between the cumulative probabilities 0.001 and 0.125, again resulting in 1 minute as the generated time.

Service times for the first 18 and the 100th customers are shown in Table 2.9. These service times were generated via the methodology described above, together with the aid of Table 2.7. (The entire table can be generated by using the Excel spreadsheet for Example 2.1 at www.bcnn.net.) 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 2.8 Time-Between-Arrival Determination

Customer	*Random Digits*	*Time between Arrivals (Minutes)*	*Customer*	*Random Digits*	*Time between Arrivals (Minutes)*
1	—	—	11	413	4
2	064	1	12	462	4
3	112	1	13	843	7
4	678	6	14	738	6
5	289	3	15	359	3
6	871	7	16	888	8
7	583	5	17	902	8
8	139	2	18	212	2
9	423	4	⋮	⋮	⋮
10	039	1	100	538	5

Table 2.9 Service Times Generated

Customer	*Random Digits*	*Service Time (Minutes)*	*Customer*	*Random Digits*	*Service Time (Minutes)*
1	84	4	11	94	5
2	18	2	12	32	3
3	87	5	13	79	4
4	81	4	14	92	5
5	06	1	15	46	3
6	91	5	16	21	2
7	79	4	17	73	4
8	09	1	18	55	3
9	64	4	⋮	⋮	⋮
10	38	3	100	26	2

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, service time, and the completion time of the previous customer. For example, the second customer arrives at time 1. But service could not begin until time 4; the server (checkout person) was busy until that time. The second customer waited in the queue for three minutes. The second customer was in the system for 5 minutes. Skip down to the fifth customer. Service ends at time 16, but the sixth customer does not arrive until time 18, at which time service began. The server (checkout person) was idle for two minutes. This process continues for all 100 customers. The rightmost two columns have been added to collect statistical measures of performance, such as each customer's time in 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.

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

1. The average waiting time for a customer is 1.74 minutes. This is computed in the following manner:

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

$$= \frac{174}{100} = 1.74 \text{ minutes}$$

2. The probability that a customer has to wait in the queue is 0.46. This is computed in the following manner:

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

$$= \frac{46}{100} = 0.46$$

Table 2.10 Simulation Table for Single-Channel Queueing Problem

		Clock		*Clock*		*Clock*		
Customer	*Interarrival Time (Minutes)*	*Arrival Time*	*Service Time (Minutes)*	*Time Service Begins*	*Waiting Time in Queue (Minutes)*	*Time Service Ends*	*Time Customer Spends in System (Minutes)*	*Idle Time of Server (Minutes)*
1		0	4	0	0	4	4	
2	1	1	2	4	3	6	5	0
3	1	2	5	6	4	11	9	0
4	6	8	4	11	3	15	7	0
5	3	11	1	15	4	16	5	0
6	7	18	5	18	0	23	5	2
7	5	23	4	23	0	27	4	0
8	2	25	1	27	2	28	3	0
9	4	29	4	29	0	33	4	1
10	1	30	3	33	3	36	6	0
11	4	34	5	36	2	41	7	0
12	4	38	3	41	3	44	6	0
13	7	45	4	45	0	49	4	1
14	6	51	5	51	0	56	5	2
15	3	54	3	56	2	59	5	0
16	8	62	2	62	0	64	2	3
17	8	70	4	70	0	74	4	6
18	2	72	3	74	2	77	5	0
19	7	79	1	79	0	80	1	2
20	4	83	2	83	0	85	2	3
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
100	5	415	2	416	1	418	3	0
Total	415		317		174		491	101

3. The proportion of idle time of the server is 0.24. This is computed in the following manner:

$$\text{probability of idle server} = \frac{\text{total idle time of server (minutes)}}{\text{total run time of simulation (minutes)}}$$
$$= \frac{101}{418} = 0.24$$

The probability of the server's being busy is the complement of 0.24, namely, 0.76.

4. The average service time is 3.17 minutes. This is computed in the following manner:

$$\text{Average service time (minutes)} = \frac{\text{total service time (minutes)}}{\text{total number of customers}}$$
$$= \frac{317}{100} = 3.17 \text{ minutes}$$

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

Expected service time =
$1(0.10) + 2(0.20) + 3(0.30) + 4(0.25) + 5(0.10) + 6(0.05) = 3.2$ minutes

The expected service time is slightly higher 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 4.19 minutes. This is computed in the following manner:

$$\text{Average time between arrivals (minutes)} = \frac{\text{sum of all times between arrival (minutes)}}{\text{number of arrivals} - 1}$$
$$= \frac{415}{99} = 4.19 \text{ minutes}$$

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 should approach the theoretical mean, $E(A)$.

6. The average waiting time of those who wait is 3.22 minutes. This is computed in the following manner:

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

$$= \frac{174}{54} = 3.22 \text{ minutes}$$

7. The average time a customer spends in the system is 4.91 minutes. This can be found in two ways. First, the computation can be achieved by the following relationship:

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

$$= \frac{491}{100} = 4.91 \text{ minutes}$$

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

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

From findings 1 and 4, this results in

$$\text{Average time customer spends in the system} = 1.74 + 3.17 = 4.91 \text{ minutes}$$

A decision maker would be interested in results of this type, but a longer simulation would increase the accuracy of the findings. However, some tentative inferences can be drawn at this point. About half of the customers have to wait; however, the average waiting time is not excessive. The server does not have an undue amount of idle time. More reliable statements about the results would depend on balancing the cost of waiting against the cost of additional servers.

Excel spreadsheets have been constructed for each of the examples in this chapter. The spreadsheets can be found at www.bcnn.net. The spreadsheets have a common format. The first sheet is One-Trial. The second sheet is Experiment. The third sheet is entitled Explain. Here, the logic in the spreadsheet is discussed, and questions pertaining to that logic are asked of the reader. Use the default seed '12345' to reproduce the One-Trial output shown in the examples in the text, and use the appropriate number of trials (or replications) to reproduce the Experiment shown in the text, again using the default seed '12345'.

Exercises relating to the spreadsheets have been prepared also. These are the last set of exercises at the end of this chapter. The first set of exercises is for manual simulation.

The spreadsheets allow for many entities to flow through the system. (In Example 2.1, the entities are customers.) For instance, the spreadsheet for Example 2.1 has 100 customers going through the system, and the number of trials can vary from one to 400. Let's say that 200 trials are selected. Then, 200 trials of the simulation, each of 100 customers, will be conducted.