

SIMULATION MODELING WITH SIMIO: A WORKBOOK V4

Jeffrey Allen Joines

Stephen Dean Roberts

North Carolina State University

Raleigh, North Carolina

November 2015

4th Edition

Copyright 2015 by Jeffrey A. Joines and Stephen D. Roberts All rights reserved.

Published by:
SIMIO LLC
504 Beaver St, Sewickley, PA 15143, USA
<http://www.simio.com>

SIMIO is a trademark of SIMIO LLC.

Table of Contents

About the Authors	v
Preface to the Fourth Edition.....	vi
PART 1: ORGANIZATION OF THIS EDITION	VI
PART 2: SPECIFIC CHANGES IN THE FOURTH EDITION.....	VII
PART 3: STYLES USED IN THIS BOOK.....	VIII
PART 4: ACKNOWLEDGEMENTS	VIII
Chapter 1 Introduction to Simulation: The Ice Cream Store	1
PART 1.1: WHAT IS SIMULATION?	1
PART 1.2: SIMULATION FUNDAMENTALS: THE ICE CREAM STORE.....	3
PART 1.3: MANUAL SIMULATION	10
PART 1.4: INPUT MODELING AND SIMULATION OUTPUT ANALYSIS.....	14
PART 1.5: ELEMENTS OF THE SIMULATION STUDY.....	19
PART 1.6: COMMENTARY	20
Chapter 2 Introduction to SIMIO: The Ice Cream Store.....	21
PART 2.1: GETTING STARTED	21
PART 2.2: THE ICE CREAM STORE	23
PART 2.3: ENHANCING THE ANIMATION.....	26
PART 2.4: LOOKING AT THE RESULTS.....	29
PART 2.5: COMMENTARY	31
Chapter 3 Modeling Distance and Examining Inputs/Outputs.....	32
PART 3.1: BUILDING THE MODEL	32
PART 3.2: USING THE 3D WAREHOUSE.....	33
PART 3.3: EXAMINING MODEL INPUT PARAMETERS	34
PART 3.4: EXAMINING OUTPUT	35
PART 3.5: USING EXPERIMENTS.....	37
PART 3.6: INPUT SENSITIVITY	40
PART 3.7: COMMENTARY	41
Chapter 4 More Detailed Modeling: Airport Revisited	42
PART 4.1: CHOICE OF PATHS.....	42
PART 4.2: CHANGING ARRIVAL RATE	44
PART 4.3: STATE VARIABLES, PROPERTIES, AND DATA TABLES	46
PART 4.4: MORE ON BRANCHING.....	50
PART 4.5: WORK SCHEDULES	51
PART 4.6: COMMENTARY	53
Chapter 5 Data-Based Modeling: Manufacturing Cell.....	55
PART 5.1: CONSTRUCTING THE MODEL	56
PART 5.2: SETTING CAPACITIES.....	58
PART 5.3: INCORPORATING SEQUENCES.....	59
PART 5.4: EMBELLISHMENT: NEW ARRIVAL PATTERN AND PROCESSING TIMES	61
PART 5.5: USING RELATIONAL TABLES.....	64
PART 5.6: CREATING STATISTICS.....	68
PART 5.7: OBTAINING STATISTICS FOR ALL PART TYPES	70
PART 5.8: AUTOMATING THE CREATION OF ELEMENTS FOR STATISTICS COLLECTION.....	72
PART 5.9: COMMENTARY	76
Chapter 6 Assembly and Packaging: Memory Chip Boards.....	78
PART 6.1: MEMORY BOARD ASSEMBLY AND PACKING	78
PART 6.2: MAKING THE ANIMATION REVEAL MORE INFORMATION.....	84

PART 6.3: EMBELLISHMENT: OTHER RESOURCE NEEDS	86
PART 6.4: CHANGING PROCESSING TIME AS A FUNCTION OF THE SIZE OF THE QUEUE.....	88
PART 6.5: CREATING STATISTICS	90
PART 6.6: COMMENTARY	92
Chapter 7 Using SIMIO Processes	93
PART 7.1: THE ADD-ON PROCESS	93
PART 7.2: THE ADD-ON PROCESS TRIGGERS: ILLUSTRATING “ASSIGN” STEP	94
PART 7.3: CREATING AN INDEPENDENT “REUSABLE” PROCESS	96
PART 7.4: COLLECTING TALLY STATISTICS	97
PART 7.5: MANIPULATING RESOURCES.....	100
PART 7.6: TOKENIZED PROCESSES	102
PART 7.7: COMMENTARY	104
Chapter 8 Working with Flow and Capacity: The DMV	105
PART 8.1: THE DMV OFFICE	106
PART 8.2: USING RESOURCES WITH SERVERS	110
PART 8.3: HANDLING FAILURES.....	111
PART 8.4: SERVER CONFIGURATION ALTERNATIVES.....	114
PART 8.5: RESTRICTING ENTITY TO FLOW TO PARALLEL SERVERS.....	116
PART 8.6: THE WAITING ROOM SIZE.....	122
PART 8.7: USING APPOINTMENT SCHEDULES.....	124
PART 8.8: CONTROLLING THE SIMULATION REPLICATION LENGTH	130
PART 8.9: COMMENTARY	134
Chapter 9 The Workstation Concept: A Kitting Process.....	135
PART 9.1: THE KITTING PROCESS.....	135
PART 9.2: SEQUENCE-DEPENDENT SETUP TIMES.....	138
PART 9.3: SEQUENCE-DEPENDENT SETUP TIMES THAT ARE RANDOM.....	141
PART 9.4: USING MATERIALS IN THE KITTING OPERATION	143
PART 9.5: RAW MATERIAL ARRIVALS DURING THE SIMULATION	146
PART 9.6: IMPLEMENTING A JUST-IN-TIME APPROACH	148
PART 9.7: COMMENTARY	151
Chapter 10 Inventories, Supply Chains, and Optimization	152
PART 10.1: BUILDING A SIMPLE SUPPLY CHAIN	152
PART 10.2: PROCESSING ORDERS IN THE SUPPLY CHAIN SYSTEM	156
PART 10.3: CREATING THE REPLENISHMENT PART OF THE SUPPLY CHAIN SYSTEM	158
PART 10.4: USING AN EXPERIMENT TO DETERMINE THE BEST VALUES.....	160
PART 10.5: USING SMORE PLOTS TO DETERMINE THE BEST VALUES	161
PART 10.6: USING RANKING AND SELECTION TO DETERMINE THE BEST SCENARIO	163
PART 10.7: USING OPTQUEST™ TO OPTIMIZE THE PARAMETERS	165
PART 10.8: MULTI-OBJECTIVE AND ADDITIONAL CONSTRAINTS USING OPTQUEST™	167
PART 10.9: COMMENTARY	170
Chapter 11 Simulation Output Analysis.....	171
PART 11.1: WHAT CAN GO WRONG?	171
PART 11.2: TYPES OF SIMULATION ANALYSES	172
PART 11.3: OUTPUT ANALYSIS.....	172
PART 11.4: AUTOMATIC BATCHING OF OUTPUT	177
PART 11.5: ALGORITHMS USED IN SIMIO BATCH MEANS METHOD	179
PART 11.6: INPUT ANALYSIS	180
PART 11.7: COMMENTARY	185
Chapter 12 Materials Handling	186
PART 12.1: VEHICLES: CART TRANSFER IN MANUFACTURING CELL	186
PART 12.2: CART TRANSFER AMONG STATIONS	191

PART 12.3: OTHER VEHICLE TRAVEL BEHAVIORS (FIXED ROUTE AND FREE SPACE TRAVEL).....	195
PART 12.4: CONVEYORS: A TRANSFER LINE	197
PART 12.5: MACHINE FAILURES IN THE CELL	201
PART 12.6: SORTING CONVEYORS	202
PART 12.7: COMMENTARY	204
Chapter 13 Management of Resources: Veterinary Clinic.....	205
PART 13.1: UTILIZING THE FIXED RESOURCE OBJECT	205
PART 13.2: DIFFERENT RESOURCE NEEDS BASED ON DIFFERENT PATIENT TYPES.....	208
PART 13.3: RESOURCE DECISION MAKING	213
PART 13.4: ADDING AN ADDITIONAL PROCESS	215
PART 13.5: CHANGING PROCESSING BASED ON ANIMAL TYPE AND VET SERVICING	217
PART 13.6: CHANGING THE RESOURCE ALLOCATION SELECTION	223
PART 13.7: COMMENTARY	224
Chapter 14 A Mobile Resource: The Worker.....	226
PART 14.1: ROUTING PATIENTS	226
PART 14.2: USING A WORKER AS A MOVEABLE RESOURCE	231
PART 14.3: RETURNING TO THE OFFICE BETWEEN PATIENTS.....	235
PART 14.4: ZERO-TIME EVENTS	238
PART 14.5: HANDLING MULTIPLE VETS	240
PART 14.6: COMMENTARY	242
Chapter 15 Adding Detail to Service: A Bank Example.....	244
PART 15.1: USING A WORKER AS A RESOURCE AND A VEHICLE.....	244
PART 15.2: HAVING THE BANKER ESCORT THE CUSTOMER TO THE DEPOSIT BOX	248
PART 15.3: USING THE TRANSPORT FUNCTION OF THE WORKER	253
PART 15.4: RESOURCE RESERVATIONS.....	256
PART 15.5: ANIMATED ENTITIES	257
PART 15.6: DETAILED SERVICE: TASKS AND TASK SEQUENCES	260
PART 15.7: USING TASK SEQUENCES	262
PART 15.8: SOME OBSERVATIONS CONCERNING TASKS	264
PART 15.9: COMMENTARY	269
Chapter 16 Modeling of Call Centers.....	270
PART 16.1: BUILDING THE SIMPLE MODEL.....	271
PART 16.2: BALKING.....	274
PART 16.3: MODELING RENEGING OF CUSTOMER CALLS	276
PART 16.4: OPTIMIZING THE NUMBER OF FIRST LINE TECHNICIANS	279
PART 16.5: USING THE FINANCIAL COSTS AS THE OPTIMIZING OBJECTIVE.....	283
PART 16.6: COMMENTARY	287
Chapter 17 Sub-Modeling: Cellular Manufacturing	289
PART 17.1: MODEL OF ONE WORK CELL.....	289
PART 17.2: CREATING THE SUB-MODEL.....	291
PART 17.3: CREATING A MODEL USING THE WORKCELL SUB-MODEL	293
PART 17.4: ADDING TO THE WORKCELL OBJECT.....	294
PART 17.5: EXPOSING RESOURCE AND CAPACITY PROPERTIES	296
PART 17.6: PASSING INFORMATION BETWEEN THE MODEL AND ITS SUB-MODELS.....	298
PART 17.7: COMMENTARY	300
Chapter 18 The Anatomy of Objects: Server	301
PART 18.1: A SIMPLE RESOURCE MODEL: WAREHOUSE PICKUP	301
PART 18.2: TAKING AN OBJECT APART TO FIGURE OUT HOW IT WORKS.....	303
PART 18.3: SIMIO OBJECTS AND CLASS HEIRARCHY	308
Chapter 19 Building New Objects via Sub-Classing: A Delay Object.....	310
PART 19.1: SUB-CLASSING THE TRANSFERNODE TO CREATE A DELAYNODE	310

PART 19.2: MODIFYING PROCESSES AND ADDING A PROPERTIES FOR THE NEW NODE.....	312
PART 19.3: CREATING A MODEL TO TEST THE NEW DELAYTRANSFERNODE.....	314
PART 19.4: COMMENTARY	318
Chapter 20 Creating New Objects.....	319
PART 20.1: CREATING A SIMPLE DELAY OBJECT.....	319
PART 20.2: ADDING COLOR AND STATES.....	325
PART 20.3: ADDING DEFINED ADD-ON PROCESS TRIGGERS.....	327
PART 20.4: EMBELLISHING WITH STATE ASSIGNMENTS	330
PART 20.5: CAPACITY AND ITS COMPLICATIONS	334
PART 20.6: ADDING MULTIPLE RESOURCES	336
PART 20.7: USING STORAGES TO DISTINGUISH WAITING VERSUS DELAYING	338
PART 20.8: SOME OBSERVATIONS ON THE DESIGN OF OBJECTS.....	341
PART 20.9: COMMENTARY	341
Chapter 21 Continuous Variables, Reneging, Interrupt, Debugging: A Gas Station.....	343
PART 21.1: SIMPLE TANK MANUAL PROCESS	343
PART 21.2: SIMPLE TANK REVISITED USING THE FLOW LIBRARY	347
PART 21.3: THE GAS STATION.....	350
PART 21.4: RENEGING THE CARS WHEN THE PUMP GOES OFF.....	356
PART 21.5: INTERRUPTING THE CARS WHEN PUMP GOES OFF	358
PART 21.6: USING ENTITIES THAT CARRY A TANK.....	359
PART 21.7: COMMENTARY	363
Chapter 22 More Subclassing: Advanced Modeling of Supply Chain Systems	364
PART 22.1: DEVELOPING A SPECIALIZED SUPPLY CHAIN WORKSTATION OBJECT.....	364
PART 22.2: ADDING THE ORDERING STATION AND CHARACTERISTICS TO HANDLE ORDERS	366
PART 22.3: ADDING THE BEHAVIOR LOGIC FOR THE ORDERING SYSTEM.....	370
PART 22.4: ADDING THE BEHAVIOR LOGIC FOR THE INVENTORY REPLENISHMENT SYSTEM.....	373
PART 22.5: USING THE NEW TO MODEL THE COMPLEX SUPPLY SYSTEM	376
PART 22.6: ADDING A SECONDARY SUPPLIER FOR OVERFLOW ORDERS.....	379
PART 22.7: COMMENTARY	382
Chapter 23 More Subclassing: Process Planning/Project Management.....	383
PART 23.1: PROCESS PLANNING	383
PART 23.2: CREATING A SPECIALIZED TIMEPATH TO HANDLE ACTIVITIES	384
PART 23.3: CREATING A JUNCTION OBJECT TO HANDLE PRECEDENT CONSTRAINTS	386
PART 23.4: CREATING SMALL NETWORK TO TEST THE NEW OBJECT	392
PART 23.5: BUILDING THE EXAMPLE NETWORK.....	394
PART 23.6: ADDING THE SLACK AND PERCENT OF TIME ON CRITICAL PATH CALCULATIONS.....	395
PART 23.7: ADDING SLACK AND PERCENT OF TIME ON CP CALCULATIONS SECOND APPROACH	398
PART 23.8: ERROR CHECKING TO MAKE SURE MODELER USES JUNCTION CORRECTLY	400
PART 23.9: COMMENTARY	402
Appendix A Input Modeling	403
PART A.1 RANDOM VARIABLES	403
PART A.2 COLLECTING DATA	403
PART A.3 INPUT MODELING: SELECTING A DISTRIBUTION TO FIT TO YOUR DATA	404
PART A.4 DISTRIBUTION SELECTION HIERARCHY	409
PART A.5 EMPIRICAL DISTRIBUTIONS IN SIMIO	409
PART A.6 SOFTWARE FOR INPUT MODELING	411
PART A.7 THE LOGNORMAL DISTRIBUTION.....	412
PART A.8 MODELING THE SUM OF <i>N</i> INDEPENDENT RANDOM VARIABLES.....	413
Index.....	416

Chapter 1

Introduction to Simulation: The Ice Cream Store

This chapter will give the novice in simulation an introduction to the terminology and mechanics of simulation, as well as statistics used in performing a simulation model. If you are already familiar with these basic simulation concepts, feel free to start with Chapter 2 which starts the simulation modeling in SIMIO.

Part 1.1: What is Simulation?

The word “simulation” has a variety of meanings and uses. Probably its most popular usage is in games. A video game “simulates” a particular environment, maybe a formula one race, a battle, or a space encounter. The game allows the user to experience something similar to what it is like to drive in a race, maybe with other people participating. In the military, commanders create a battlefield simulation where soldiers act according to their training perhaps defending a base or assaulting an enemy position. In aeronautical engineering, an engineer may take a model airplane to a wind tunnel and test its aerodynamics. Most simulations have elements of reality with the intention that the participant will learn something about the environment through the simulation or perhaps only learning to play the game better.

We employ simulation to study and improve “systems” of people, equipment, materials, and procedures. We use simulation to mimic or imitate the behavior of systems like factories, warehouses, hospitals, banks, supermarkets, theme parks – just about anywhere a service is provided or an item is being produced. Our simulations are different from gaming and training simulations in that we want the simulation to model the real system, so we can investigate various changes before making recommendations. In that sense, simulation is a *performance improvement tool*. The simulation acts as an experimental laboratory, except that our laboratory is not physical but instead a computer model. We can then perform experiments on our computer model.

Modeling

Many performance improvement tools (i.e., Lean, Six Sigma, etc.) rely on models. For example value stream maps, spaghetti diagrams, process flow charts, waste walks, etc. are useful conceptual/descriptive models for performance improvement based on direct observations of the system. These models provide a wide-range of vehicles for describing and analyzing various systems. More formal models employ mathematical and statistical methods. For example, linear regression is a popular modeling technique in statistics. Queuing models offer a means for describing an important group of stable stochastic processes. Linear programming is a formal optimization method of finding the values of variables that minimize or maximize a linear objective function subject to linear constraints. Nevertheless, all these methods require a variety of assumptions about the system being modeled. For example, the variables are constants (i.e., real or integer valued) and often related linearly. If the variables have statistical variation, they are assumed to be normally or exponentially (i.e., Markovian) distributed.

Simulation is a *model-based* improvement tool. However, few assumptions need to be made to build the model. The model can be non-linear, described by arbitrary random variables, have a complex relationship, and change with time (i.e., dynamic). In fact, the simulation model is limited only by your imagination and the nature of the system being considered. You determine the nature of the model, based on what you think is important about the system being studied.

Computer Simulation Modeling

Instead of a formal model, our simulations are *computational*. The model is essentially a logical description of how the components of a system interact. This description is translated into a computational structure within the computer using a simulation language. The computer simulation of the system is executed over and over to generate statistical information about the system behavior. We use the statistics to describe the system

performance measures. Based on what we learn about the system, we modify the computer simulation models to study alternative systems (i.e., experiment). By comparing these alternative systems statistically, we are able to offer performance improvement recommendations.

Of course, we could just experiment directly with the system. If we thought an additional person on the assembly line would improve its production, we could try that. If we thought a new configuration of the hospital emergency room would provide more efficient care, we could create that configuration. If we thought a new inventory policy would reduce inventory, we could implement it. But now the value of a computer simulation model becomes apparent. A change in a computer simulation model is clearly cheaper and less risky as compared to changing the real system. It is also faster to use a computer simulation to determine if the changes are beneficial. Also, it might be safer to try a change in a simulation model than to try it in real life. In general, it is much easier to try changes in a computer model than in a real operating system, especially since changes disrupt people and facilities.

A computer simulation model allows us to develop confidence in making performance improvement recommendations. We can try out a wide range of alternatives before disrupting an existing system. You can now see why many companies and many managers require that a simulation study be done before making substantial changes to any working system, especially when there are potential negative consequences of performance changes as well as costs that don't provide improvement.

Verification and Validation

Since simulation models often have serious consequence and are developed with a minimum set of assumptions, the validity of a simulation model needs to be carefully considered before we believe in the recommended benefits. In simulation, we often use the words “*verification*” and “*validation*” and we have specific definitions for each as they relate to simulations.

The word “*verification*” refers to the model and its behavior. We most often develop our simulation models with a simulation language. This language translates our modeling “intent” into a computation structure that produces output statistics. Most simulation languages can be used to describe a complex operating system and the language provides a framework of components for viewing that system. As our models become more complex, we employ more complex simulation language constructs to model the behavior. This relationship is critically dependent on our understanding of the simulation language intricacies. We may or may not fully understand computational code and thus we confront the key question in verification: *Does the simulation model behave the way we expect?* Is it possible that we may have made a mistake in employing the simulation language or perhaps the simulation language creates an unexpected behavior in the computational code? Therefore, even after we have created our simulation model, it needs to be tested to see if it behaves as expected. If we make the processing time longer, does it result in longer times in the system? If we reduce the arrival rate to a queue, does the waiting time decrease? The answers to any of these questions are specific to the system and your model of it. But, above all else, we need to be sure that our model is behaving the way we expect without error.

The word “*validation*” refers to the relationship between the model and the real system. *Does the simulation produce performance measures consistent with the real system?* Now, we are assuming our model has been verified but can we validly infer to the real system? This question strikes at the heart of our modeling effort because without a valid model we cannot legitimately say much about performance improvement. Many people new to simulation may want the model to be a substitute for the “real system” and that is a limitless task. After all, the only “model” that is perfectly representative of the real system is the system itself!

We must always remember in simulation that our model is only an *approximation* of the real system. So the most relevant way to validate our model is to concern ourselves with its approximation. How do we decide what to approximate? It depends on our performance measures. If our performance measure is time in the system, then we concern ourselves with those factors that impact time in system. If our performance measure is production, then we focus on those factors that impact production. Usually we are interested in several

performance measures, but those measures will be the focus of our concern and will limit our modeling activity. Otherwise without a clear set of performance objectives, we are left with a search for reality in our model, which is a never-ending task.

Generally, people who have only a general familiarity with simulation, think simulation models can mimic anything and often drive the development of a model that is needlessly complex. One of the difficult responsibilities of anyone engaged in simulation modeling is the education of the stakeholders on the benefits and limitations of a simulation model as well as the simulation modeling activity.

Computer Simulation Languages

The process of creating a computer simulation model varies from programming your own model in a programming language such as C++, C#, or Java to using a spreadsheet like Excel. There are a variety of simulation languages including Simula™, GPSS™, SIMSCRIPT™, Arena/SIMAN™, SLX™, ProModel™, Flexsim™, AutoMod™, ExtendSim™, Witness™, AnyLogic,™ among many others. Each of these simulation languages differ in the way they require users to construct a simulation model. An important distinction is the degree to which computer programming is required. Also many of the simulation languages have evolved from a particular industry or set of applications and are especially useful in that context.

We have chosen to use the simulation language SIMIO which is a relatively new language, having been developed over the past several years. The developers of the language had previously developed the Arena simulation system. SIMIO benefits from more recent developments in object-oriented design and agent modeling. SIMIO is a “multi-modeling” language having agents as well as discrete event and continuous language components. SIMIO was developed to provide visual appeal through its 3-D animation and graphical representations. SIMIO provides a wide range of extensions, from direct modification of executing processes to user developed objects. SIMIO also provides interoperation with various spreadsheets and databases. Finally, SIMIO has become widely adopted in industry as well as academic institutions. Learning SIMIO provides you with one broadly-based simulation modeling tool which can help you learn to use others if the need arises.

Part 1.2: Simulation Fundamentals: The Ice Cream Store

A fundamental understanding of simulation will be beneficial throughout your study of simulation. It's easy to get caught up in the creation of a computer model using a simulation language and miss important basic principles of simulation modeling. Too many people associate simulation with a simulation language and for them, simulation is simply learning a simulation language. Learning a simulation language is necessary to using simulation, but it does not substitute for understanding at a fundamental level. If you understand the fundamentals of simulation, then you establish a basis for understanding any simulation language and any simulation model. In fact, this understanding will be a key to learning almost everything else in this text.

The Ice Cream Store

It is helpful if the discussion of simulation is done in the context of a problem – albeit a simple problem. Using this simple problem, we will be able to describe simulation elements of modeling, execution, and analysis. Our problem is the common ice cream store since everyone loves ice cream and probably has visited an ice cream store. As seen in Figure 1.1, customers arrive to the ice cream store where they obtain an ice cream cone. It's a simple store where there is only one attendant and people will wait in a single line to order and receive their ice cream cone. The attendant waits on each customer, one at a time in the order they arrive.

Question 1: If you owned or managed the ice cream store, what might be your operational concerns?

Likely one of your most prominent concerns would be this store's operation and how you can improve it. For example, could you buy a new cone making machine to make cones faster or should you hire someone to help service customers? Should you resize the waiting space? These are performance improvement concerns.

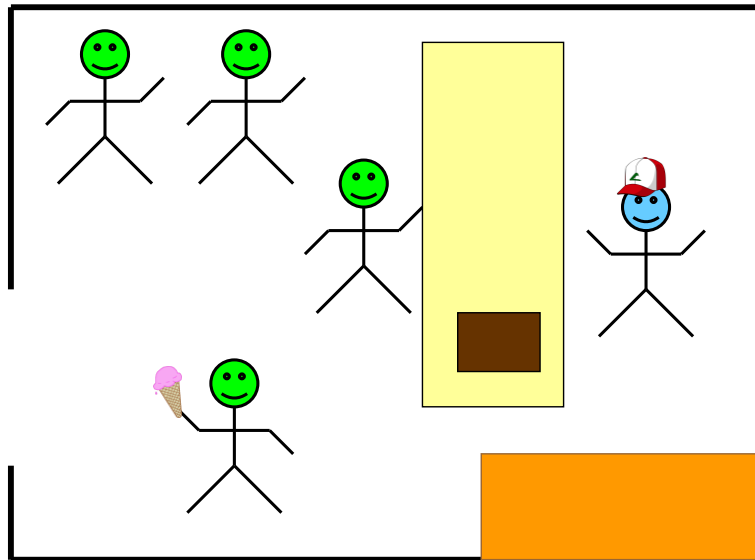


Figure 1.1: The Ice Cream Store

Question 2: If you made one of these changes to the ice cream store, how would you decide if it improved the store's operation?

In other words, what are the performance measures? Here are some possible measures: number of customers served per day, time customers spent in the store, the waiting time of customers, the number of customer waiting, and the utilization of servers. We will discover these are common performance measures and will often be your first performance measurement choices.

Question 3: How can you expect to improve performance if you don't know what is going on inside the ice cream store?

You need to employ all the descriptive tools you know to understand what is going on inside the system. At this stage, one might create a value stream map, a flow chart, a relationship chart, a spaghetti diagram, etc. These techniques will greatly improve your understanding of what happens to customers and the attendant during the sale of an ice cream cone. Perhaps you develop the conceptual model (i.e., flow chart) presented in Figure 1.2.

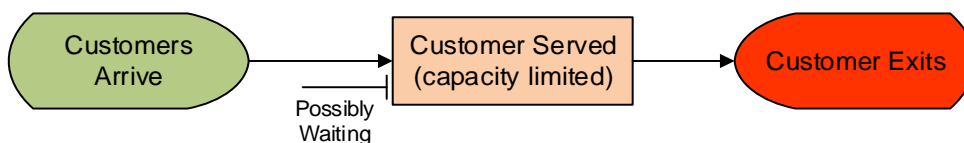


Figure 1.2: Flow Chart of the Ice Cream Store

The conceptual model shows how the customers are served. Notice, we have added the possibility of waiting due to the fact that service is limited by the availability of the single attendant.

Gathering Data about the Ice Cream Store

Descriptive information helps us understand the service process in the ice cream store; however it doesn't give us the performance measures. For that we need to document what is happening (i.e., we need to do a "*present systems analysis*"). Suppose we decide to do a "time study" of the store operations which is shown in Table 1.1.

Table 1.1: Direct Observation Time Study (Event View) of Ice Cream Store

Time	Customer	Process
0		Store Opens – Server Idle
0	1	Arrives –
		Start service on Customer 1 – Server Busy
8.36	1	Departs service – Server Idle
9.01	2	Arrives –
		Start service on Customer 2 – Server Busy
9.98	3	Arrives – Customer 3 waits
13.5	4	Arrives – Customer 4 waits
19.36	2	Departs Service – Customer 2 leaves:
	3	Start service on Customer 3 – Server Busy
23.07	5	Arrives – Customer 5 waits
27.22	3	Departs service – Customer 3 leaves:
	4	Start service on Customer 4 – Server Busy
33.82	4	Departs service – Customer 4 leaves:
	5	Start service on Customer 5 – Server Busy
38.18	6	Arrives – Customer 6 waits
40		End observations

In our time study, we simply record all the “events” and the event time that occurs in the ice cream store. An event occurs when something operationally happens in the store, like an arrival of a customer or the departure of a customer from service. Also we recorded when the attendant (i.e., server) becomes busy or idle. Here we only observed the first 40 minutes of the store’s operation. We recognize this isn’t really enough time to gain a full understanding, but we are not intending to solve a problem at this time – only to demonstrate a method.

The time study is sufficient for us to compute some performance measures, but first it will be helpful to re-organize our time study data. We are not going to add (or subtract) any information. We are simply going to re-organize our data from an *event view* to an *entity* (i.e., customer) *view*. The event view recorded events but the entity view allows us to follow our customers. The re-organized data is shown in Table 1.2 which shows when the customer arrives to the store, enters service, and leaves the store. Notice, customers five and six service and exiting store happen after the 40 minutes.

Table 1.2: Re-organized Event Data in Customer View From

Customer	Arrives to Store	Enters Service	Leaves Store
1	0.00	0.00	8.36
2	9.01	9.01	19.36
3	9.98	19.36	27.22
4	13.50	27.22	33.82
5	23.07	33.82	??
6	38.18	??	??

Performance Measure Calculations

From the entity view of the time study data, we can easily compute a number of performance measures.

- **Production:** Number of people served
The number of people served is easily calculated by looking at the number of customers that have exited the system in the simulation time. In our example, four people were served in 40 minutes or the system had a production rate of six per hour (i.e., 4 customers/40 minutes * 60minutes/hour).
- **Flowtime, Cycle Time:** Time in System

The time in system is calculated by averaging the difference when the customer entered the system from when they exited the system. Again, only four customers exited the system and contributed to the average time in system in our simple ice cream system as seen in Table 1.3.

Table 1.3: Calculating Average Time in System for the Ice Cream Store

Customer 1	Customer 2	Customer 3	Customer 4	Average Time in System
8.36-0.0	19.36-9.01	27.22-9.98	33.82-13.5	$56.27/4 = 14.07$ minutes

- **Non-Value Added Time: Waiting Time in Queue**

The time in queue (i.e., the waiting or holding time) represents the time customers waited in line before being seen by the ice cream attendant. It is calculated by averaging the difference when the customer entered the system from when they entered service. During our observation window, five customers entered service as seen in Table 1.4.

Table 1.4: Calculating Average Waiting Time in the Queue

Customer 1	Customer 2	Customer 3	Customer 4	Customer 5	Average Waiting Time
0.0-0.0	9.01-9.01	19.36-9.98	27.22-13.50	33.82-23.07	$33.85/5 = 6.77$ minutes

- **Number Waiting on In Queue:**

This measure determines the average number of customers one would expect to see waiting in line to receive service. Unlike the previous metrics, number waiting in queue or number in the system measures are *time-persistent statistics* or *time-weighted statistics*. You may have never computed a time-persistent statistic which is a statistic for which we need to know the amount of time a value was observed. We weight the value of the observation by the amount of time that value persists. A graphical depiction of the number in the queue at the ice cream store is in Figure 1.3.

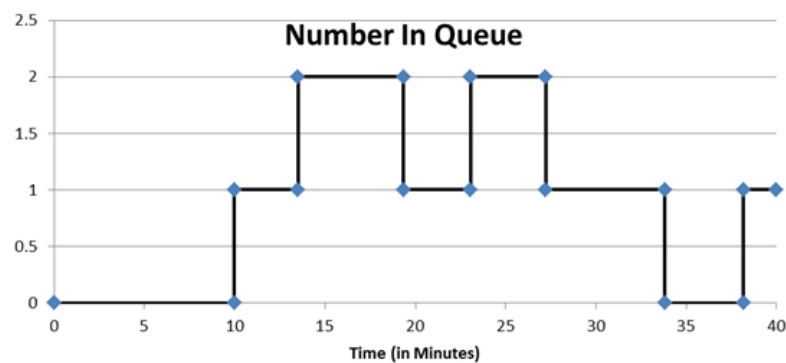


Figure 1.3: Graph of the Number in Queue in the Ice Cream Store

Consider, computing the average number in queue. Suppose we observed the number in queue to be two one time and ten another time. Would you say that on average number we would expect to see six in line? Of course “not”! You need to know how long the value of two was observed and how long the value of ten was observed. Suppose we observe two in queue for ten minutes of the time and ten in the queue for only one minute. So the queue was observed a total of 11 minutes. As a result, the value of two was observed 10/11ths of the total time while ten was observed 1/11th of the total time. Our average then is $2 \cdot (10/11) + 10 \cdot (1/11)$ or $= 2.73$ customers. Or another way of computing is to realize that the total waiting time² observed was $20 + 10$ over a total of 11 minutes which also yields 2.73 people. Now, looking at the data from the ice cream store, we need the percentage of time (40 minutes) that there was zero waiting, one waiting, two waiting, three and so forth³. In our case, a maximum of two was observed and the time waiting would be computed as seen in Table 1.5.

² Total waiting time is also the “area” under the curve in Figure 1.3.

³ We don’t include 0 since it makes no contribution to the total waiting time.

Table 1.5: Calculating Average Number in Queue

Number In Queue	Time Period	Time Spent (min)
0	0.00 – 9.98	$0 \times 9.98 = 0.000$
1	9.98 – 13.5	$1 \times 3.52 = 3.520$
2	13.50 – 19.36	$2 \times 5.86 = 11.72$
1	19.36 – 23.07	$1 \times 3.71 = 3.710$
2	23.07 – 27.22	$2 \times 4.15 = 8.300$
1	27.22 – 33.82	$1 \times 6.60 = 6.600$
0	33.82 – 38.18	$0 \times 4.36 = 0.000$
1	38.18 – 40.00	$1 \times 1.82 = 1.820$
Total Waiting Time		35.67 minutes
Average Number in Queue		$35.67/40 = 0.89$ Customers

- **Utilization:**
This performance is the percentage time the server is busy servicing customers which is calculated by dividing the time spent servicing customers divided by the time available. For our example, the attendant is only idle during the time period between finish servicing customer one and the arrival of customer two (i.e., $9.01 - 8.36 = .65$ minutes). Therefore, the attendant's utilization will be $39.35/40$ or 98.4%:
- Other possible performance measures are maximum values, standard deviations, time between departures, etc.

Table 1.6: Types of Performance Simulation Measures/Metrics

Type	Description/Examples
Counts	The number of parts that exited, entered, etc. (e.g., production).
Observation-based Statistics	These measures typically deal with time like waiting time, time in system etc. SIMIO calls observation-based statistics “Tally” statistics.
Time-Persistence/Time-Average Statistics	These measures deal with numbers in system, queue, etc. when the values can be classified into different states. SIMIO calls these “State” statistics.

Question 4: What is another example of an observations-based performance measure?

Question 5: What is another example of a time-persistence performance measure?

Question 6: Is “inventory” a time-persistent or observation-based statistics?

Question 7: Is the amount of time that a job is late a time-persistent or observation-based statistic?

Terminology of a Queuing System

Figure 1.4 shows some elements of a “common” queueing system with the normal terminology. In general we will refer to the arriving objects as “model entities” and the counter/attendant as a “server”. You can see the members of the queue and the customer in service. The input processes for this system are the arrival process and service processes. We will provide details on these inputs later.

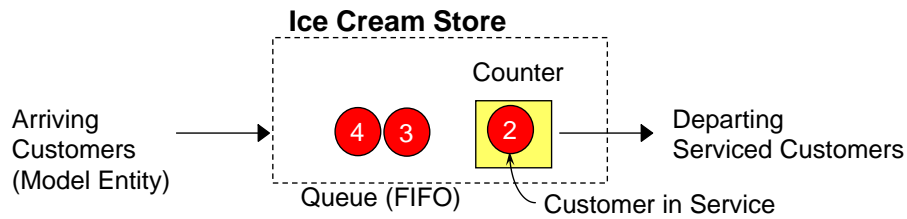


Figure 1.4: A Breakdown of Terminology

Can we simulate it?

We want to be able to reproduce the time-study data collection exercise that we used for the ice cream store. But we want to synthesize it numerically as opposed to observing it – in essence simulate it! If we can recreate the time study data, we can compute the performance measures. Notice that the time study records are centered on “**events**”. Recall these events are points in time when the system changes its state (i.e., status). A quick review of that data reveals the following three events occur.

Table 1.7: Events of the Ice Cream Store

Number	Type	Event Description
1	Arrival	Indicates an entity will arrive at this time.
2	End of Service	An entity will be departing from service (i.e., service has finished)
3	End	The simulation will terminate and end of all observations.

To facilitate our simulation we need a method to keep track of our events. Although there are other ways to keep track of events, it is convenient to keep them in an “**event calendar**”. An event calendar contains the records of future events (i.e., things we think are going to happen in the future), ordered by time (with the earliest event first). Simulation can be executed by removing and inserting events into the event calendar.

Let’s add some specifics to our simulation problem (i.e., the ice cream store). We will use “minutes” as our base measure of time. The input data that we currently have is shown in Table 1.8.

Table 1.8: Input Data

Customer	Arrival Time	Interarrival Time	Processing Time
1	0.00	0.00	8.36
2	9.01	9.01	10.35
3	9.98	0.97	7.86
4	13.50	3.52	6.60
5	23.07	9.57	8.63
6	38.18	15.11	10.33
7	42.08	3.90	10.46
8	48.80	6.72	7.96

In the table, the arrival time has been re-stated as an “interarrival time” – namely time between arrivals. Doing this re-statement does not change the arrival times but simply changes how they are presented. Such a representation also requires the time of the first arrival from which the interarrival times are sequentially computed. Table 1.8 provides data on only the first customers. We may need more data, but this is a later discussion.

The Simulation Algorithm

To synthesize the system as presented in the time study, we need a systematic means of moving through time by removing and inserting events on the event calendar. Consider the simulation algorithm shown in Figure 1.5.

- Algorithm Step 1:* Setup the simulation (i.e., initialize the system)
Algorithm Step 2: Remove the next event from the event calendar
Algorithm Step 3: Update simulation time (i.e., *TimeNow*) to the time of that event
Algorithm Step 4: Execute the processes associated with the event (i.e., adding additional events as needed and collecting statistics)
 Arrival Event Process (see Figure 1.6)
 Departure Event Process (see Figure 1.7)
 End Event
Algorithm Step 5: Repeat *Algorithm Steps 2-4* until complete

Figure 1.5: Simple Simulation Algorithm

Remember that the event calendar is ordered according to the next most recent event. So we can move through time by removing the “next” event from the event calendar, updating time to the time of that event, and executing whatever processes are associated with that event. This simple method is then repeated until we reach some terminating time or condition.⁴ In our case, the “events” are the arrival of an entity, the service (departure) of the entity, and the end of the simulation period. The arrival and departure of the entities are the most important. Consider now how the arrival (see Figure 1.6) and the departure (see Figure 1.7) are processed within the context of our simple single queue, single server system. The word “schedule” means to insert this event into the event calendar.

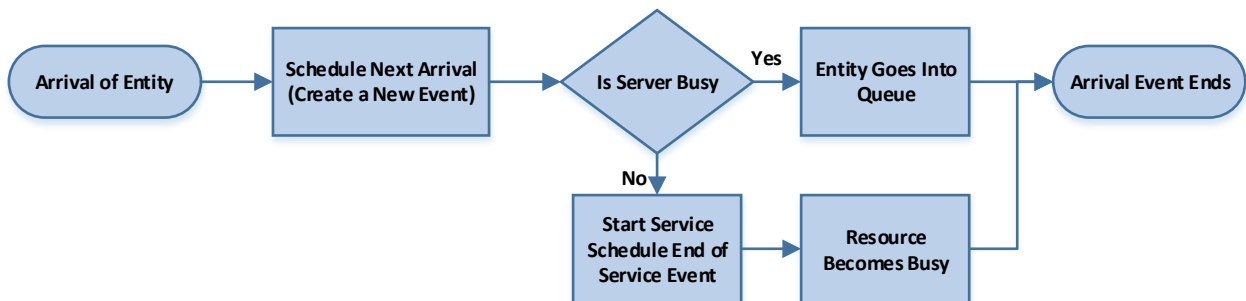


Figure 1.6: Event Process Associated with Arrival of Entity

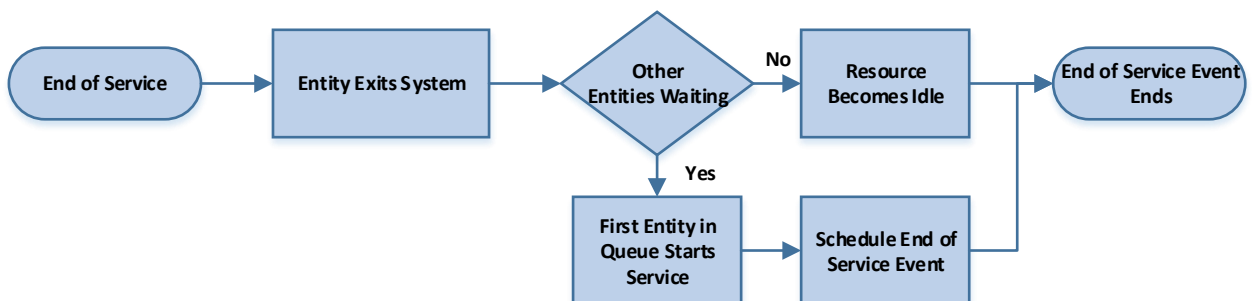


Figure 1.7: Event Process Associated with End of Service

The arrival of an entity creates a new event because we know the next entity’s arrival time since we have the interarrival times of entities. The newly arriving entity either must wait because the server (i.e., resource) is busy or that entity can engage the server and start service. If the entity can start service, we now know a new future event, namely the service departure because we know the processing time. So if the removed event is an arrival, we insert a next arrival into the event calendar and may insert the service departure event in the event calendar provided the entity can start service immediately. If the event removed from the event calendar is a service departure, then the entity that has finished service will exit the system. If the waiting queue is empty, then the resource (server) becomes idle. On the other hand, if there is at least one customer in queue,

⁴ The practice of removing the “next” event has caused some to refer to our simulation as a “next or discrete event simulation”.

then the first customer is brought into service and the resource remains busy and a new future event, namely the service departure is inserted into the event calendar because we have the processing time. Note that a service departure causes the entity to depart the system regardless of what happens to the server.

Question 8: What is the maximum number of new events that are added to the event calendar when an entity arrival event occurs? What are they?

Question 9: What is the minimum number of new events that are added to the event calendar when an entity arrival event occurs?

Question 10: What is the maximum number of new events that are added to the event calendar when an entity departure event occurs? What are they?

Question 11: What is the minimum number of new events that are added to the event calendar when an entity departure event occurs?

Finally, we note that the very first step of the simulation algorithm calls for the system to be “initialized.” In other words, how will we start operation relative to the number of entities in line and the state of the server? It will be convenient to use the “empty and idle” configuration. By “empty and idle” we mean that the server is idle and the system is empty of all entities.

Part 1.3: Manual Simulation

To further understand how a discrete event simulation operates, a simple data structure will be employed to execute a manual simulation as seen in Figure 1.8. Our manual simulation will consist of a “system animation” graphical representation of what is going on in the system with the current customer being serviced shown inside the square and other customers waiting outside the square. The current simulation time (i.e., called “Time Now”) will be shown. The current event will be identified, followed by a description of the process. Finally, the “Event Calendar” will be maintained, consisting of the event time and typ and the entity ID number.

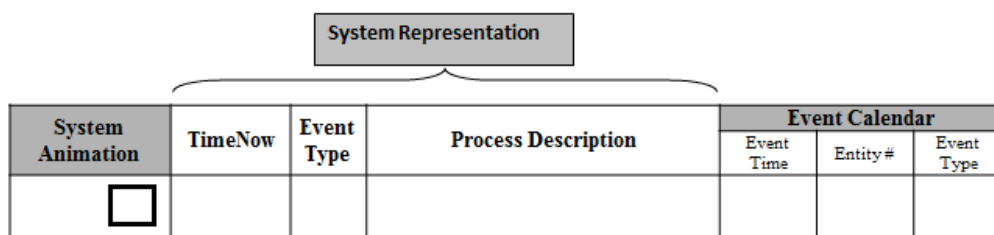


Figure 1.8: Manual Simulation Data Structure

Step 1: Algorithm Step 1 of the simulation as defined in Figure 1.5 is used to “initialize” the system as seen in executing structure of Figure 1.9. Note, that we have inserted into the event calendar the time of the first entity arrival and we have inserted the end event or the time to stop making observations.


System Animation	Time Now	Event Type	Process Description	Event Calendar		
				Event Time	Entity #	Event Type
	0.0		Initialize simulation	0.0 40.0	1	Arr End

Figure 1.9: System Initialization

Step 2: Executing the simulation algorithm, we remove the next event (Algorithm Step 2) from the event calendar, update simulation time to the time of that event (Algorithm Step 3), and execute the appropriate processes (Algorithm Step 4). The next event is the “Arrival of Entity #1” at time 0.0. The arrival of this entity allows us to schedule into the *Event Calendar* the “Arrival of Entity #2” to occur since the interarrival time between Entity #1 and Entity #2 is 9.01 minutes and thus the *Arrival* event for Entity #2 is at time 9.01 (i.e., $0.0 + 9.01$). Because Entity #1 arrives when the server is idle, that entity enters service and we can now schedule its *Departure* event because we know the processing time for Entity #1 as 8.36 minutes thus the event time is 8.36 (i.e., $0.0 + 8.36$). The result is that our data structure now appears as Figure 1.10.


System Animation	Time Now	Event Type	Process Description	Event Calendar		
				Event Time	Entity #	Event Type
	0.0	Arr	Store opens and Entity 1 arrives and goes into service	8.36 9.01 40.0	1 2	Dep Arr End

Figure 1.10: Processing Time 0.0 Event

Question 12: We inserted the arrival of Entity #2 event before the departure of the Entity #1 event, so why is the departure event ahead of the arrival event in the event calendar?

Question 13: Why do we compute the new event times as $0.0 + ?$ What does the 0.0 mean?

Step 3: We are done with the processes associated with time 0.0. Now, the next event in the event calendar is the “Departure of Entity #1” from the system at time 8.36. Since there are no entities in the queue, the server is allowed to become idle and no new events are added to the event calendar as seen in Figure 1.11.


System Animation	Time Now	Event Type	Process Description	Event Calendar		
				Event Time	Entity #	Event Type
	8.36	Dep	Entity 1 departs service and server become idle	9.01 40.0	2	Arr End

Figure 1.11: At Time 8.36

Step 4: The next event is the arrival of Entity #2 at time 9.01. So we remove it from the event calendar and update the simulation time to 9.01. We can schedule the arrival of Entity #3 at the current time (9.01) plus the interarrival time from Entity #2 to Entity #3 (0.97) which is event time 9.98. Next, Entity #2 can start service immediately since the server is idle, so its departure event can be scheduled as the current time (9.01) plus the processing time for Entity #2 of (10.35), which yields an event time of 19.36. The new status of the simulation is shown in Figure 1.12. We also now have added animation of the status of the entities and the server.


System Animation	Time Now	Event Type	Process Description	Event Calendar		
				Event Time	Entity #	Event Type
	9.01	Arr	Entity 2 arrives and enters service	9.98 19.36 40.0	3 2	Arr Dep End

Figure 1.12: At Time 9.01

Step 5: The next event occurs at time 9.98 which is the arrival of Entity #3. Its arrival allows us to schedule the next arrival (i.e., Entity #4) at time 13.50. But now the arriving entity must wait which is indicated in the “System Animation” section of the data structure in Figure 1.13.


System Animation	Time Now	Event Type	Process Description	Event Calendar		
				Event Time	Entity #	Event Type
	9.98	Arr	Entity 3 arrives and must wait	13.5 19.36 40.0	4 2	Arr Dep End

Figure 1.13: At Time 9.98

Question 14: How did we get 13.50 for the arrival of Entity #4?

Step 6: The next event is the arrival of Entity #4, which will schedule the arrival of Entity #5, but will have no other actions since the server remains busy. The new simulation time is 13.50 and the updated status is shown in Figure 1.14


System Animation	Time Now	Event Type	Process Description	Event Calendar		
				Event Time	Entity #	Event Type
	13.50	Arr	Entity 4 arrives and must wait	19.36 23.07 40.0	2 5	Dep Arr End

Figure 1.14: At time 13.50

Step 7: Finally, we see that Entity #2 finishes service at 19.36 and departs. Entity #3 can go into service and we can schedule the service departure of Entity #3 at 27.22 as shown in Figure 1.15


System Animation	Time Now	Event Type	Process Description	Event Calendar		
				Event Time	Entity #	Event Type
	19.36	Dep	Entity 2 departs and Entity 3 goes into service	23.07 27.22 40.0	5 3	Arr Dep End

Figure 1.15: At Time 19.36

Question 15: How did we compute the service departure for Entity #3 to be 27.22?

Question 16: What is the next event?

Question 17: What new events are added as a result of this event? (Give the event time, the entity, and the event type?)

Step 8: From Figure 1.15, we see the next event is the arrival of Entity #5 at time 23.07 which triggers the addition of the next arrival (Entity #6).


System Animation	Time Now	Event Type	Process Description	Event Calendar		
				Event Time	Entity #	Event Type
	23.07	Arr	Entity 5 arrives and must wait	27.22	3	Dep
				38.18	6	Arr
				40.0		End

Figure 1.16: At Time 23.07

Step 9: Entity #3 completes service at 27.22 which allows Entity #4 to enter serviced which schedules the end of service event for Entity #4 at 33.82 as shown in Figure 1.17.


System Animation	Time Now	Event Type	Process Description	Event Calendar		
				Event Time	Entity #	Event Type
	27.22	Dep	Entity 3 departs and Entity 4 goes into service	33.82	4	Dep
				38.18	6	Arr
				40.0		End

Figure 1.17: At Time 27.22

Question 18: What is the next event?

Question 19: What new events are added as a result of this event? (Give the event time, the entity, and the event type?)

Step 10: The result of this next event is shown in Figure 1.18.


System Animation	Time Now	Event Type	Process Description	Event Calendar		
				Event Time	Entity #	Event Type
	33.82	Dep	Entity 4 departs and Entity 5 goes into service	38.18	6	Arr
				40.0		End
				42.46	5	Dep

Figure 1.18: At Time 33.82

Question 20: What is the next event?

Question 21: What new events are added as a result of this event? (Give the event time, the entity, and the event type?)

Step 11: We are getting closer to the time to quit observing the system; however we have one more event and one more time update. The result of this next event is shown in Figure 1.18.

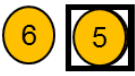
System Animation	Time Now	Event Type	Process Description	Event Calendar		
				Event Time	Entity #	Event Type
	38.18	Arr	Entity 6 arrives and must wait	40.0		End
				42.08	7	Arr
				42.46	5	Dep

Figure 1.19: At Time 38.18

Step 12: Finally, the next event calls for the “End” of the simulation at time 40.0. Only time is updated as the event calendar is unchanged. The final state is given in Figure 1.20.


System Animation	Time Now	Event Type	Process Description	Event Calendar		
				Event Time	Entity #	Event Type
	40.00	End	Simulation Ends	40.0		End
				42.08	7	Arr
				42.46	5	Dep

Figure 1.20: At Time 40.0

Our simulation approach is referred to as a “*Discrete-Event Simulation*” because the system only changes states at defined event times and the state changes are discrete in that the number in queue changes discretely as well as the number in system. If our model included variables that changed continuously with time, say like the water in a tank, then we wouldn’t have a discrete-change system and we would have to consider all points in time, not just those when the system changes state. Simulations that contain continuous variables are called “Continuous Simulations.” And we can have combinations of the two types of simulations. In fact, SIMIO can model both kinds of systems together – a multi-method simulation language.

Part 1.4: Input modeling and Simulation Output Analysis

Since we now have simulated the ice cream store time study, we can re-organize this information from an entity viewpoint, and compute the *same performance statistics* that we computed previously!

Question 22: Do we have enough information to draw conclusions about the present system?

Question 23: What can we do to extend the information available?

If we had a total of 45 interarrival times and 43 processing times, we could simulate a total of 480 minutes of time. For example, we may obtain the information given in Table 1.9. We have included the minimum, maximum, and average values over the 480 minutes. Currently we simulated “only” one day of 480 minutes.

Question 24: Now do you have enough information for a “present systems analysis?”

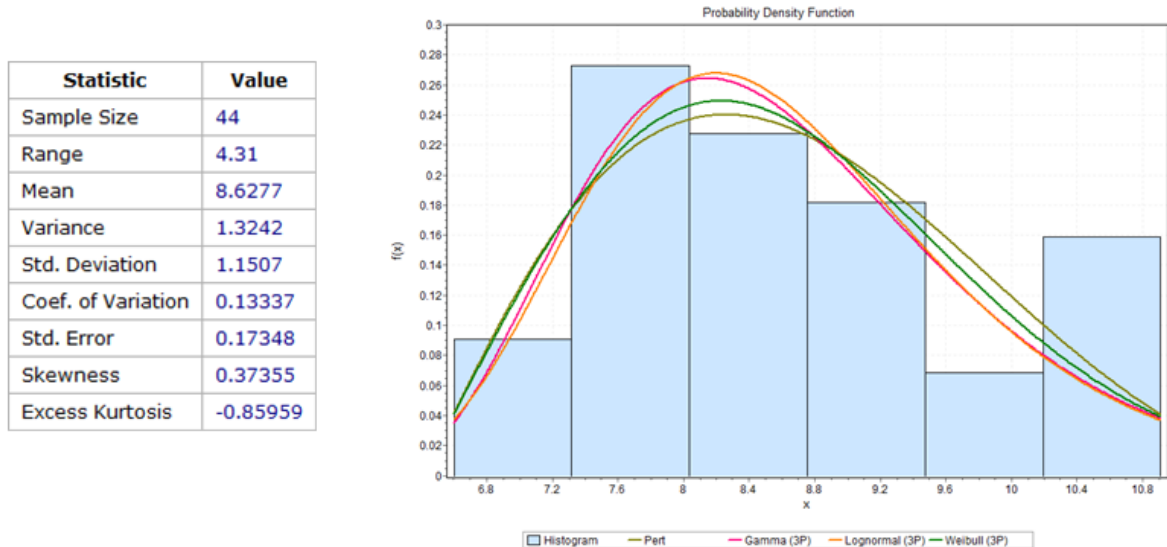
Question 25: Is a one-day of simulation long enough?

Table 1.9: Final Performance Measures for 480 Minutes

Performance Measure	Value
Total Production	43
Average waiting time in queue	9.59
Maximum waiting time in queue	35.65
Average total time in system	17.87
Maximum total time in system	42.65
Minimum total time in system	7.46
Time-average number of parts in queue	0.88
Maximum number of parts in queue	4
Ice Cream Attendant utilization	78%

One day doesn't give us much information about the day so more "days" of information is needed. But that means more interarrival times and more processing times. If we did a simulation of ten days, we would need approximately 450 interarrival times and 430 processing times. If that information came from an electronic record, then getting more data may not be a problem.⁵ But if our only alternative is time studies, then we will need to observe 10 days and that may be a greater intrusion into the actual operation than we can expect.

An alternative to more data collection is for us to find a "model" of this input data (i.e., interarrival and processing times). Perhaps we can use a statistical representation since processing times and interarrival times are most certainly random variables. We can often match the processing times and interarrival times to standard statistical distributions. First we make a histogram of the data we observed. Second, we try to pick out a statistical distribution, like a Gamma, Lognormal, Weibull, or Pert, to match the data. In fact, there is a wide range of software that can help with this undertaking. An example of this activity is shown in Figure 1.21.

**Figure 1.21: Matching Observed Processing Time to a Standard Statistical Distribution**

As a result of this input modeling, we can characterize a random processing time or a random interarrival time by a statistical model. For example, we might use a Pert distribution which has three parameters as: Pert (6, 8, 12) for a processing time or maybe a Gamma distribution which has two parameters as: 6.3 + Gamma(3.1, 0.7) where the 6.3 is the "location" of the distribution's origin.

⁵ An implication of this need for more data is that performance improvement should be one of the bases on which an information system is designed. The information should not be limited to accounting and reporting, but also performance improvement.

Once we characterize an input with a statistical model, there is simulation technology⁶ that allows us to “sample” from that statistical model. Repeated sampling will statistically reproduce the input model, but more importantly give us an “infinite” supply of input data. With that extended data, we can simulate many days of operation. Having many days of simulation provides us with the opportunity to gain an estimate of the precision of the average performance measures we compute. For example suppose we simulate our system for five days. Using simulation terminology, we performed five “replications” or “runs” of our simulation model. Each replication obtains new “samples” for its interarrival times and processing times. To compute the average and the standard deviation, we use the averages from each day (one day yields one average). The results are shown in Table 1.10.

Table 1.10: Results from Five Replication of 480 Minutes Each

<i>Replication</i>	1	2	3	4	5	Avg.	Std.Dev.
Avg. time in queue	9.59	15.26	12.98	8.08	21.42	13.47	5.26
Avg. no. in queue	0.88	1.62	1.17	0.76	2.31	1.35	0.63
Machine Utilization	0.78	0.86	0.75	0.78	0.81	0.80	0.04

Question 26: How many observations are used to compute the average and standard deviation in Table 1.10?

Question 27: Why only the averages instead of all the queue waiting times within a given replication?

We know that these statistics, like the average and the standard deviation, are themselves random variables. If we looked at another five days (either simulated or real) we wouldn’t get the exact same results because of the underlying variability in the model. We need to have some idea about how precise these summary statistics are. In order to judge how precise a given statistic is, we often use a *confidence interval*. For example, we computed the average waiting time as 13.47, but does this estimate have a lot of variability associated with it or are we pretty confident about that value?

Question 28: Confidence intervals in statistics are based on what famous distribution as well as famous theorem?

Any observations for which we want to create a confidence interval must satisfy the assumptions that the observations are “*independent and identically*” distributed. As a consequence the **Central Limit Theorem (CLT)** can be employed to compute a confidence interval.

Question 29: Are the observations, say of the waiting times, during a simulated day independent and identically distributed?

Question 30: For example, would the waiting time for entity #1, entity #2, and entity #3 be independent of each other? Would you expect the distributions of these waiting times to be identical?

Question 31: Would the number waiting in the queue at 9am be independent of the number waiting at 8:45am? Would the distribution of the number waiting be identical?

⁶ Random number and random variate generation methods.

Almost no statistic (performance measure)⁷ computed during a single simulation replication will be independent and identically distributed (e.g., the waiting time of entity #4 maybe dependent on entity #3 waiting and processing times). So instead, we use, for example, the average values computed over the day as an observation (i.e., we will determine the average of the averages).

Question 32: Are daily averages independent and identically distributed? Why?

Since these daily averages (or maximums or others) are independent and identically distributed, we can compute confidence intervals for our output statistics (based on the Central Limit Theorem).

Confidence Intervals on Expectations

A confidence interval for an expectation is computed using the following formula:

$$\bar{X} \pm t_{n-1, 1-\frac{\alpha}{2}} \frac{\hat{s}}{\sqrt{n}},$$

where \bar{X} is the sample mean, \hat{s} is the sample standard deviation of the data, $t_{n-1, 1-\frac{\alpha}{2}}$ is the upper $1 - \frac{\alpha}{2}$ critical point from the Student's t distribution with $n - 1$ degrees of freedom, and n is the number of observations (i.e., replications). Note, $\frac{\hat{s}}{\sqrt{n}}$ is often referred as the standard error or the standard deviation of the mean. As the

number of replications is increased, the standard error decreases. So a 95% confidence interval of the expected time in the queue using the data from Table 1.10 would be computed as the following calculation.

$$13.47 \pm 2.776 \frac{5.26}{\sqrt{5}}, \text{ or } 13.47 \pm 5.53 \text{ minutes or the interval of } [6.93, 20.00] \text{ minutes}$$

Question 33: Are you confident in the 13.47 average?

Question 34: Would you bet your job that the “true” average waiting time is 13.47 minutes?

So another way to express the confidence interval is given by the *Mean* \pm the *Half-width*, as in 13.47 ± 5.53

which SIMIO will use. The Central Limit Theorem allows the calculation of the confidence interval by asserting that the standard error (or the Standard Deviation of the Mean) can be computed by dividing the standard deviation of observations by the square root of the number of observations. If the observations are not independent and identically distributed then that relationship doesn't hold. As n increases sufficiently large the CLT and the inferential statistics on the mean of the population become valid.

Question 35: Does the confidence interval we computed earlier (i.e., 13.47 ± 5.53) mean that 95% of the waiting times fall in this interval?⁸

⁷ Of course there are exceptional cases.

⁸ Prediction intervals are used for observations.

Question 36: Does it mean that if we simulated 100 days that 95% average daily waiting times would fall in this interval?

Question 37: Or looked at another way. Can we say that there is a 95% chance that the “true, unknown” overall mean daily waiting time would fall in this interval?

So the confidence interval measures our “confidence” about the computed performance measures. A confidence interval is a statement about the mean (i.e., the mean waiting time), not about observations (i.e., individual waiting times). Confidence intervals can be “wider” than one would like. However, in simulation, we control how many days (i.e., replications) we perform in our analysis thus affecting the confidence on the performance measures.

Question 38: Using simulation, how can we improve the precision of our estimates (reduce the confidence interval width)?

So we can run more replications in our simulation if we want “tighter” confidence intervals.⁹

Comparing Alternative Scenarios

In simulation, we usually refer to a single model as a simulation “scenario” or a simulation “experiment”. Our model of the present ice cream store is a single scenario. However using simulation as a performance improvement tool, we are interested in simulating alternative models of, for example, our ice cream store. We would refer to each model as a scenario. So, for example, if we added a new ice cream making machine, this change would constitute a different simulation scenario. If we started an ad campaign and expected an increase in business in the ice cream store, we would have yet another scenario. In general, we would typically explore a whole bunch of scenarios, expecting to find improvements in the operations of the ice cream store. The simulation is our experimental lab.

So let’s reconsider a different scenario for our ice cream store. What would happen if the arrival rate was increased by 10% (i.e., more customers arrive per hour owing to an ad campaign)? We could reduce the interarrival time by 10% and make five additional replications of the 480 hour day. The results of the original and this alternative scenario are shown in Table 1.11 along with the confidence intervals.

Table 1.11: Results of Original and Added Time Scenarios

Replication	1	2	3	4	5	Avg.	Std.Dev.	LCL	UCL
Avg. time in queue	9.59	15.26	12.98	8.08	21.42	13.47	5.26	6.93	20.00
Avg. no. in queue	0.88	1.62	1.17	0.76	2.31	1.35	0.63	0.56	2.13
Machine Utilization	0.78	0.86	0.75	0.78	0.81	0.80	0.04	0.74	0.85

Increased Customer Arrival

Replication	1	2	3	4	5	Avg.	Std.Dev.	LCL	UCL
Avg. time in queue	14.77	24.42	18.39	12.10	31.18	20.17	7.69	10.62	29.72
Avg. no. in queue	1.50	2.77	1.80	1.18	3.74	2.20	1.05	0.90	3.50
Machine Utilization	0.85	0.92	0.82	0.82	0.85	0.85	0.04	0.80	0.90

Variability in the outcome creates problems for us. We recognize that we simply cannot compare scenarios of only one replication, but even with five, it’s hard to know if there are any real differences (although it appears so). Once again we need to rely on our statistical analysis to be sure we are drawing appropriate conclusions.

⁹ Note that we can also create a smaller confidence level by using a larger significance level or α value)

When comparing different sets of statistics like this, we would resort to the Student's t-test. A way to conduct the t-test is to compare confidence intervals for each of the two scenarios. If the confidence intervals overlap then we will fail to reject the null hypothesis that the mean time in system for the original system equals the new system.

The 95% confidence interval for the original five days was [6.94, 20.00] while for the increased arrival rate, the confidence interval is [12.48, 29.76]. Now comparing the confidence intervals, we see that they “overlap”, meaning *we cannot say there is a statistical difference in the average waiting time*. Without a statistical difference, any statement about the practical difference¹⁰ is without statistical foundation.

Question 39: What can we do to increase our chance of obtain a statistical difference?

Question 40: It is necessary to have a statistical foundation for our recommendations?

Although it is probably unnecessary to have a statistical foundation for every recommendation, we strive to do so to avoid the embarrassing situation of making a claim that is later shown to be erroneous – especially since our job may be at risk. By striving to have a statistical foundation for our recommendations, we take advantage of our entire toolbox in decision-making and promote our professionalism.

Part 1.5: Elements of the Simulation Study

The entire simulation study is composed of a number of elements, which we present here. Although these are presented in a sequential manner, rarely is a simulation study done without stopping to return to an earlier issue whose understanding has been enhanced. In many instances we work on several of the elements at the same time. But regardless of the order, we usually try to complete all the elements.

- *Understand the system:* Getting to know and understand the system is perhaps the most intense step and one that you will return to often as you develop a better understanding of what needs to be done.
- *Be clear about the goals:* Try to avoid “feature or scope creep”. There is a tendency to continue to expand the goals well beyond what is reasonable. Without clear goals, the simulation effort wanders. During this phase identify the performance measures of interest that will be used to evaluate the simulation.
- *Formulate the model representation:* Here you are clearly formulating the structure and input for your model. Don't spend a lot of time doing data collection at this point because, as you develop the model, its data needs will become clearer. Also be sure to involve the stakeholders in your formulation, to avoid missing important concerns.
- *Translate your conceptual representation into modeling software,* which in our case is SIMIO. A lot of time is spent learning SIMIO so you have a wide range of simulation modeling tools with which to build this and other models.
- *Determine the necessary input modeling:* At some point data will need to be collected on the inputs identified during the formulation and translation of the system into a computer model. The initial simulation model can be used to determine which inputs are the most sensitive with regard to the output which need to be collected. Fitting distributions is generally better but expert opinion can be used to get the model up and running.
- *Verify the simulation:* Be sure the simulation is working as it is expected without errors. Do some “stress tests” to see if it behaves properly when resources are removed or when demand is increased. Explain any

¹⁰ A practical difference means that the difference is important within the context of the problem. When we are concerned with a practical difference of unimportant or cheap items, then a statistical difference is not necessary.

“zeros” that appear in the output. Don’t assume you are getting counter-intuitive results when they may just be wrong.

- *Validate the model:* How does the model fit the real world? Is the simulation giving sufficient behavior that you have confidence in its output. Can you validly use the model for performance improvement?
- *Design scenarios:* Determine which alternatives you think will improve the performance of the present system and create the alternative simulation models and associate the models with scenarios.
- *Make runs:* do the simulation experiments for the scenarios. Be sure your simulation output generates the appropriate performance measures. Make multiple runs for each scenario.
- *Analyze results and get insight:* Examine carefully the output from the scenarios and begin to develop judgments about how to increase the performance of the system. Be sure the statistical analysis supports your conclusions.
- *Make Recommendation and Document:* Be sure to discuss the results with all the stakeholders and decision-makers. Make sure you have addressed the important problems and developed feasible recommendations. Document your work so you or someone else can return one year later and understand what you have done.

Part 1.6: Commentary

If you have worked carefully through this chapter, you will have a fundamental understanding of simulation that is completely independent of the simulation software or any particular application. Some of the key points have been.

- A simulation model consists of a system structure and input.
- The insertion and removal of events drive a simulation.
- Random variables are used to represent input.
- Simulation statistics include observations and time-persistent values.
- A simulation may consist of many performance measures.
- Verification and validation are important concerns in any simulation.
- By using a confidence interval, we have some measure of variability as well as central tendency of the simulation output.
- Computerization of simulation greatly facilitates its value in the present and in the future.