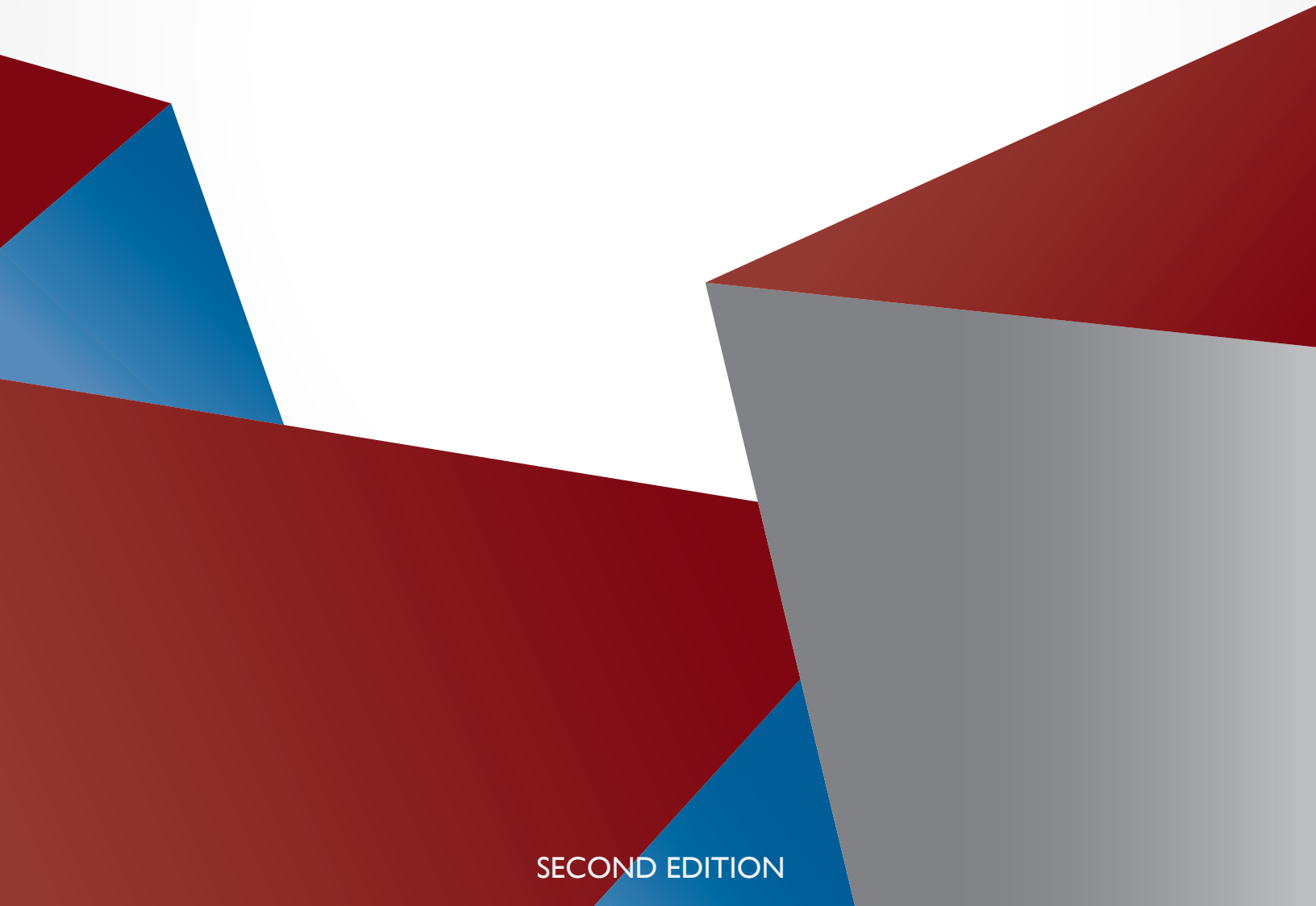
Beyond Lean: Simulation in Practice

Charles R. Standridge, Ph.D.



SECOND EDITION

Beyond Lean:
Simulation in Practice
Second Edition

©Charles R. Standridge, Ph.D.
Professor of Engineering
Assistant Dean
Padnos College of Engineering and Computing
Grand Valley State University
301 West Fulton
Grand Rapids, MI 49504
616-331-6759
Email:standric@gvsu.edu
Fax: 616-331-7215

December, 2011
Second Edition: April, 2013

Table of Contents

Part II Basic Organizations for Systems

Part III Lean and Beyond Manufacturing

Preface

Perspective

Lean thinking, as well as associated processes and tools, have involved into a ubiquitous perspective for improving systems particularly in the manufacturing arena. With application experience has come an understanding of the boundaries of lean capabilities and the benefits of getting beyond these boundaries to further improve performance. Discrete event simulation is recognized as one beyond-the-boundaries of lean technique. Thus, the fundamental goal of this text is to show how discrete event simulation can be used in addition to lean thinking to achieve greater benefits in system improvement than with lean alone.

Realizing this goal requires learning the problems that simulation solves as well as the methods required to solve them. The problems that simulation solves are captured in a collection of case studies. These studies serve as metaphors for industrial problems that are commonly addressed using lean and simulation.

Learning simulation requires doing simulation. Thus, a case problem is associated with each case study. Each case problem is designed to be a challenging and less than straightforward extension of the case study. Thus, solving the case problem using simulation requires building on and extending the information and knowledge gleaned from the case study. In addition, questions are provided with each case problem so that it may be discussed in a way similar to the traditional discussion of case problems used in business schools, for example.

An understanding of simulation methods is prerequisite to the case studies. A simulation project process, basic simulation modeling methods, and basic simulation experimental methods are presented in the first part of the text. An overview of how a simulation model is executed on a computer is provided. A discussion of how to select a probability distribution function to model a random quantity is included. Exercises are included to provide practice in using the methods.

In addition to simulation methods, simple (algebra-level) analytic models are presented. These models are used in partnership with simulation models to better understand system behavior and help set the bounds on parameter values in simulation experiments.

The second part of the text presents application studies concerning prototypical systems: a single workstation, serial lines, and job shops. The goal of these studies is to illustrate and reinforce the use of the simulation project process as well as the basic modeling and experimental methods. The case problems in this part of the text are directly based on the case study and can be solved in a straightforward manor. This provides students the opportunity to practice the basic methods of simulation before attempting more challenging problems.

The remaining parts of the text present case studies in the areas of system organization for production, supply chain management, and material handling. Thus, students are exposed to typical simulation applications and are challenged to perform case problems on their own.

A typical simulation course will make use of one simulation environment and perhaps probability distribution function fitting software. Thus, software tutorials are provided to assist students in learning to use the AutoMod simulation environment and probability distribution function fitting in JMP.

The text attempts to make simulation accessible to as many students and other professionals as possible. Experience seems to indicate that students learn new methods best when they are presented in the context of realistic applications that motivate interest and retention. Only the most fundamental simulation statistical methods, as defined in Law (2007) are presented. For example, the t-confidence interval is the primary technique employed for the statistical analysis of

simulation results. References to more advanced simulation statistical analysis techniques are given as appropriate. Only the most basic simulation modeling methods are presented, plus extensions as needed for each particular application study.

The text is intended to help prepare those who read it to effectively perform simulation applications.

Using the Text

The text is designed to adapt to the needs of a wide range of introductory classes in simulation and production operations. Chapters 1 - 5 provide the foundation in simulation methods that every student needs and that is pre-requisite for studying the remaining chapters. Chapters 6, 7, and 8 cover basic ideas concerning how the simulation methods are used to analysis systems as well as how systems work. I would suggest that these 8 chapters be a part of every class.

A survey of simulation application areas can be accomplished by selecting chapters from parts III, IV, and V. A focus on manufacturing systems is achieved by covering chapters 9, 10, 11, and 12. A course on material handling and logistics could include chapters 13 through 18.

Compute-based activities that are a part of the problem sets can be used to help students better understand how systems operate and how simulation methods work. The case problems can be discussed in class only or a student can perform a complete analysis of the problem using simulation.

Acknowledgements

The greatest joy I have had in developing this text is to recall all of the colleagues and students with whom I have worked on simulation projects and other simulation related activities since A. Alan B. Pritsker introduced me to simulation in January 1975.

One genesis for this text came from Professor Ronald Askin. As we completed work on the text: *Modeling and Analysis of Manufacturing Systems*, we surmised that an entire text on the applications of simulation was needed to fully discuss the material that had been condensed into a single chapter.

Professor Jon Marvel provided invaluable advice and direction on the development of the chapter on cellular manufacturing systems.

Special thanks are due to Dr. David Heltne, retired from Shell Global Solutions. Our joint work on using simulation to address logistics and inventory management issues over much of two decades greatly influenced those areas of the text.

The masters work of several students in the School of Engineering at Grand Valley State University is reflected within the text. These include Mike Warber, Carrie Grimard, Sara Maas, and Eduardo Perez. Joel Oostdyk and Todd Frazee helped gather information that was used in this text.

The specific contribution of each individual has been noted at the appropriate place in the text as well.

Part I
Introduction

This book discusses how, in a practical way, to overcome the limitations of the lean approach to transforming manufacturing systems as well as related in-plant, plant-to-plant, and plant-to-customer logistics. The primary tools in this regard are algebra based mathematical models and computer-based systems simulation as well as the integration of these two. Concepts, methods, and application studies related to designing and operating such systems are presented. Emphasis is on learning how to effectively make design, planning, and operations decisions by using a model based analysis and synthesis process. This process places equal emphasis on building models and performing analyses. This first part of the book discusses this process as well as the methods required for performing each of its steps.

The beyond lean approach is introduced in chapter 1 and illustrated with an industrial application. One proven process for performing a beyond lean study is presented. Some principles that guide such studies are discussed.

Methods are considered in chapters 2 through 5: model building, the computations needed to simulate a model and experimentation with models as well as modeling time delays and other random quantities. The basic logic used in simulation models is discussed. A process by which a distribution function can be selected to represent a random quantity, in the presence or absence of data, is given. An overview of the internal operations of a simulation engine is presented.

Chapter 2 describes the most basic ways in which systems are represented in simulation models. These basic ways provide a foundation for the models that are presented in the application study chapters. The modeling of common system components: arrivals, operations, routing, batching, and inventories, is discussed.

Chapter 3 presents a discussion of how to choose a distribution function to characterize the time between arrivals, processing times, and other system components that are modeled as random variables. A computer based interactive procedure for fitting a distribution function to data is emphasized. A discussion of selecting a distribution function in the absence of data is presented.

Chapter 4 presents the design and analysis of simulation experiments. Model and experiment verification and validation are included. An approach to specifying simulation experiments is discussed. Methods, including statistical analyses, for examining simulation results are included. An overview of animation is given.

Chapter 5 discusses the unseen work of a simulation engine in performing the computations necessary to simulate a model on a computer. Topics include the following: random number stream generation, random sampling from probability distribution functions, performance measure computation, event and entity list management, and state event detection.

Chapter 1
Beyond Lean: Process and Principles

1.1 Introduction

The application of lean concepts to the transformation of manufacturing and other systems has become ubiquitous and is still expanding (Learnsigma, 2007). The use of lean concepts has yielded impressive results. However, there seems to be a growing recognition of the limitations of lean and for a need to overcome them, that is to build upon lean successes or in other words to get beyond lean.[1] Getting beyond lean is the subject of this book.

Ferrin, Muller, and Muthler (2005) identify an important goal of any process improvement or transformation: find a correct, or at least a very good, solution that meets system design and operation requirements before implementation. Lean seems to be unable to meet this goal. As was pointed out by Marvel and Standridge (2009), a lean process does not typically validate the future state before implementation. Thus, there is no guarantee that a lean transformation will meet measurable performance objectives.

Marvel, Schaub & Weckman (2008) give one example of the consequences of not validating the future state before implementation in a case study concerning a tier-two automotive supplier. Due to poor performance of the system, a lean transformation was performed. One of the important components of the system was containers used to ship product to a number of customers. Each customer had a dedicated set of containers. The number of containers needed in the future state was estimated using a tradition lean static (average value) analysis, without taking the variability of shipping time to and from customers nor the variability in the time containers spent a customers into account. Thus, the number of containers in the future state was too low. Upon implementation, this resulted in the production system being idled due to the lack of containers. Thus, customer demand could not be met.

Standridge and Marvel (2006) describe the lean transformation of a system consisting of three processes. The second process, painting, was outsourced and performed in batches of 192 parts. Fearful of starving the third step in the process, the lean supply chain team deliberately over estimated the number of totes used to transport parts to and from the second step. In this system, totes are expensive and have a large footprint. Thus, the future state was systematically designed to be more expensive that necessary.

It seems obvious that in both these examples, the lean transformation resulted in a future state that was less than lean because it was not validated before implementation. Miller, Pawloski, and Standridge (2010) present a case study that further emphasizes this point and shows the benefits of such a validation. Marvel and Standridge (2009) suggest a modification of the lean process that includes future state validation as well as proposing that discrete event computer simulation be the primary tool for such a transformation because this tool has the following capabilities.

1. A simulation model can be analyzed using computer based experiments to assess future state performance under a variety of conditions.
2. Time is included so that dynamic changes in system behavior can be represented and assessed.
3. The behavior of individual entities such as parts, inventory levels, and material handling devices can be observed and inferences concerning their behavior made.
4. The effects of variability, both structural and random, on system performance can be evaluated.
5. The interaction effects among components can be implicitly or explicitly included.

[1] It is assumed that the reader has some familiarity with lean manufacturing / Toyota Production System concepts.

The discussion in this book focuses on how to build and use models to enhance lean transformations, that is to get beyond lean or to make lean more lean. The modeling perspective used incorporates the steps required to operate the system and how these steps are connected to each other. Models include the equipment and other resources needed to perform each step as well as the decision points and decision rules that govern how each step is accomplished and is connected to other steps. These models can include the sequencing procedures, routing, and other logic that is needed for a system to effectively operate.

Computer simulation models provide information about the temporal dynamics of systems that is available from no other source. This information is necessary to determine whether a new or transformed system will perform as intended before it is put into everyday use. Simulation leads to an understanding of why a system behaves as it does. It helps in choosing from among alternative system designs and operating strategies.

When a new system is designed or an existing system substantially changed, computer simulation models are used to generate information that aids in answering questions such as the following:

1. Can the number of machines or workers performing each operation adequate or must the number be increased?
2. Will throughput targets be reached that is will the required volume of output per unit time be achieved?
3. Can routing schemes or production schedules be improved?
4. Which sequencing scheme for inputs is best?
5. What should be the work priorities of material handling devices?
6. What decision rules work best?
7. What tasks should be assigned to each worker?
8. Why did the system behave that way?
9. What would happen if we did “this” instead?

1.2 An Industrial Application of Simulation

In order to better understand what simulation is and what problems it can be used to address, consider the following industrial application, which can was used to validate the future state of a plant expansion (Standridge and Heltne, 2000). A particular plant is concerned about the capital resources needed to load and ship rail cars in a timely fashion. A major capital investment in the plant will be made but the chance for future major expansions is minimal. Thus, all additional loading facilities, called load spots, needed for the foreseeable future must be built at the current time and must be justified based on product demand forecasts.

Each product can be loaded only at specific load spots. A load spot may be able to load more than one product. However, it is not feasible to load all products on all load spots. Cleverly assigning products to load spots may reduce the number of new load spots needed. Furthermore, maintenance of loading equipment is required. Thus, a load spot is not available every day.

The structure of the product storage and loading portion of the plant is shown in Figure 1-1. Products are stored in tanks with each tank dedicated to a particular product. Tanks are connected with piping to one or more load spots that serve one or two rail lines for loading. These numerous connections are not shown.

The primary measure of system performance is the percent of shipments made on time. The number of late shipments and the number of days each is late are also of interest. Shipping patterns are important so the number of pounds shipped each day for each product must be recorded. The utilization of each load spot must be monitored.

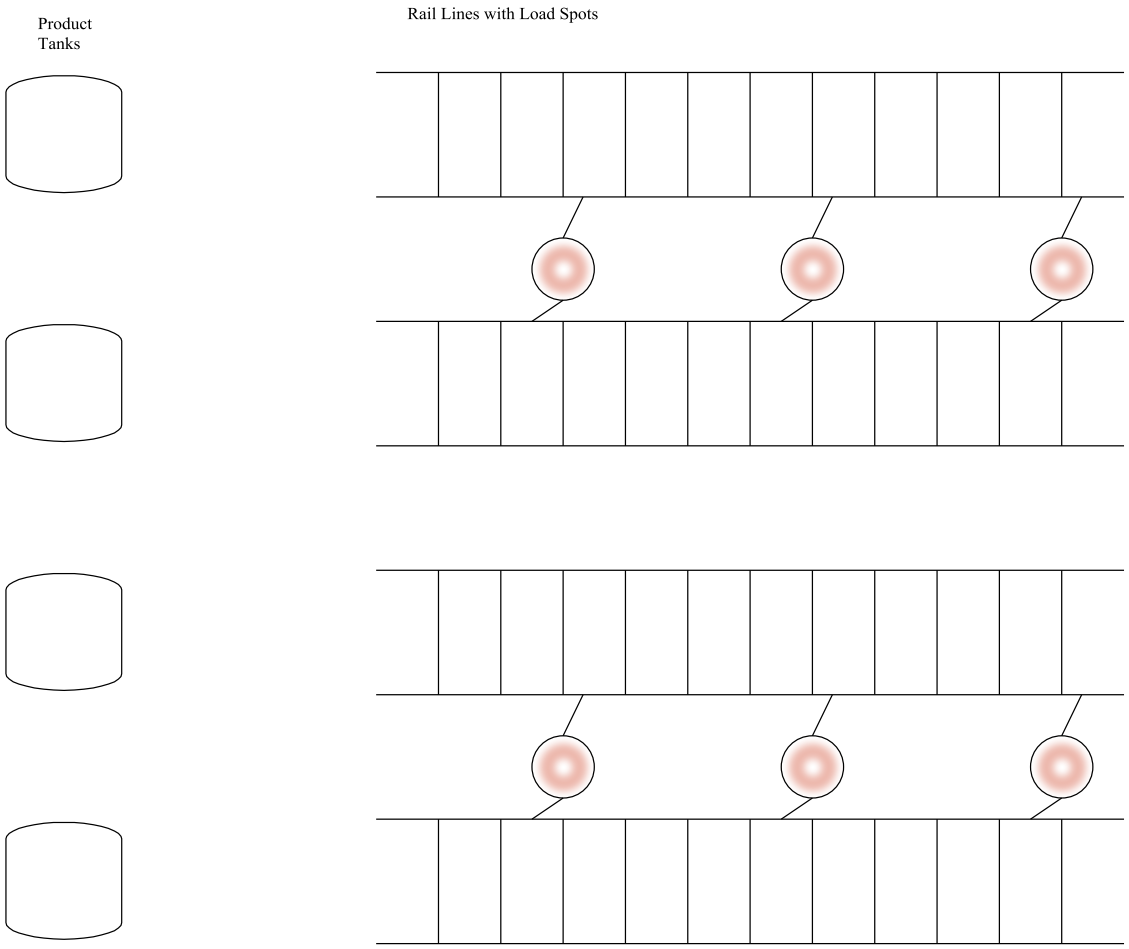


Figure 1-1: Industrial Simulation Application Example

The plant must lease rail cars to ship product to customers. Different products may require different types of rail cars, so the size of multiple rail car fleets must be estimated. In addition, the size of the plant rail yard must be determined as a function of the number of rail cars it must hold.

The model must account for customer demand for each product. Monthly demand ranges from 80% to 120% of its expected value. The expected demand for some products varies by month. In addition, each load spot must be defined by its capacity in rail cars loaded per day as well as the products it can load. Rail car travel times to customers from the plant and from the customer to the plant as well as rail car residence time at the customer site must be considered. Rail car maintenance specifications must be included.

Model logic is as follows. Each day the number of rail cars to ship is determined for each product. A rail car of the proper type waiting at the plant is assigned to the shipment. If no such rail car exists, the model simply creates a new one and the fleet size is increased by one.

Product loading of each rail car is assigned to a load spot. Load spots that can load the product are searched in order of least utilized first until an available load spot is assigned. A load spot may have been previously assigned loading tasks up to its capacity or may be in maintenance and unavailable for loading. Note that searching the load spots in this order tends to balance load spot utilization. The car is loaded and waits for the daily outbound train to leave the plant. The rail car proceeds to the customer, remains at the customer site until it is unloaded, and then returns to the plant. Maintenance is performed on the car if needed.

Analysts formulate alternatives by changing the assignment of products to load spots, the number of load spots available, and the demand for each product. These alternatives are based, in part, on model results previously obtained.

This example shows some of the primary benefits and unique features of simulation. Unique system characteristics, such as the assignment of particular products to particular load spots, can be incorporated into models. System specific logic for assigning a shipment to a load spot is used. Various types of performance measures can be output from the model such as statistical summaries about on time shipping and time series of values about the amount of product shipped. Statistics other than the average can be estimated. For example, the size of the rail yard must accommodate the maximum number of rail cars that can be there not the average. Random and time varying quantities, such as product demand, can be easily incorporated into a model.

1.3 The Process of Validating a Future State with Models

The simulation process used throughout this book is presented in this section.

Using simulation in designing or improving a system is itself a process. We summarize these steps into five strategic process phases (Standridge and Brown-Standridge, 1994; Standridge, 1998), which are similar to those in Banks, Carson, Nelson, and Nicol (2009). The strategic phases and the tactics used in each are shown in Table 1-1.

The first strategic phase in the simulation project process is the definition of the system design or improvement issues to be resolved and the characteristics of a solution to these issues. This requires identification of the system objects and system outputs that are relevant to the problem as well as the users of the system outputs and their requirements. Alternatives thought to result in system improvement are usually proposed. The scope of the model is defined, including the specification of which elements of a system are included and which are excluded. The quantities used to measure system performance are defined. All assumptions on which the model is based are stated. All of the above items should be recorded in a document. The contents of such a document is often referred to as the **conceptual model.** A team of simulation analysts, system experts, and managers performs this phase.

The construction of models of the system under study is the focus of the second phase. Simulation models are constructed as described in the next chapter. If necessary to aid in the construction of the simulation model, descriptive models such as flowcharts may be built.

Gaining an understanding of a system requires gathering and studying data from the system if it exists or the design of the system if it is proposed. Simulation model parameters are estimated using system data.

The simulation model is implemented as a computer program. Simulation software environments include model builders that provide the functionality and a user interface tailored to model building as well as automatically and transparently preparing the model for simulation.

Table 1-1: Phases and Tactics of the Simulation Project Process

Strategic Phase	Tactics
1. Define the Issues and Solution Objective	1. Identify the system outputs as well as the people who use them and their requirements. 2. Identify the systems that produce the outputs and individuals responsible for these systems. 3. Propose initial system alternatives that offer solution possibilities. 4. Identify all elements of the system that are to be included in the model. 5. State all modeling assumptions. 6. Specify system performance measures.
2. Build Models	1. Construct and implement simulation models. 2. Acquire data from the system or its design and estimate model parameters.
3. Identify Root Causes and Assess Initial Alternatives	1. Verify the model 2. Validate the model. 3. Design and perform the simulation experiments. 4. Analyze and draw inferences from the simulation results about system design and improvement issues. 5. Examine previously collected system data to aid in inference drawing.
4. Review and Extend Previous Work	1. Meet with system experts and managers. 2. Modify the simulation model and experiment. 3. Make additional simulation runs, analyze the results and draw inferences.
5. Implement Solutions and Evaluate	1. Modify system designs or operations. 2. Monitor system performance.

The third strategic phase involves identifying the system operating parameters, control strategies, and organizational structure that impact the issues and solution objectives identified in the first phase. Cause and effect relationships between system components are uncovered. The most basic and fundamental factors affecting the issues of interest, the root causes, are discovered. Possible solutions proposed during the first phases are tested using simulation experiments. Verification and validation are discussed in the next section as well as in Chapter 3.

Information resulting from experimentation with the simulation model is essential to the understanding of a system. Simulation experiments can be designed using standard design of experiments methods. At the same time, as many simulation experiments can be conducted as computer time and the limits on project duration allows. Thus, experiments can be replicated as needed for greater statistical precision, designed sequentially by basing future experiments on the results of preceding ones, and repeated to gather additional information needed for decision making.

The fourth strategic phase begins with a review of the work accomplished in phases one through three. This review is performed by a team of simulation analysts, system experts, and managers. The results of these meetings are often requests for additional alternatives to be evaluated, additional information to be extracted from simulation experiments, more detailed models of system alternatives, and even changes in system issues and solution objectives. The extensions and embellishments defined in this phase are based on conclusions drawn from the system data, simulation model, and simulation experiment results. The fourth stage relies on the ability to adapt simulation models during the course of a project and to design simulation experiments

sequentially. Alternative solutions may be generated using formal ways for searching a solution space such as a response surface method. In addition, system experts may suggest alternative strategies, for example alternative part routings based on the work-in-process inventory at workstations. Performing additional experiments involves modifications to the simulation model as well as using new model parameter values.

Physical experiments using the actual system or laboratory prototypes of the system may be performed to confirm the benefits of the selected system improvements.

In the fifth phase, the selected improvements are implemented and the results monitored.

The iterative nature of the simulation project process should be emphasized. At every phase, new knowledge about the system and its behavior is gained. This may lead to a need to modify the work performed at any preceding phase. For example, the act of building a model, phase 2, may lead to a greater understanding of the interaction between system components as well as to redoing phase 1 to restate the solution objectives. Analyzing the simulation results in phase 3 may point out the need for more detailed information about the system. This can lead to the inclusion of additional system components in the model as phase 2 is redone.

Sargent (2009) states that model **credibility** has to do with creating the confidence managers and systems experts require in order to use a model and the information derived from that model for decision making. Credibility should be created as part of the simulation process. Managers and systems experts are included in the development of the conceptual model in the first strategic phase. They review the results of the second and third phases including model verification and validation as well as suggesting model modifications and additional experimentation. Simulation results must include quantities of interest to managers and systems experts as well as being reported in a format that they are able to review independently. Simulation input values should be organized in a way, such as a spreadsheet, that they understand and can review. Thus, managers and systems experts are an integral part of a simulation project and develop a sense of ownership in it.

Performing the first and last steps in the improvement process requires knowledge of the context in which the system operates as well as considerable time, effort, and experience. In this book, the first step will be given as part of the statement of the application studies and exercises and the last step assumed to be successful. Emphasis is given to building models, conducting experiments, and using the results to help validate and improve the future state of a transformed or new system.

1.4 Principles for Simulation Modeling and Experimentation

Design (analysis and synthesis) applies the laws of basic science and mathematics. Ideally, simulation models would be constructed and used for system design and improvement based on similar laws or principles. The following are some general principles that have been found to be helpful in conceiving and performing simulation projects, though derivation from basic scientific laws or rigorous experimental testing is lacking for most of them.

1. Simulation is both an art and a science (Shannon, 1975).

One view of the process of building a simulation model and applying it to system design and improvement is given in Figure 1-2.

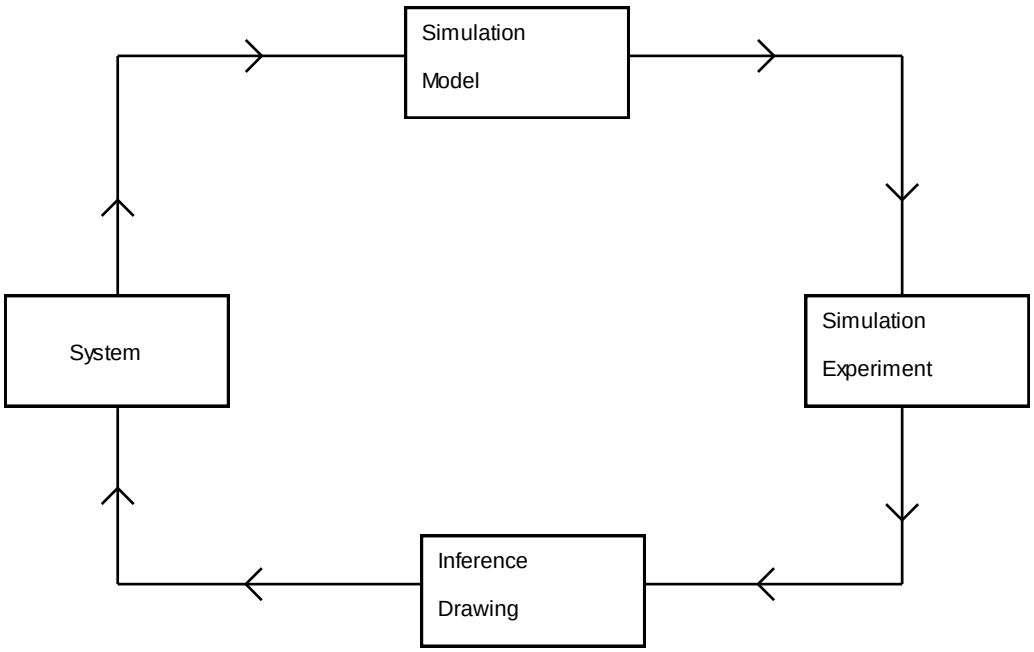


Figure 1-2: Simulation for Systems Design and Improvement.

A mathematical-logical form of an existing or proposed system, called a simulation model, is constructed (art). Experiments are conducted with the model that generates numerical results (science). The model and experimental results are interpreted to draw conclusions about the system (art). The conclusions are implemented in the system (science and art).

2. Computer simulation models conform both to system structure and to available system data (Pritsker, 1989).

Simulation models emphasize the direct representation of the structure and logic of a system as opposed to abstracting the system into a strictly mathematical form. The availability of system descriptions and data influences the choice of simulation model parameters as well as which system objects and which of their attributes can be included in the model. Thus, simulation models are analytically intractable, that is exact values of quantities that measure system performance cannot be derived from the model by mathematical analysis. Instead, such inferencing is accomplished by experimental procedures that result in statistical estimates of values of interest. Simulation experiments must be designed as would any laboratory or field experiment. Proper statistical methods must be used in observing performance measure values and in interpreting experimental results.

3. Simulation supports experimentation with systems at relatively low cost and at little risk.

Computer simulation models can be implemented and experiments conducted at a fraction of the cost of the P-D-C-A cycle of lean used to improve the future state to reach operational performance objectives. Simulation models are more flexible and adaptable to changing requirements than P-D-C-A. Alternatives can be assessed without the fear that negative consequences will damage day-to-day operations. Thus, a great variety of options can be considered at a small cost and with little risk.

For example, suppose a lean team want to know if a new proposed layout for a manufacturing facility would increase the throughput and reduce the cycle time. The existing layout could be

changed and the results measured, consistent with the lean approach. Alternatively, simulation could be used to assess the impact of the proposed new layout.

4. Simulation models adapt over the course of a project.

As was discussed in the previous section, simulation projects can result in the iterative definition of models and experimentation with models. Simulation languages and software environments are constructed to help models evolve as project requirements change and become more clearly defined over time.

5. A simulation model should always have a well-defined purpose.

A simulation model should be built to address a clearly specified set of system design and operation issues. These issues help distinguish the significant system objects and relationships to include in the model from those that are secondary and thus may be eliminated or approximated. This approach places bounds on what can be learned from the model. Care should be taken not to use the model to extrapolate beyond the bounds.

6. "Garbage in - garbage out" applies to models and their input parameter values (Sargent, 2009).

A model must accurately represent a system and data used to estimate model input parameter values must by correctly collected and statistically analyzed. This is illustrated in Figure 1-3.

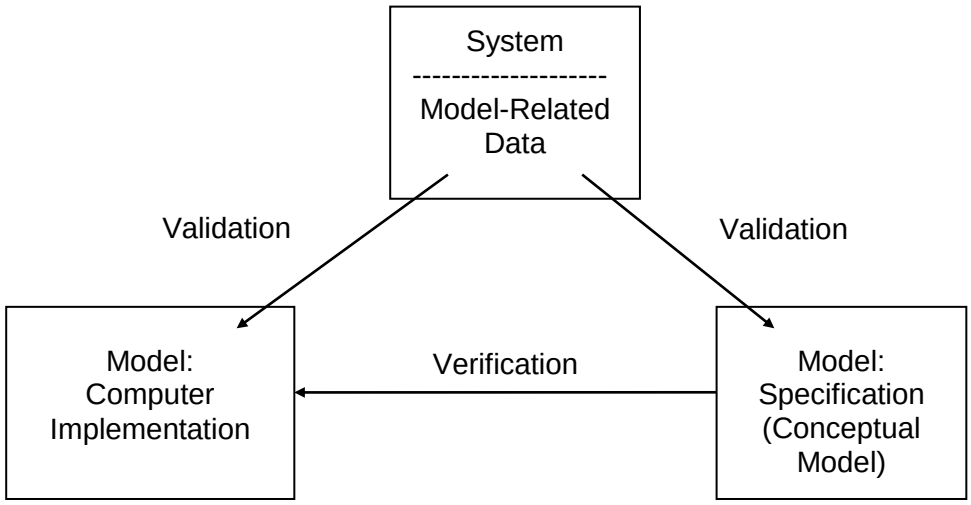


Figure 1-3: Model Validation and Verification.

There are two versions of a simulation model, the one specified "on paper" (the conceptual model) in the first strategic phase of the project process and the one implemented in the computer in the second strategic phase. Verification is the process of making sure these two are equivalent. Verification is aided, at least in part, by expressing the "on paper" model in a graphical drawing whose computer implementation is automatically performed.

Validation involves compiling evidence that the model is an accurate representation of the system with respect to the solution objectives and thus results obtained from it can be used to make decisions about the system under study. Validation has to do with comparing the system and the data extracted from it to the two simulation models and experimental results. Conclusions drawn from invalid models could lead to system "improvements" that make system performance worse

instead of better. This makes simulation and system designers who use it useless in the eyes of management.

7. Variation matters.

A story is told of a young university professor who was teaching an industrial short course on simulation. He gave a lengthy and detailed explanation of a sophisticated technique for estimating the confidence interval of the mean. At the next break, a veteran engineer took him aside and said "I appreciate your explanation, but when I design a system I pretty much know what the mean is. It is the variation and extremes in system behavior that kill me."

Variation has to do with the reality that no system does the same activity in exactly the same way or in the same amount of time always. Of course, estimating the mean system behavior is not unimportant. On the other hand, if every aspect of every system operation always worked exactly on the average, system design and improvement would be much easier tasks. One of the deficiencies of lean is that such an assumption is often implicitly made.

Variation may be represented by the second central moment of a statistical distribution, the variance. For example, the times between arrivals to a fast food restaurant during the lunch hour could be exponentially distributed with mean 10 seconds and, therefore, variance 100 seconds. Variation may also arise from decision rules that change processing procedures based on what a system is currently doing or because of the characteristics of the unit being processed. For instance, the processing time on a machine could be 2 minutes for parts of type A and 3 minutes for parts of type B.

There are two kinds of variation in a system: special effect and common cause. Special effect variation arises when something out of the ordinary happens, such as a machine breaks down or the raw material inventory becomes exhausted because of an unreliable supplier. Simulation models can show the step by step details of how a system responds to a particular special effect. This helps managers respond to such occurrences effectively.

Common cause variation is inherent to a normally operating system. The time taken to perform operations, especially manual ones, is not always the same. Inputs may not be constantly available or arrive at equally spaced intervals in time. They may not all be identical and may require different processing based on particular characteristics. Scheduled maintenance, machine set up tasks, and worker breaks may all contribute. Often, one objective of a simulation study is to find and assess ways of reducing this variation.

Common cause variation is further classified in three ways. Outer noise variation is due to external sources and factors beyond the control of the system. A typical example is variation in the time between customer orders for the product produced by the system. Variational noise is indigenous to the system such as the variation in the operation time for one process step. Inner noise variation results from the physical deterioration of system resources. Thus, maintenance and repair of equipment may be included in a model.

8. Looking at all the simulated values of performance measures helps.

Computer-based simulation experiments result in multiple observations of performance measures. The variation in these observations reflects the common cause and special effect variation inherent in the system. This variation is seen in graphs showing all observations of the performance measures as well as histograms and bar charts organizing the observations into categories. Summary statistics, such as the minimum, maximum, and average, should be computed from the observations. Figure 1-4 shows three sample graphs.

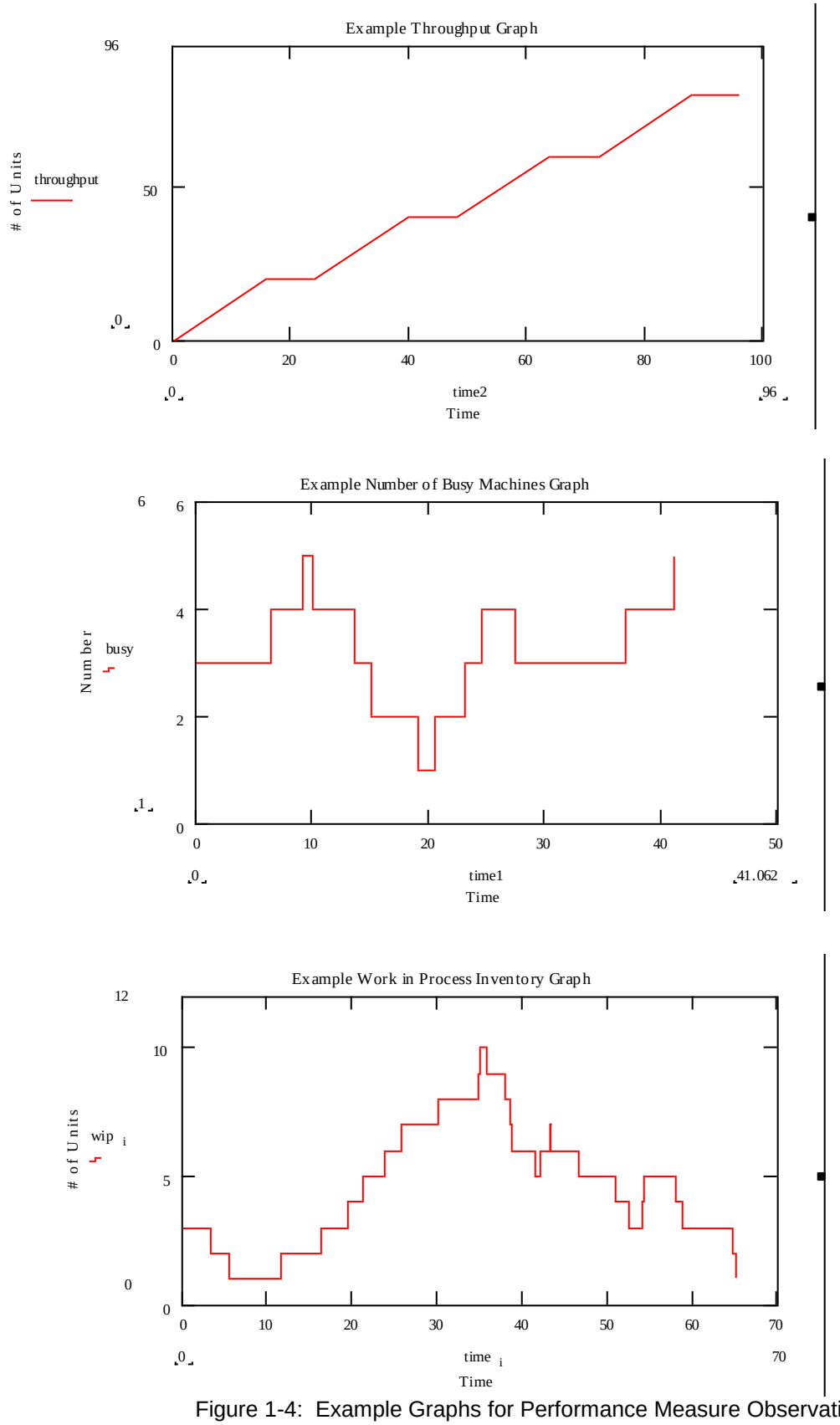
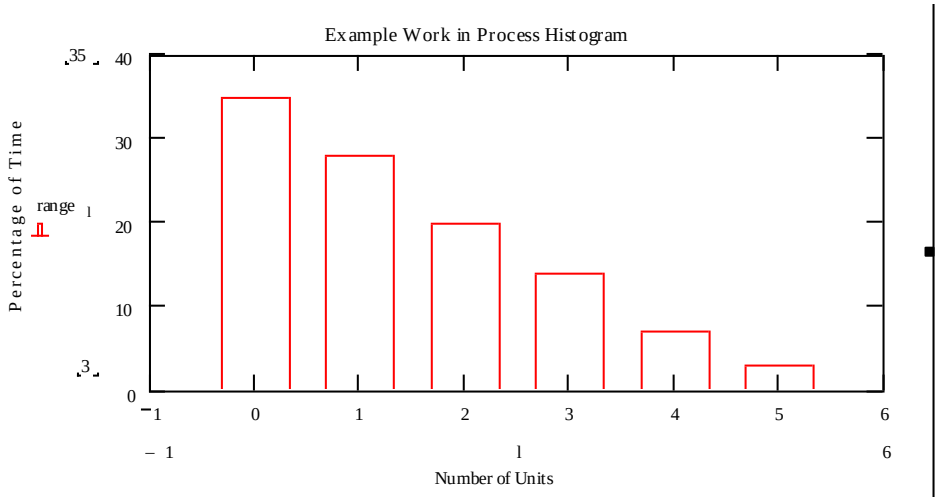


Figure 1-4: Example Graphs for Performance Measure Observations

The first shows how a special effect, machine failure, results in a build up of partially completed work. After the machine is repaired, the build up declines. The second shows the pattern of the number of busy machines at one system operation over time. The high variability suggests a high level of common cause variation and that work load leveling strategies could be employed to reduce the number of machines assigned to the particular task. The third graph shows the total system output, called throughput, over time. Note that there is no increase in output during shutdown periods, but otherwise the throughput rate appears to be constant.

Figure 1-5 shows a sample histogram and a sample bar chart. The histogram shows the sample distribution of the number of discrete parts in a system that appears to be acceptably low most of the time. The bar chart shows how these parts spend their time in the system. Note that one-half of the time was spent in actual processing which is good for most manufacturing systems.



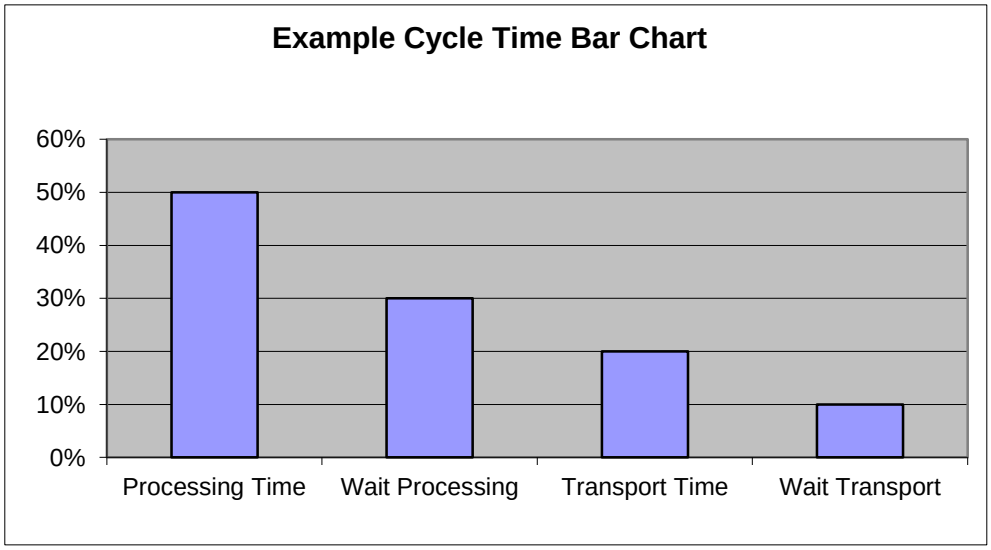


Figure 1-5: Example Histogram and Bar Charts for Performance Measure Observations

9. Simulation experimental results conform to unique system requirements for information.

Using simulation, the analyst is free to define and compute any performance measure of interest, including those unique to a particular system. Transient or time varying behavior can be observed by examining individual observations of these quantities. Thus, simulation is uniquely able to generate information that leads to a thorough understanding of system design and operation.

Though unique performance measures can be defined for each system, experience has shown that some categories of performance measures are commonly used:

1. System outputs per time interval (throughput) or the time required to produce a certain amount of output (makespan).
2. Waiting for system resources, both the number waiting and the waiting time.
3. The amount of time, lead time, required to convert individual system inputs to system outputs.
4. The utilization of system resources.
5. Service level, the ability of the system to meet customer requirements.

10. What goes into a model must come out or be consumed.

Every unit entering a simulation model for processing should be accounted for either as exiting the model or assembled with other units or destroyed. Accounting for every unit aids in verification.

11. Results for analytic models should be employed.

Analytic models can be used to enhance simulation modeling and experimentation. Result of analytic models can be used to set lower and upper bounds on system operating parameters such as inventory levels. Simulation experiments can be used to refine these estimates. Analytic models and simulation models can compute the same quantities, supporting validation efforts.

12. Simulation rests on the engineering heritage of problem solving.

Simulation procedures are founded on the engineering viewpoint that solving the problem is of utmost importance. Simulation was born of the necessity to extract information from models where analytic methods could not. Simulation modeling, experimentation, and software environments have evolved since the 1950's to meet expanding requirements for understanding complex systems.

Simulation assists lean teams in building a consensus based on quantitative and objective information. This helps avoid "design by argument". The simulation model becomes a valuable team member and is often the focus of team discussions.

This principle is the summary of the rest. Problem solving requires that models conform to system structure and data (2) as well as adapting as project requirements change (4). Simulation enhances problem solving by minimizing the cost and risk of experimentation with systems (3). Information requirements for problem solving can be met (9). Analytic methods can be employed where helpful (11).

1.5 Approach

The fundamental premise of this book is that learning the problems that simulation solves, as well as well as the modeling and experimental methods needed to solve these problems, is fundamental to understanding and using this technique.

Simulation models are built both from knowledge of basic simulation methods and by analogy to existing models. Similarly, computer-based experiments are constructed using basic experiment design techniques and by analogy to previous experiments. This premise is the foundation of this book. First, simulation methods for model building, simulation experimentation, modeling time delays and other random quantities as well as the implementation of simulation experiments on a computer are presented in the remaining chapters in this part of the book.

While each simulation model of each system can be unique, experience has shown that models have much in common. These common elements and their incorporation into models are discussed in chapter 2. These elementary models, as well as extensions, embellishments, and variations of them, are used in building the models employed in the application studies.

Starting in the next part of the book, the simulation project process discussed in section 1.4 is used in application studies involving a wide range of systems. Basic simulation modeling, experimentation, and system design principles presented in part 1 are illustrated in the application study. Additional simulation modeling and experimental methods applicable to particular types of problems are introduced and illustrated. Readers are asked to solve application problems based on the application studies and related simulation principles.

1.6 Summary

Simulation is a widely applicable modeling and analysis method that assists in understanding the design and operations of diverse kinds of systems. A simulation process directs the step by step activities that comprise a simulation project. Basic methods, principles, and experience guide the development and application of simulation models and experiments.

Questions for Discussion

1. List ways of validating a simulation model.

2. Why might building a model graphically be helpful?

3. List the types of variation and tell why dealing with each is important in system design.

4. Differentiate "art" and "science" with respect to simulation.

5. What is the engineering heritage influence on the use of models?

6. Why does variation matter?

7. Why is the simulation project process iterative?

8. How does the simulation project process foster interaction between system experts and simulation modelers?

9. Why must simulation experiments be designed?

10. Why does a simulation project require both a model and an experiment?

11. List the steps in the simulation process.

12. List three ways in which the credibility of a simulation model could be established with managers and system experts.

13. Distinguish between verification and validation.

14. Make two lists, one of the simulation project process activities that managers and system experts participate in and one of those that they don't.

Active Learning Exercises.

1. Create a paper clip assembly line. A worker at the first station assembles two small paper clips. A worker at the second station adds one larger paper clip. One student feeds the first station with two paper clips at random intervals. Another student observes the line to note the utilization of the stations, the number of assemblies in the inter-station buffer, and the number of completed assemblies. Run the assembly line for 2 minutes. Discuss how this exercise is like a simulation model and experiment.

2. Have the students act out the operation of the drive through window of a fast food restaurant in the following steps.

a. Enter line of cars
b. Place order
c. Move to pay window
d. Pay for order
e. Move to pick-up window
f. Wait for order to be delivered
g. Depart

Before beginning, define performance measures and designate students to keep track of them. Emphasize that the actions taken by the students are the ones performed by the computer during a simulation.

Chapter 2
Simulation Model Building

2.1 Introduction

How simulation models are constructed is the subject of this chapter. A **simulation model** consists of a description of a system and how it operates. The model is expressed in the form of mathematical and logical relationships among the system components. Model building is the act and art of representing a system with a model (principle 1) to address a well-defined purpose (principle 5).

Since simulation models conform to system structure and available data (principle 2), model building involves some significant judgment and art. Thus, learning to build simulation models includes learning typical ways system components are represented in models as well as how to adapt and embellish these modeling strategies to each system.

Model building is greatly aided by using a **simulation language** that provides a set of standard, pre-defined modeling constructs. These modeling constructs are combined to construct models. A simulation software environment provides the user interface and functionality for model construction.

This chapter presents common system components: arrivals, operations, routing, batching, and inventories, as well as describing typical models of each component. These models of components can be combined, extended and embellished to form models of entire systems. Elementary modeling constructs commonly found in simulation languages are presented. The use of these constructs in modeling the common system components is illustrated.

2.2 Elementary Modeling Constructs

This section presents the model building perspective taken in this text. Basic modeling constructs are presented.

The operation of many systems can be effectively described by the sequence of steps taken to transform the inputs to the outputs as shown in a value stream map. This will be our perspective for model building. A sequence of steps specified in a model is called a **process**. A model consists of one or more such processes.

The modeling construct used to represent the part, customer, information, etc. that is transformed from an input to an output by the sequence of processing steps is an **entity.** Each individual entity is tracked as it moves through the processing steps. Processing can be unique for each entity with respect to such things as processing times and route through the steps. Thus, it is essential to be able to differentiate among the entities. This is accomplished by associating a unique set of quantities, called **attributes**, with each entity. Typical attributes include time of arrival to the system and type of part, customer, or other information.

Note that a value stream map shows in general how parts flow through a system. This might be viewed as a “macro” or big picture view of how a system operates. One characteristic of beyond lean is the use of models that give a more detailed or “micro” representation of a system. This helps in gaining of understanding of how a future state will actually operate and aids in ensuring a successful transformation.

Certain components of a system are required in performing a processing step on an entity. However, such a component may be scarce, that is of insufficient number to be available to every entity upon demand. Such a component is called a **resource**. Waiting or queuing by entities for a resource may be required. Typical resources may be machines or workers. Other system components, totes or WIP racks, may be scarce and modeled using resources.

A resource has at least two **states**, unique circumstances in which it can be. One of these is **busy**, for example in use operating on an entity. Another is **idle**, available for use. Typical model logic requires an entity to wait for the resource (or resources) required for a particular processing step to be in the idle state before beginning that step. When the processing step is begun, the resources enter the busy state. As many resource states as necessary may be defined and used in a model. Other common resource states are broken and under repair, off shift and unavailable, and in setup for a new part type.

Consider a workstation consisting of two machines. A basic modeling issue is: Should each machine be modeled using a distinct resource? Often, it does not matter which of the two machines is used to process an entity and operating information such as utilization is not required for each machine. In such a case, it is simpler to use a single resource to model the two machines. However, an additional concept: **number of units**, is necessary to model two (or mores) machines with one resource. There is one unit of the resource for each machine, two in this example.

State variables, and their values, describe the conditions in a system at any particular time. State variables must be included in a simulation model and typically include the number of units of each resource in each possible resource state as well as the number of entities waiting for idle units of each resource, the number of entities that have completed processing, and inventory levels.

Time is a significant part of simulation models of systems. For example, when in simulated time each process step begins and ends for each entity processed by that step must be determined. Sequencing what happens to each entity correctly in time, and thus correctly with respect to what happens to all other entities, must be accomplished.

In summary, modeling the behavior of a system over **time** requires specifying how the **entities** use the **resources** to perform the steps of a **process** as well as how the values of the entity **attributes** and of the **state variables**, including the state of the **units of each resource**, change.

2.3 Models of System Components

Typical system components include arrivals, operations, routing, batching, and inventories. Models of these components are presented in this section. The models may be used directly, modified, extended, and combined to support the construction of new models of entire systems.

2.3.1 Arrivals

Consider a workstation with a single machine in a manufacturing facility as shown in Figure 2-1. One issue encountered when modeling the workstation is specifying at what simulated time each entity arrives. This specification includes the following:

- When the first entity arrives.
- The time between arrivals of entities, which could be a constant or a random variable modeled with a probability distribution. In a lean environment, the time between arrivals should equal the takt time as will be discussed in chapter 6.
- How many entities arrive during the simulation, which could be infinite or have an upper limit.

Suppose the first entity arrives at time 0, a common assumption in simulation modeling, and the time between arrivals is the same for all entities, a constant 10 seconds. Then the first arrival is at time 0, the second at time 10 seconds, the third at time 20 seconds, the fourth at time 30 seconds, and so forth.

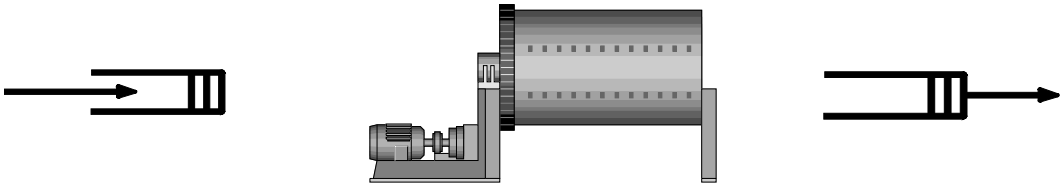


Figure 2-1: Example Single Server (Machine) Workstation

Suppose that time between arrivals varies. This implies that the time between arrivals is described by some probability distribution whose mean is the average time between arrivals. The variance of the distribution characterizes how much the individual times between arrivals differ from each other. For example, suppose the time between arrivals is exponentially distributed with mean 10 seconds, implying a variance of 10*10 = 100 seconds^2. The first arrival is at time 0. The second arrival could be at time 25 seconds, the third at time 31 seconds, the fourth at time 47 seconds, and so forth. Thus, the arrival process is a source of outer noise.

An example specification in pseudo-English follows.

Define Arrivals:		// mean must equal takt time
	Time of first arrival:	0
	Time between arrivals:	Exponentially distributed with a mean of 10 seconds Exponential (6) seconds
	Number of arrivals:	Infinite

2.3.2 Operations

The next issue encountered when modeling the single machine workstation in Figure 2-1 is specifying how the entities are processed by the workstation. Each entity must wait in the buffer (queue) preceding the machine. When available, the machine processes the entity. Then the entity departs the workstation.

A workstation resource is defined with the name WS to have 1 unit with states BUSY and IDLE using the notation WS/1: States(BUSY, IDLE). It is assumed that all units of a resource are initially IDLE. The operation of the machine is modeled as follows. Each entity waits for the single unit of the WS (workstation) resource to be available to operate on it that is the WS resource to be in the IDLE state. At the time processing begins WS becomes BUSY. After the operation time, the entity no longer needs WS, which becomes IDLE and available to operate on the next entity.

This may be expressed with the following statements:

```
Define Resources:
        WS/1 with states (Busy, Idle)

Process Single Workstation
Begin
        Wait until WS/1 is Idle in Queue QWS    // part waits for its turn on the machine
        Make WS/1 Busy                          // part starts turn on machine; machine is busy
        Wait for OperationTime                  // part is processed
        Make WS/1 Idle                          // part is finished; machine is idle
End
```

Note the pattern of resource state changes. For every process step that makes a resource enter the busy state, there must be a corresponding step that makes the resource enter the idle state. However, there may be many process steps between these two corresponding steps. Thus, many operations may be performed in sequence on an entity using the same resource.

Note also the two types of wait statements that delay entity movement through a process. Wait for means wait for a specified amount of simulation time to pass. Wait unit means wait until a state variable value meets a specified logical condition. This is a very powerful construct that also is consistent with ideas in event-based programming.

Consider another variation on the single machine workstation operation that illustrates the use of conditional logic in a simulation model. The machine requires a setup operation of 1.5 minutes duration whenever the Type of an entity differs from the Type of the preceding entity processed by the machine. For example, the machine may require one set of operational parameter settings when operating on one type of part and a second set when operating on a second type of part. The model of this situation is as follows.

```
Define Resources:
        WS/1 with states (Busy, Idle, Setup)

Define State Variables:
        LastType

Define Entity Attributes:
        Type

Process Single Workstation with Setup
Begin
        Wait until WS/1 is Idle in Queue QWS    // part waits for its turn on the machine
        Make WS/1 Busy                          // part starts turn on machine; machine is busy
        If LastType ! = Type then
        Begin
                Make WS/1 Setup
                Wait for SetupTime
                LastType = Type
                Make WS/1 Busy
        End
        Wait for OperationTime                  // part is processed
        Make WS/1 Idle                          // part is finished; machine is idle
End
```

A state variable, LastType stores the type of the last entity operated upon by the resource WS. Notice the use of conditional logic. Setup is performed depending on the sequence of entity

types. Such logic is common in simulation models and makes most of them analytically intractable. A new state, SETUP, is defined for WS. This resource enters the SETUP state when the setup operation begins and returns to the BUSY state when the setup operation ends.

Often a workstation is subject to interruptions. In general, there are two types of interruptions: scheduled and unscheduled. Scheduled interruptions occur at predefined points in time. Scheduled maintenance, work breaks during a shift, and shift changes are examples of scheduled interruptions. The duration of a scheduled interruption is typically a constant amount of time. Unscheduled interruptions occur randomly. Breakdowns of equipment can be viewed as unscheduled interruptions. An unscheduled interruption typically has a random duration.

A generic model of interruptions is shown in Figure 2-2. An interruption occurs after a certain amount of operating time that is either constant or random. The duration of the interrupt is either constant or random. After the end of the interruption, this cycle repeats.

Note that the transition to the INTERRUPTED state is modeled as occurring from the IDLE state. Suppose the resource is in the BUSY state when the interruption occurs. Typically, a simulation engine will assume that the resource finishes a processing step before entering the INTERRUPTED state. However, the end time of the interruption will be the same that is the amount of time in the INTERRUPTED state will be reduced by the time spent after the interruption occurs in the BUSY state. This approximation normally has little or no effect on the simulation results. In many systems, interruptions are often only detected or acted upon at the end of an operation. Using this approximation avoids having to include logic in the model as to what to do with the entity being processed when the interruption occurs. On the other hand, such logic could be included in the model if required.

This breakdown-repair process may be expressed in statements as follows:

```
Define Resource:
        WS/1: States(Busy, Idle, Unavailable)

Process Breakdown—Repair
Begin
        Do While 1=1                                    // Do forever
                Wait for TimeBetweenBreakdowns
                Wait until WS/1 is Idle
                Make WS/1 Unavailable
                Wait for RepairTime
                Make WS/1 Idle
        End Do
End
```

Consider an extension of the model of the single machine workstation with breakdowns (random interruptions) added. This model combines the process model for the single workstation with the process model for breakdown and repair. The WS resource now has three states: BUSY operating on an entity, IDLE waiting for an entity, and UNAVAILABLE during a breakdown. This illustrates how a model can consist of more than one process.

This model illustrates one of the most powerful capabilities of simulation. Two processes can change the state of the same resource but otherwise do not transfer information between each other. Thus, processes can be built in parallel and changed independently of each other as well as added to the model or deleted from the model as necessary. Thus, simulation models can be built, implemented, and evolved piece by piece.

Resource Definition
WS/1: States(IDLE, BUSY, UNAVAILABLE)

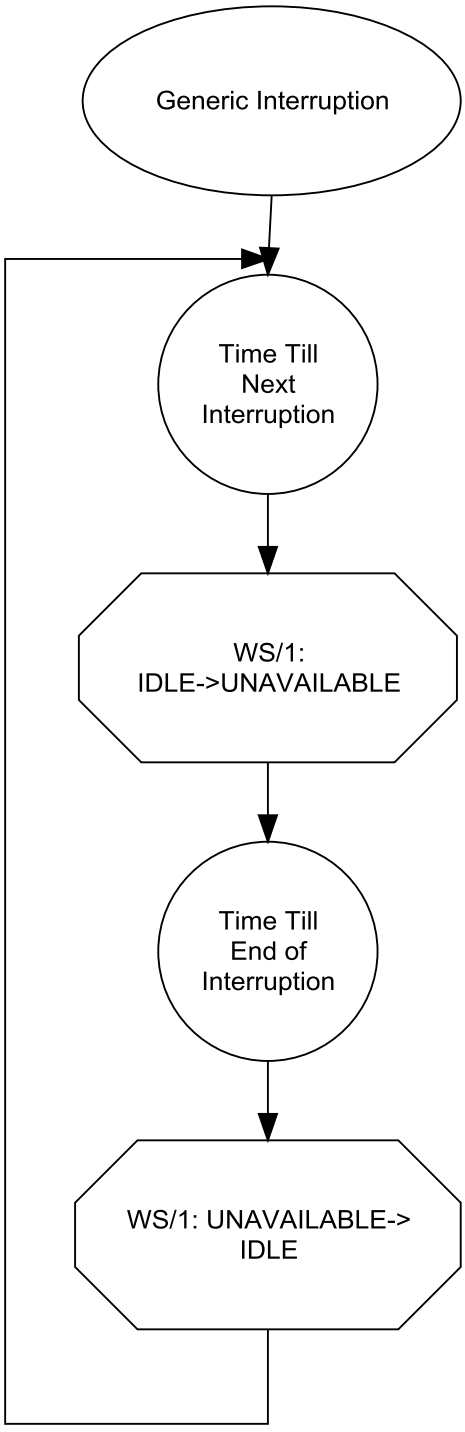


Figure 2-2: Generic Interruption Process

2.3.3 Routing Entities

In many systems, decisions must be made about the routing of entities from operation to operation. This section discusses typical ways systems make such routing decisions as well as techniques for modeling them. A distinct process for making the routing decision can be included in the model.

Consider a system that processes multiple item types using several workstations. Each workstation performs a unique function. Each job type is operated on by a unique sequence of workstations called a route. In a manufacturing environment, this style of organization is referred to as a job shop. It could also represent the movement of customers through a cafeteria where different foods: hot meals, sandwiches, salads, desserts, and drinks are served at different stations. Customers are free to visit the stations in any desired sequence.

Each entity in the model of a job shop organization could have the following attributes:

ArrivalTime: Time of arrival
Type: The type of job, which implies the route.
Location: The current location on the route relative to the beginning of the route.

In addition, the model needs to store the route for each type of job.

Suppose there are four workstations and three job types in a system. Figure 2-3 shows a possible set of routings in matrix form.

Job Type	First Operation	Second Operation	Third Operation	Fourth Operation
1	Workstation 1	Workstation 2	Workstation 3	Workstation 4
2	Workstation 3	Workstation 4	None	None
3	Workstation 4	Workstation 2	Workstation 3	None

Figure 2-3: Example Routing Matrix for A Manufacturing Facility.

A routing process is included in the simulation model to direct the entity to the workstation performing the next processing step. The routing process requires zero simulation time.

```
Define State Variable:
        Route(3, 4)

Define Attribute:
        Location

Define Attribute:
        Type

Process Routing
Begin
        Location += 1
        If Route(Type, Location) != None Then
        Begin
                Send to Process Route(Type, Location)
        End
        Else Begin
                Send to Process Depart
        End
End
```

The value of the Location attribute is incremented by one. The next workstation to process the entity is the one at matrix location (Type, Location). If this matrix location has a value of None, then processing of the entity is complete. Note again that the ability to compose a model of multiple processes is important.

Alternatively, routes may be selected dynamically based on current conditions in a system as captured in the values of the state variables. Such decision making may be included in a simulation model. This is another unique and very powerful capability of simulation modeling.

Consider a highly automated system in which parts wait in a central storage area for processing at any workstation. A robot moves the parts between the central storage area and the workstations. Alternatively if the following workstation is IDLE when an operation is completed, the robot moves the part directly to the next workstation instead of the storage area. The routing process for this situation follows, where WSNext is the resource modeling the following workstation.

```
Define Resource:
        WSNext/1:   States(Busy, Idle)
Define State Variable:
        CentralStorage

Process Conditional Routing
Begin
        If WSNext/1 is Idle Then
        Begin
                Send to Process ForWSnext
        End
        Else Begin
                CentralStorage += 1
        End
End
```

2.3.4 Batching

Many systems use a strategy of grouping items together and then processing all the items in the group jointly. This strategy is called batching. Batching may occur preceding a machine on the factory floor. Parts of the same type are grouped and processed together in order to avoid frequent setup operations. Other examples include:

- Bags of product may be stacked on a pallet until its capacity is reached. Then a forklift could be used to load the pallet on a truck for shipment.
- All deposits and checks received by a bank during the day are sent together to a processing center where bank account records are updated overnight.
- A tram operating from the parking lot in an amusement park to the entrance gate waits at the stop until a certain number of passengers have boarded.

Batching is the opposite of the lean concept of one-piece flow or one part per batch. There is a trade-off between batching and one-piece flow. Batching minimizes set up times and can make individual machines more productive. One-piece flow minimizes lead time, the time between starting production on a part and completing it, as well as in-process inventories.

Batching may be modeled by creating one group entity consisting of multiple individual entities. The group entity can be operated upon and routed as would an individual entity. Batching can be undone, that is the group can be broken up into the original individual entities.

Returning to the single server workstation example of Figure 2-2, suppose that completed parts are transported in groups of 20 from the output buffer to input buffer of a second workstation some distance away. At the second workstation, items are processed individually. In the simulation model of this situation, the first 19 entities, each modeling an item to be transported, must wait until the twentieth entity arrives. Then the 20 entities are moved together as a group. Each group is referred to as a **batch**. Upon arrival at the second workstation, each entity individually enters the workstation buffer. The batching and un-batching extension to the single server workstation model is as follows.

```
Define Resource:  WS/1: States(Busy, Idle)
Define List:      OutputBuffer

Process Single Workstation with Output Buffer
Begin
        Wait until WS/1 is Idle
        Make WS/1 Busy
        Wait for Operation Time
        Make WS/1 Idle

        If Length (OutputBuffer) < 19 then
        Begin
                Add to List(OutputBuffer)
        End
        Else Begin
                Wait for Transportation Time
                Send All Entities on List(OutputBuffer) to Process WS2
        End
End
```

Consider the case where each of the types of items processed by the workstation must be transported separately. In this situation, batching, transportation, and un-batching can be done as above except one batch is formed for each type of item.

2.3.5 Inventories

In many systems, items are produced and stored for subsequent use. A collection of such stored items is called an inventory, for example televisions waiting in a store for purchase or hamburgers under the hot lamp at a fast food restaurant. Customers desiring a particular item must wait until one is inventory.

Inventory processes have to do with adding items to an inventory and removing items from an inventory. The number of items in an inventory can be modeled using a state variable, for example INV_LEVEL. Placing an item in inventory is modeled by adding one to the state variable: INV_LEVEL += 1. Modeling the removal of an item from an inventory requires two steps:

1. Wait for an item to be in the inventory: WAIT UNTIL INV_LEVEL > 0
2. Subtract one from the number of items in the inventory: INV_LEVEL -= 1

2.4 Summary

Simulation model construction approaches have been presented. System components: arrivals, operations, routing, batching, and inventory management have been identified. How each component is commonly represented in simulation models has been discussed and illustrated.

Problems

1. Discuss why it is important to be able to employ previously developed models of system components in addition to the more basic modeling constructs provided by a simulation language in model building.

2. Discuss the importance of allowing multiple, parallel processes in a model.

(For each of the modeling problems that follow, use the pseudo-English code that has been presented in this chapter.)

3. Develop a model of a single workstation whose processing time is a constant 8 minutes. The station processes two part types, each with an exponentially distributed interarrival time with mean 20 minutes.

4. Embellish the model developed in 3 to include breakdowns. The time between breakdowns in exponentially distributed with mean 2 days. Repair time is uniformly distributed between 1 and 3 hours.

5. Build a model of a two-station assembly line serving three types of parts. The sequence of part types is random. The part types are distributed as follows: part type 1, 30%; part type 2; 50%, and part 3, 20%. Inter-arrival time is a constant 5 minutes. The first station requires a setup task of 1.5 minutes duration whenever the current part type is different from the preceding one. The operation times are the same regardless of part type: station 1, 3 minutes and station 2, 4 minutes.

6. Embellish the model in problem 5 for the case where there are two stations that perform the second operation. The part goes to the station with the fewer number of waiting parts.

7. Embellish the model in problem 5 for the case where a robot loads and unloads the second station. Loading and unloading each take 15 seconds.

8. Combine problems 5, 6, and 7 in one model.

9. Consider Bob's Burger Barn. Bob has a simple menu: burgers made Bob's way, french fries (one size), and soft drinks (one size). Customers place orders with one cashier who enters the order and collects the payment. They then wait near the counter until the order is filled. The time between customer arrivals during the lunch hour from 11:30 to 1:00 P.M. is exponentially distributed with mean 30 seconds. It takes a uniformly distributed time between 10 seconds and 40 seconds for order placement and payment at the cashier. The time to fill an order after the payment is completed is normally distributed with mean 20 seconds and standard deviation 5 seconds.

   a. Build a model of Bob's Burger Barn.

   b. Embellish the model for the case where a customer will leave upon arrival if there are more than 7 customers waiting for the cashier.

10. Consider the inside tellers at a bank. There is one line for 3 tellers. Customers arrive with an exponentially distributed time between arrivals with mean 1 minute. There are three customer types: 1, 10%; 2, 20%; and 3, 70%. The time to serve a customer depends on the type as follows: 1, 3 minutes; 2, 2 minutes; and 3; 30 seconds. Build a model of the bank.

11. Modify the model in problem 10 for the case where there is one line for each teller. Arriving customers choose the line with the fewest customers.

12. Develop a process model of the following situation. Two types of parts are processed by a station. A setup time of one minute is required if the next part processed is of a different type that the preceding part. Processing time at the station is the same for both part types: 10 minutes. Type 1 parts arrive according to an exponential distribution with mean 20 minutes. Type 2 parts arrive at the constant rate of 2 per hour.

13. Develop a process model of the following situation. A train car is washed and dried in a rail yard between each use. The same equipment provides for washing and drying one car at a time. Washing takes 30 minutes and drying one hour. Cars arrive at the constant rate of one each hour and three-quarters.

14. Develop a model of a service station with 10 self-service pumps. Each pump dispenses each of three grades of gasoline. Customer service at the pump time is uniformly distributed between 30 seconds and two minutes. One-third of the customers pay at the pump using a credit card. The remainder must pay a single inside cashier which takes an additional 1 minute to 2 minutes, uniformly distributed. The time between arrivals of cars is exponentially distributed with mean 1 minute.

15. Mike's Market has three check out lanes each with its own waiting line. One check out lane is for customers with 10 items or fewer. The check out time is 10 seconds plus 2 seconds per item. The number of items purchased is triangularly distributed with minimum 1, mode 10, and maximum 100. The time between arrivals to the check-out lanes is exponentially distributed with mean 1 minute.

16. Develop a more detailed model of Bob's Burger Barn (discussed in problem 9). Add an inventory for completed burgers and another inventory for completed bags of fries. Filling an order is an assembly process that requires removing a burger from its inventory and a bag of fries from its inventory. The burgers are completed at a constant rate of 2 per minute. It takes three minutes to deep fry six bags of fries.

17. Develop a model of the following inventory management situation. A part completes processing on one production line every 2 minutes, exponentially distributed and is placed in an inventory. A second production line removes a part from this inventory every two minutes.

18. Visit a fast food restaurant in the university student union and note how it serves customers. Specify a model of the customer service aspect of the restaurant using the component models in this chapter.

Chapter 3
Modeling Random Quantities

3.1 Introduction

This chapter deals with how to select a probability distribution to represent a random quantity in a simulation model. As seen in previous examples, random quantities are used to represent operation times, transportation times, and repair times well as the time between the arrival of entities and the time between equipment breakdowns. The type of an entity could be a random quantity, as could the number of units demanded by each customer from a finished goods inventory.

In determining the particular probability distribution function to use to model each random quantity, available data as well as the properties of the quantity being modeled must be taken into account. Estimation of distribution function parameters must be performed.

Frequently, data is not available. Choosing a distribution function in the absence of data is discussed including which distributions are commonly used in this situation. Software based procedures for choosing a distribution function when data is available, including fitting the data to a distribution function, are presented. The probability distributions commonly employed in simulation models are described.

3.2 Determining a Distribution in the Absence of Data

Often, parameter values for probability distributions used to model random quantities must be determined in the absence of data. There are many possible reasons for a lack of data. The simulation study may involve a proposed system. Thus, no data exists. The time and cost required to obtain and analyze data may be beyond the scope of the study. This could be especially true in the initial phase of a study where an initial model is to be built and initial alternatives analyzed in a short amount of time. The study team may not have access to the information system where the data resides.

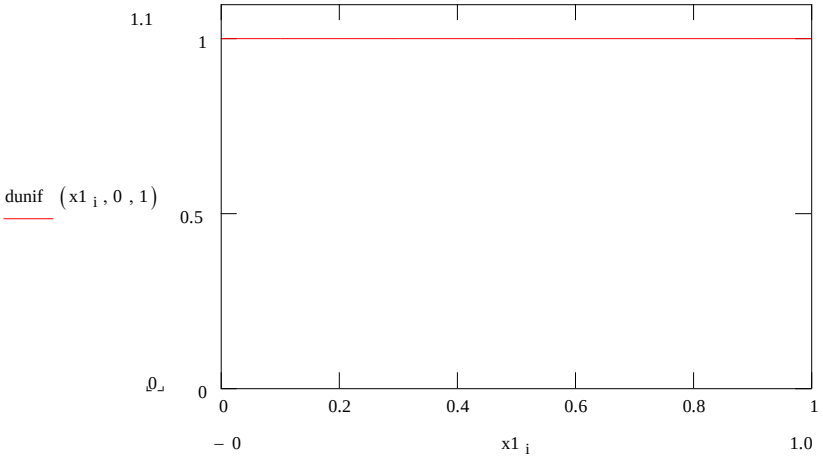
The distribution functions commonly employed in the absence of data are presented. An illustration of how to select a particular distribution to model a random quantity in this case is given.

3.2.1 Distribution Functions Used in the Absence of Data

Most often system designers or other experts have a good understanding of the “average” value. Often, what they mean by “average” is really the most likely value or mode. In addition, they most often can supply reasonable estimates of the lower and upper bounds that is the minimum and maximum values. Thus, distribution functions must be used that have a lower and upper bound and whose parameters can be determined using no more information than a lower bound, upper bound, and mode.

First consider the distribution functions used to model operation times. The uniform distribution requires only two parameters, the minimum and the maximum. Only values in this range [min, max] are allowed. All values between the minimum and the maximum are equally likely. Normally, more information is available about an operation time such as the mode. However, if only the minimum and maximum are available the uniform distribution can be used.

Figure 3-1 provides a summary of the uniform distribution.



Density Function Illustration

Parameters: min(imum) and max(imum)

Range: [min, max]

Mean: $\text{mean} = \frac{\min + \max}{2}$

Variance: $\text{variance} = \frac{(\max - \min)^2}{12}$

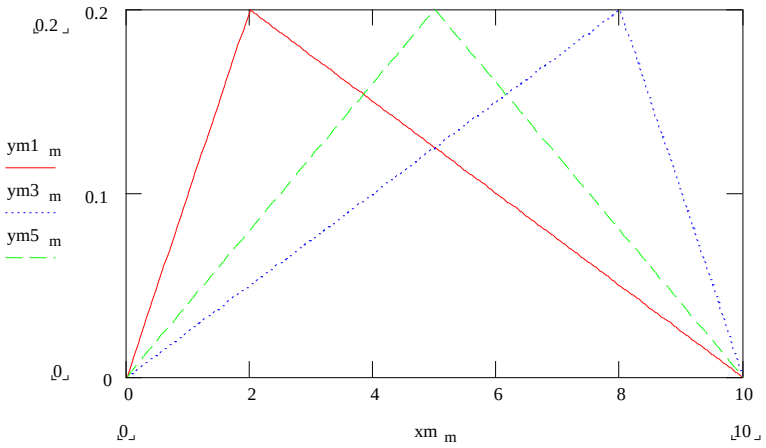
Density function: $f(x) = \frac{1}{\max - \min}; \min \le x \le \max$

Distribution function: $F(x) = \frac{x - \min}{\max - \min}; \min \le x \le \max$

Application: In the absence of data, the uniform distribution is used to model a random quantity when only the minimum and maximum can be estimated.

Figure 3-1: Summary of the Uniform Distribution

If the mode is available as well, the triangular distribution can be used. The minimum, maximum, and mode are the parameters of this distribution. Note that the mode can be closer to the minimum than the maximum so that the distribution is skewed to the right. Alternatively, the distribution can be skewed to the left so that the mode is closer to the maximum than the minimum. The distribution can be symmetric with the mode equidistant from the minimum and the maximum. These cases are illustrated in Figure 3-2 where a summary of the triangular distribution is given.



Density Function Illustrations

Parameters: min(imum), mode, and max(imum)

Range: [min, max]

Mean: $\frac{\text{min} + \text{mode} + \text{max}}{3}$

Variance: $\frac{\text{min}^2 + \text{mode}^2 + \text{max}^2 - \text{min} * \text{mode} - \text{min} * \text{max} - \text{mode} * \text{max}}{18}$

Density function:
$$f(x) = \begin{cases} \frac{2 * (x - \text{min})}{(\text{max} - \text{min}) * (\text{mode} - \text{min})} ; \text{min} \le x \le \text{mode} \\ \frac{2 * (\text{max} - x)}{(\text{max} - \text{min}) * (\text{max} - \text{mode})} ; \text{mode} < x < \text{max} \end{cases}$$

Distribution function:
$$F(x) = \begin{cases} \frac{(x - \text{min})^2}{(\text{max} - \text{min}) * (\text{mode} - \text{min})} ; \text{min} \le x \le \text{mode} \\ 1 - \frac{(\text{max} - x)^2}{(\text{max} - \text{min}) * (\text{max} - \text{mode})} ; \text{mode} < x < \text{max} \end{cases}$$

Application: In the absence of data, the triangular distribution is used to model a random quantity when the most likely value as well as the minimum and maximum can be estimated.

Figure 3-2: Summary of the Triangular Distribution

The beta distribution provides another alternative for modeling an operation time in the absence of data. The triangular distribution density function is composed of two straight lines. The beta distribution density function is a smooth curve. However, the beta distribution requires more information and computation to use than does the triangular distribution. In addition, the beta distribution is defined on the range [0,1] but can be easily shifted and scaled to the range [min,

max] using min + (max-min)*X, where X is a beta distributed random variable in the range [0, 1]. Thus, as did the uniform and triangular distributions, the beta distribution can be used for values in the range [min, max].

Using the beta distribution requires values for both the mode and the mean. Subjective estimates of both of these quantities can be obtained. However, it is usually easier to obtain an estimate of the mode than the mean. In this case, the mean can be estimated from the other three parameters using equation 3-1.

$$\text{mean} = \frac{\text{min} + \text{mode} + \text{max}}{3} \tag{3-1}$$

Pritsker (1977) gives an alternative equation that is similar to equation 3-2 except the mode is multiplied by 4 and the denominator is therefore 6.

The two parameters of the beta distribution are α_1 and α_2. These are computed from the minimum, maximum, mode, and mean using equations 3-2 and 3-3.

$$\alpha_1 = \frac{(\text{mean} - \text{min}) * (2 * \text{mode} - \text{min} - \text{max})}{(\text{mode} - \text{mean}) * (\text{max} - \text{min})} \tag{3-2}$$

$$\alpha_2 = \frac{(\text{max} - \text{mean}) * \alpha_1}{\text{mean} - \text{min}} \tag{3-3}$$

Most often for operation times, $\alpha_1 > 1$ and $\alpha_2 > 1$. Like the triangular distribution, the beta distribution can be skewed to the right $\alpha_1 < \alpha_2$, skewed to the right, $\alpha_1 > \alpha_2$, or symmetric, $\alpha_1 = \alpha_2$. A summary of these and other characteristics of the beta distribution is given in Figure 3-3.

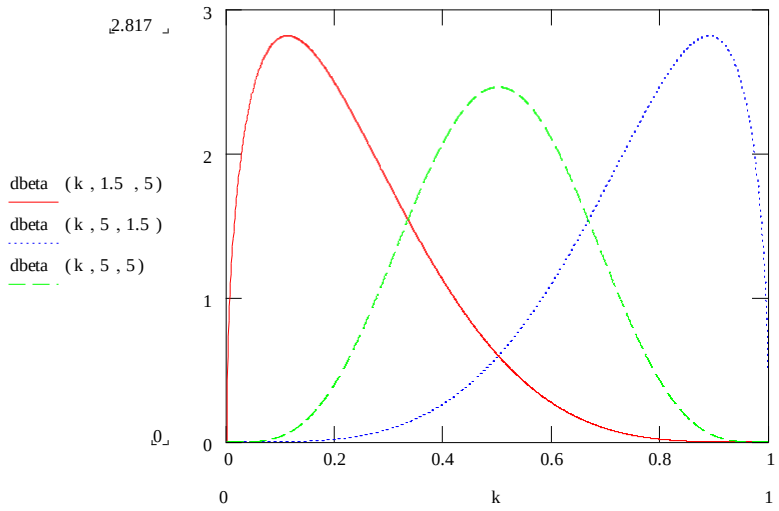
Next, consider modeling the time between entity arrivals. In the absence of data, all that may be known is the average number of entities expected to arrive in a given time interval. The following assumptions are usually reasonable when no data are available.

1. The entities arrive one at a time.
2. The mean time between arrivals is the same over all simulation time.
3. The numbers of customers arriving in disjoint time intervals are independent.

All of this leads to using the exponential distribution to model the times between arrivals. The exponential has one parameter, its mean. The variance is equal to the mean squared. Thus, the mean is equal to the mean time between arrivals or the time interval of interest divided by an estimate of the number of arrivals in that interval.

Using the exponential distribution in this case can be considered to be a conservative approach as discussed by Hopp and Spearman (2007). These authors refer to a system with exponentially distributed times between arrivals and service times as the practical worst case system. This term is used to express the belief that any system with worse performance is in critical need of improvement. In the absence of data to the contrary, assuming that arrivals to a system under study are no worse than in the practical worse case seems safe.

Figure 3-4 summarizes the exponential distribution.



Density Function Illustrations

Parameters: min(imum), mode, mean, and max(imum)

Range: [min, max]

Mean: $\frac{\alpha_1}{\alpha_1 + \alpha_2}$

Variance: $\frac{\alpha_1 * \alpha_2}{(\alpha_1 + \alpha_2)^2 * (\alpha_1 + \alpha_2 + 1)}$

Density function: $f(x) = \frac{x^{\alpha_1 - 1}(1 - x)^{\alpha_2 - 1}}{B(\alpha_1, \alpha_2)}; 0 < x < 1$

The denominator is the beta function.

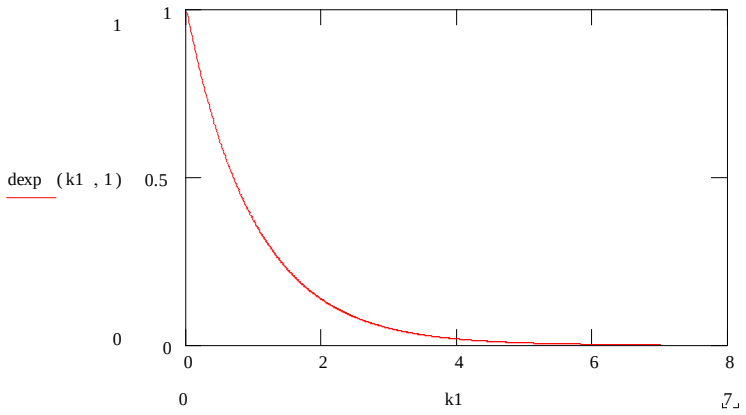
Distribution function: No closed form.

Application: In the absence of data, the beta distribution is used to model a random quantity when the minimum, mode, and maximum can be estimated. If available, an estimate of the mean can be used as well or the mean can be computed from the minimum, mode, and maximum.

Traditionally, the beta distribution has been used to model the time to complete a project task.

When data are available, the beta can be used to model the fraction, 0 to 100%, of something that has a certain characteristic such as the fraction of scrap in a batch.

Figure 3-3: Summary of the Beta Distribution



Density Function Illustration

Parameter:	mean
Range:	[0, ∞)
Mean:	given parameter
Variance:	$mean^2$
Density function:	$f(x) = \frac{1}{mean} e^{-x/mean} \; ; x \geq 0$
Distribution function:	$F(x) = 1 - e^{-x/mean} \; ; x \geq 0$
Application:	The exponential is used to model quantities with high variability such as entity inter-arrival times and the time between equipment failures as well as operation times with high variability. In the absence of data, the exponential distribution is used to model a random quantity characterized only by the mean.

Figure 3-4: Summary of the Exponential Distribution

3.2.2 Selecting Probability Distributions in the Absence of Data – An Illustration

Consider the operation time for a single workstation. Suppose the estimates of a mode of 7 seconds, a minimum of 5 seconds, and a maximum of 13 seconds were accepted by the project team. Either of two distributions could be selected.

1. A triangular with the given parameter values and having a squared coefficient of variation[1] of 0.042.

2. A beta distribution with parameter values α_1 = 1.25 and α_2 = 1.75 and a squared coefficient of variation of 0.061 where equations 3-3 and 3-4 were use to compute α_1 and α_2.

The mean of the beta distribution was estimated as the arithmetic average of the minimum, maximum, and mode. Thus, the mean of the triangular distribution and of the beta distribution are the same.

Note that the choice of distribution could significantly affect the simulation results since the squared coefficient of variation of the beta distribution is about 150% of that of the triangular distribution. This means the average time in the buffer at workstation A will likely be longer if the beta distribution is used instead of the triangular. This idea will be discussed further in Chapter 5.

Figure 3-5 shows the density functions of these two distributions.

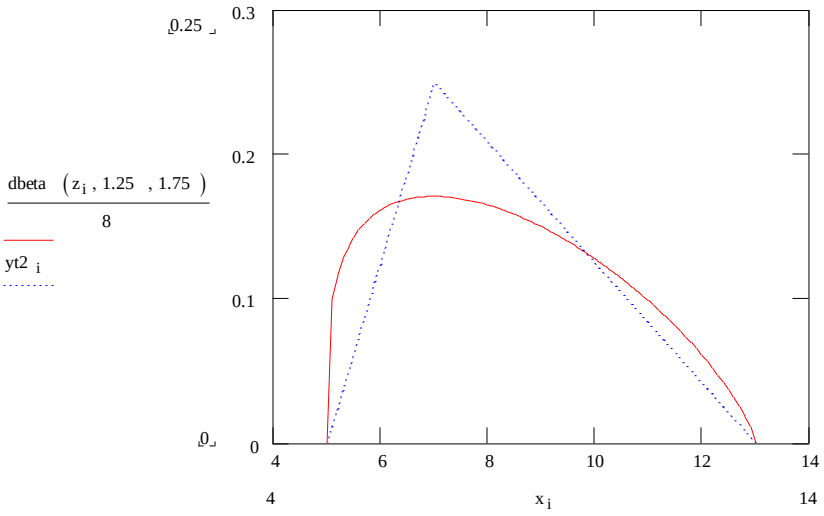


Figure 3-5: Probability Density Functions of the Triangular (5, 7, 13) and Beta (1.25, 1.75)

A word of caution is in order. If there is no compelling reason to choose the triangular or the beta distribution then a conservative course of action would be to run the simulation first using one distribution and then the other. If there is no significant difference in the simulation results or at least in the conclusions of the study, then no further action is needed. If the difference in the results is significant, both operationally and statistically, further information and data about the random quantity being model should be collected and studied.

[1] The coefficient of variation is the standard deviation divided by the mean. The smaller this quantity the better.

Furthermore, it was estimated that there would be 14400 arrivals per 40-hour week to the two workstations in a series system. Thus, the average time between arrivals is 40 hours / 14400 arrivals = 10 seconds. The time between arrivals was modeled using an exponential distribution with mean 10 seconds.

3.3 Fitting a Distribution Function to Data

This section discusses the use of data in determining the distribution function to use to model a random quantity as well as values for the distribution parameters. Some common difficulties in obtaining and using data are discussed. The common distribution functions used in simulation models are given. Law (2007) provide an in depth discussion of this topic, including additional distribution functions. A software based procedure for using data in selecting a distribution function is presented.

3.3.1 Some Common Data Problems

It is easy to assume that data is plentiful and readily available in a corporate information system. However, this is often not the case. Some problems with obtaining and using data are discussed.

1. Data are available in the corporate information system but no one on the project team has permission to access the data.

   Typically, this problem is resolved by obtaining the necessary permission. However, it may not be possible to obtain this permission in a timely fashion. In this, case the procedures for determining a distribution function in the absence of data should be used at least initially until data can be obtained.

2. Data are available but must be transformed into values that measure the quantity of interest.

   For example, suppose the truck shipment time between a plant and a customer is of interest. The company information system records the following clock times for each truck trip: departure from the plant, arrival to the customer, departure from the customer, and arrival to the plant. The following values can be computed straightforwardly from this information for each truck trip: travel time from the plant to the customer, time delay at the customer, travel time from the customer to the plant.

   This example raises some other questions. Is the there any reason to believe that the travel time from the plant to the customer is different from the travel time from the customer to the plant? If not, the two sets of values could be combined and a single distribution could be determined from all the values. If there is a known reason that the travel times are different, the two data sets must be analyzed separately. Of course, a statistical analysis, such as the paired-t method discussed in chapter 4, could be used to assess whether any statistically significant difference in the mean travel times exists.

   What is the level of detail included in the model? It may be necessary to include all three times listed in the previous paragraph in the model. Alternatively, only the total round trip time, the difference between the departure from the plant and the arrival to the plant, could be included.

3. All the needed data is available, but only from multiple sources.

   Each of the multiple sources may measure quantities in different ways or at different times. Thus, data from different sources need to be made consistent with each other. This is discussed by Standridge, Pritkser, and Delcher (1978).

For example, the amount of sales of a chemical product is measured in pounds of product in the sales information system and in volume of product in the shipping information system. The model must measure the amount of product in either pounds or volume. Suppose pounds were chosen. The data in the shipping information system could be used after dividing it by product density (pounds/gallon).

Consider another example. A sales forecast is used to establish the average volume of demand for a product used in a model. The sales forecast for the product is a single value. A distribution of product demand is needed. A distribution is determined using historical sales data. The sales forecast is used as the mean of a distribution instead of the mean computed from historical data. This assumes that only the mean will change in the future. The other distribution parameters such as the variance as well as the particular distribution family, normal for example, will remain the same.

4. All data are “dirty”.

It is tempting to assume that data from a computer information system can be used without further examination or processing. This is often not the case. Many data collection mechanisms do not take into account the anomalies that occur in day-to-day system operations.

For example, an automated system records the volume of a liquid product produced each day. This production volume is modeled as a single random quantity. The recorded production volume for all days is greater than zero. However, on a few days it is two orders of magnitude less than the rest of the days. It was determined that these low volumes meant that the plant was down for the day. Thus, the production volume was modeled by a distribution function for the days that the plant was operating and zero for the remaining days. Each day in the simulation model, a random choice was made as to whether or not the plant was operating that day. The probability the plant was operating was estimated from the data set as percent of days operating / total number of days.

3.3.2 Distribution Functions Most Often Used in a Simulation Model

In this section, the distribution functions most often used in simulation models are presented. The typical use of each distribution is described. A summary of each distribution is given.

In section 3.2.1, distribution functions used in the absence of data were presented. The uniform and triangular distributions are typically only used in this case. The beta distribution is used as well. The beta is also useful modeling project task times.

In addition, the use of the exponential distribution to model the time between entity arrivals was discussed. Again, the conditions for using the exponential distribution are: there is one arrival at a time, the numbers of arrivals in disjoint time intervals are independent, and the average time until the next arrival doesn’t change over the simulation time period. In some cases, the latter assumption is not true. One way of handling this situation is illustrated in the application study concerning contact center management.

If the system exerts some control over arrivals, this information may be incorporated in the simulation. For example, arrivals of part blanks to a manufacturing system could occur each hour on the hour. The time between arrivals would be a constant 1 hour. Suppose that workers have noted that the blanks actually arrive anywhere between 5 minutes before and 5 minutes after the hour. Thus, the arrival process could be modeled as with a constant time between arrivals of 1 hour followed by a uniformly distributed delay of 0 to 10 minutes before processing begins.

Often it is important to include the failure of equipment in a simulation model. Models of the time till failure can be taken from reliability theory. The exponential distribution may also be used to

model the time until the next equipment breakdown if the proper conditions are met: there is one breakdown at a time (for each piece of equipment), the number of breakdowns in disjoint time intervals are independent, and the average time until the next breakdown doesn't change over the simulation time period.

Suppose either of the following is true:

1. The time from now till failure does depend on how long the equipment has been functioning.
2. Failure occurs when the first of many components or failure points fails.

Under these conditions, the Weibull distribution is an appropriate model of the time between failures. The Weibull is also used to model operation times. A Weibull distribution has a lower bound of zero and extends to positive infinity.

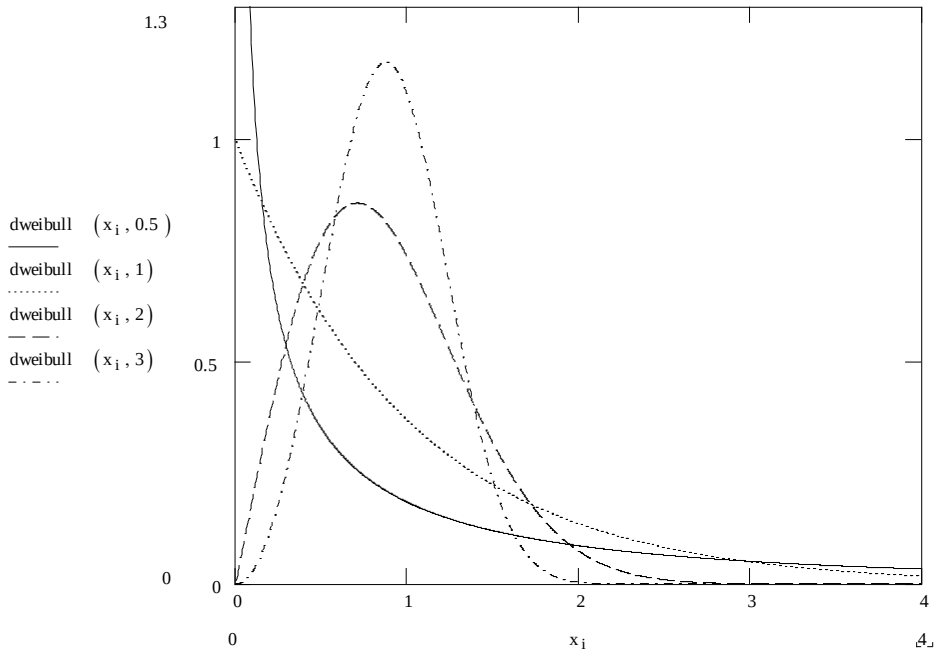
A Weibull distribution has two parameters: a shape parameter $\alpha > 0$ and a scale parameter $\beta > 0$. Note that the exponential distribution is a special case of the Weibull distribution for $\alpha = 1$. A summary of the Weibull distribution is given in Figure 3-6.

Suppose failure is due to a process of degradation and a mathematical requirement that the degradation at any point in time is a small random proportion of the degradation to that point in time is acceptable. In this case the lognormal distribution is appropriate. The lognormal has been successfully applied in modeling the time till failure in chemical processes and with some types of crack growth. It is also useful in modeling operation times.

The lognormal distribution can be thought of in the following way. If the random variable X follows the lognormal distribution then the random variable ln X follows the normal distribution. The lognormal distribution parameters are the mean and standard deviation of the normal distribution results from this operation. A lognormal distribution ranges from 0 to positive infinity. The lognormal distribution is summarized in Figure 3-7.

Consider operation, inspection, repair and transportation times. In modeling automated activities, these times may be constant. A constant time also could be appropriate if a standard time were assigned to a task. If human effort is involved, some variability usually should be included and thus a distribution function should be employed.

The Weibull and lognormal are possibilities as mentioned above. The gamma could be employed as well. A gamma distribution has two parameters: a shape parameter $\alpha > 0$ and a scale parameter $\beta > 0$. It is one of the most general and flexible ways to model a time delay. Note that the exponential distribution is a special case of the gamma distribution for $\alpha = 1$.



Density Function Illustrations

Parameters: Shape parameter, $\alpha>0$, and a scale parameter, $\beta>0$.

Range: $[0, \infty)$

Mean: $\frac{\beta}{\alpha}\Gamma(\frac{1}{\alpha})$, where Γ is the gamma function.

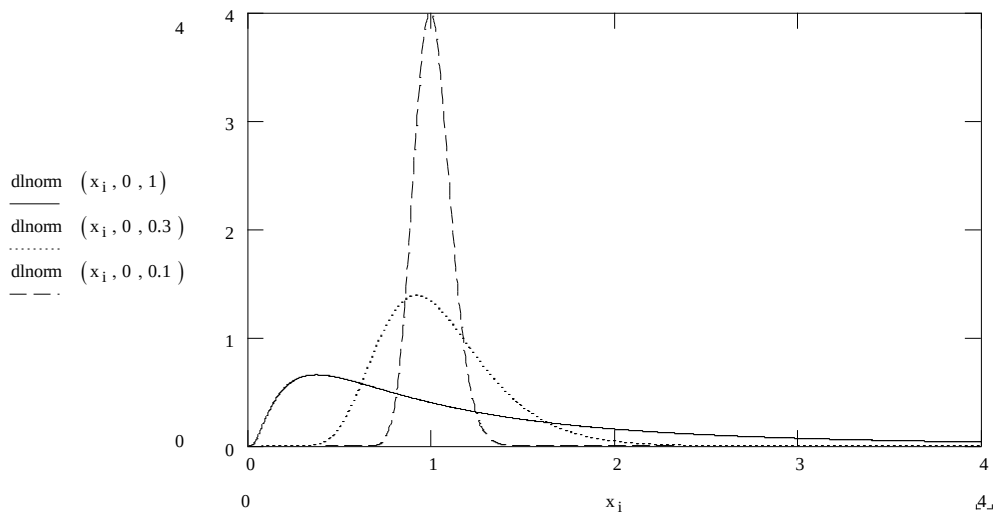
Variance: $\frac{\beta^2}{\alpha}\left\{2\Gamma\left(\frac{2}{\alpha}\right)-\frac{1}{\alpha}\left[\Gamma\left(\frac{1}{\alpha}\right)\right]^2\right\}$

Density function: $f(x) = \alpha\beta^{-\alpha}x^{\alpha-1}e^{-(x/\beta)^{\alpha}};\ x \geq 0$

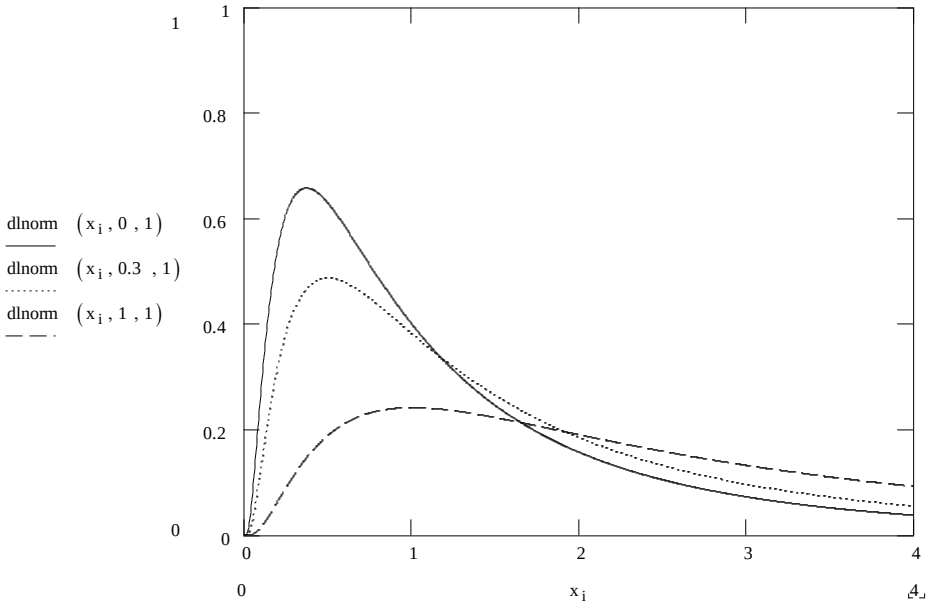
Distribution function: $F(x) = 1 - e^{-(x/\beta)^{\alpha}};\ x \geq 0$

Application: The Weibull distribution is used to model the time between equipment failures as well as operation times.

Figure 3-6: Summary of the Weibull Distribution.



Constant Mean



Constant Variance

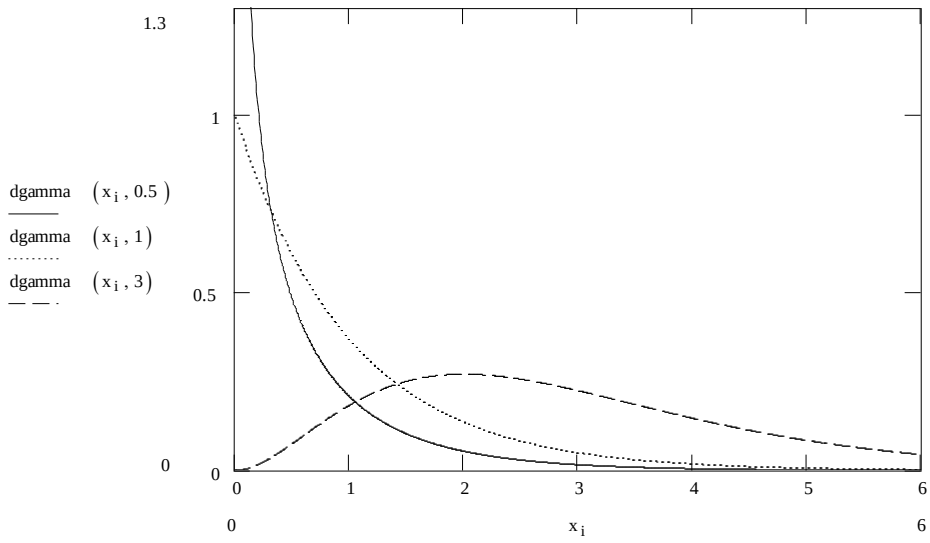
Density Function Illustrations

Figure 3-7: Summary of the Lognormal Distribution.

Parameters:	mean (μ) and standard deviation (σ) of the normal distribution that results from taking the natural logarithm of the lognormal distribution
Range:	$[0, \infty)$
Mean:	$e^{\mu + o^2/2}$
Variance:	$e^{2\mu + o^2}\left(e^{o^2} - 1\right)$
Density function:	$f(x) = \frac{1}{x\sqrt{2\pi o^2}} e^{\frac{-(\ln(x)-\mu)^2}{2\sigma^2}}; x > 0$
Distribution function:	No closed form
Application:	The lognormal distribution is used to model the time between equipment failures as well as operation times. By the central limit theorems, the lognormal distribution can be used to model quantities that are the products of a large number of other quantities.

Figure 3-7: Summary of the Lognormal Distribution, concluded.

The gamma distribution is summarized in Figure 3-8.



Density Function Illustrations

Parameters: Shape parameter, $\alpha>0$, and a scale parameter, $\beta>0$.

Range: $[0, \infty)$

Mean: $\alpha * \beta$

Variance: $\alpha * \beta^2$

Density function: $$f(x) = \frac{\beta^{-\alpha} x^{\alpha-1} e^{-(x/\beta)}}{\Gamma(\alpha)}; x > 0$$

Distribution function: No closed form, except when α is a positive integer.

Application: The gamma distribution is the most flexible and general distribution for modeling operation times.

Figure 3-8: Summary of the Gamma Distribution

It is often argued that the simulation experiment should include the possibility of long operation, inspection, and transportation times. A single such time can have a noticeable effect on system operation since following entities wait for occupied resources. In this case, a Weibull, lognormal, or gamma distribution can be used since each extends to positive infinity.

A counter argument to the use of long delay times is that they represent special cause variation. Often special cause variation is not considered during the initial phases of system design and thus would not be included in the simulation experiment. The design phase often considers only the nominal dynamics of the system.

Controls are often used during system operation to adjust to long delay times. For example, a part requiring a long processing time may be out of specification and discarded after a pre-specified amount of processing is performed. Such controls can be included in simulation models if desired.

The normal distribution, by virtue of central limit theorems (Law, 2007), is useful in representing quantities that are the sum of a large number (25 to 30 at least) of other quantities. For example, a sales region consists of 100 convenience stores. Demand for a particular product in that region is the sum of the demands at each store. The regional demand is modeled as normally distributed. This idea is illustrated in the application study on automated inventory management.

A single operation can be used to model multiple tasks. In this case, the operation time represents the sum of the times to perform each task. If enough tasks are involved, the operation time can be modeled using the normal distribution.

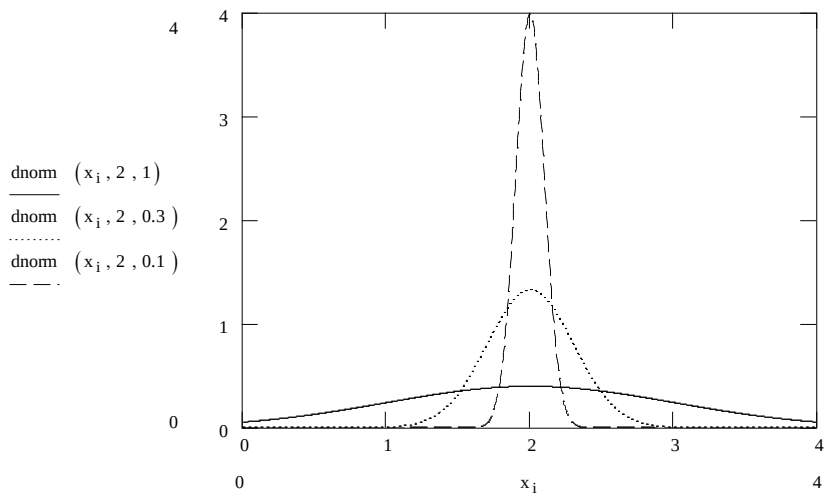
The parameters of a normal distribution function are the mean (μ) and the standard deviation (σ). Figure 3-8 shows several normal distribution density functions and summarizes the normal distribution.

Some quantities have to do with the number of something, such as the number of parts in a batch, the number of items a customer demands from inventory or the number of customers arriving between noon and 1:00 P.M. Such quantities can be modeled using the Poisson distribution.

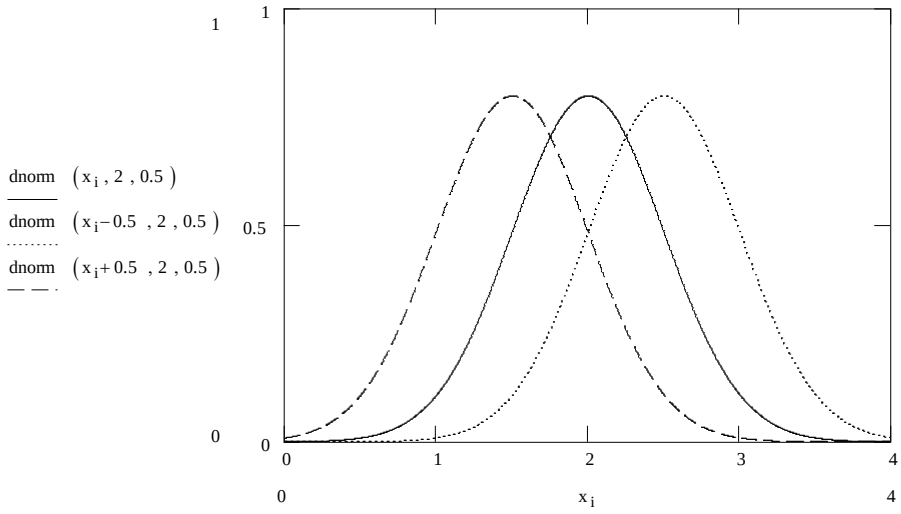
Unlike the distributions previously discussed, the range of the Poisson distribution is only non-negative integer values. Thus, the Poisson is a discrete distribution. The Poisson has only one parameter, the mean.

Note that if the Poisson distribution is used to model the number of events in a time interval, such as the number of customers arriving between noon and 1:00 P.M., that the time between the events, arrivals, is exponentially distributed. In addition, the normal distribution can be used as an approximation to the Poisson distribution. The Poisson distribution is summarized in Figure 3-9.

Some quantities can take one of a small number of values, each with a given probability. For example, a part is of type “1” with 70% probability and of type “2” with 30% probability. In these cases, the probability mass function is simply enumerated, e.g. $p_1 = 0.70$ and $p_2 = 0.30$. The enumerated probability mass function is summarized in Figure 3-10.



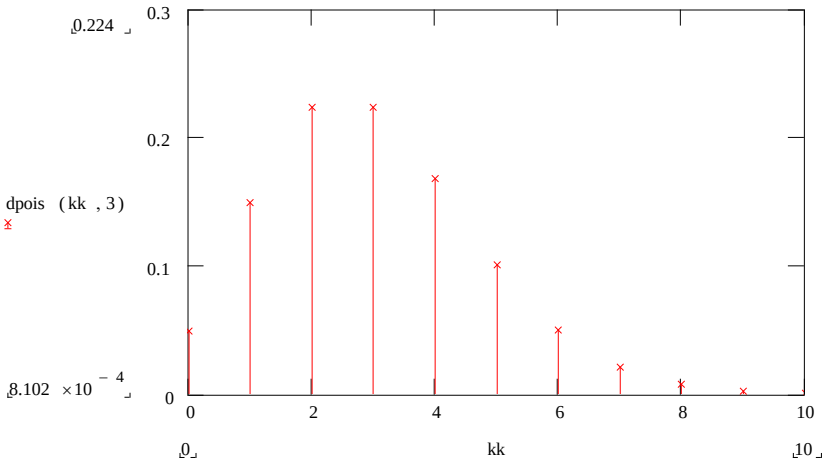
Constant Mean



Constant Variance
Density Function Illustrations

Parameters:	mean (μ) and standard deviation (σ)
Range:	(-∞, ∞)
Mean:	μ
Variance:	σ^2
Density function:	$f(x) = \frac{1}{\sqrt{2\pi o^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$
Distribution function:	No closed form
Application:	By the central limit theorems, the normal distribution can be used to model quantities that are the sum of a large number of other quantities.

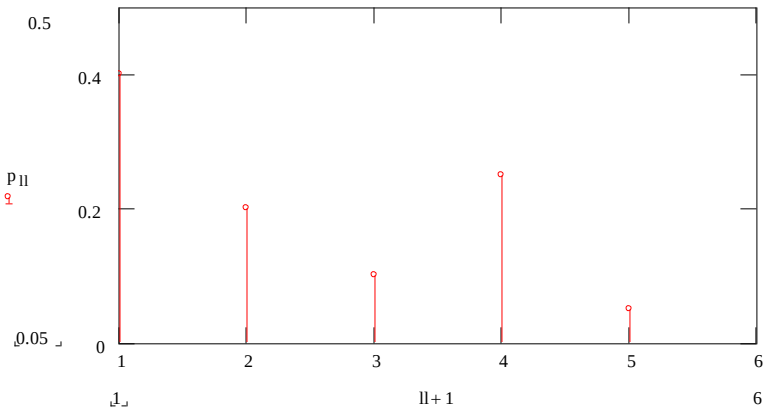
Figure 3-8: Summary of the Normal Distribution.



Density Function Illustration

Parameter:	mean
Range:	Non-negative integers
Mean:	given parameter
Variance:	mean
Mass function:	$p(x) = \frac{e^{-mean} * mean^{x}}{x!}$; x is a non - negative integer
Distribution function:	$F(x) = e^{-mean} * \sum_{i=0}^{x} \frac{mean^{i}}{i!}$; x is a non - negative integer
Application:	The Poisson distribution is used to model quantities that represent the number of things such as the number of items in a batch, the number of items demanded by a single customer, or the number of arrivals in a certain time period.

Figure 3-9: Summary of the Poisson Distribution



Density Function Illustration

Parameter: set of value-probability pairs (x_i, p_i), number of pairs, n

Range: [minimum x_i , maximum x_i]

Mean: $\sum_{i=1}^{n} p_i * x_i$

Variance: $\sum_{i=1}^{n} p_i * (x_i - mean)^2$

Mass function: $p(x_i) = p_i$

Distribution function: $F(x_i) = \sum_{k=1}^{i} p_k$

Application: An enumerated probability mass function is used to model quantities that represent the number of things such as the number of items in a batch and the number of items demanded by a single customer where the probability of each number of items is known and the number of possible values is small.

Figure 3-10: Summary of the Enumerated Probability Mass Function

Law and McComas (1996) estimate that "perhaps one third of all data sets are not well represented by a standard distribution." In this case, two options exist:

1. Form an empirical distribution function from the data set.
2. Fit a generalized functional form to the data set that has the capability of representing an unlimited number of shapes.

The former can be accomplished by using the frequency histogram of a data set to model a random quantity. The disadvantages of this approach are that the simulation considers only values within the range of the data set and in proportion to the cells that comprise the histogram.

One way to accomplish the latter is by fitting a Bezier function to the data set using an interactive Windows-based computer program as described by Flannigan Wagner and Wilson (1995, 1996).

3.3.3 A Software Based Approach to Fitting a Data Set to a Distribution Function

This section discusses the use of computer software in fitting a distribution function to data. Software should always be used for this purpose and several software packages support this task. The following three activities need to be performed.

1. Selecting the distribution family or families of interest.
2. Estimating the parameters of particular distributions.
3. Determining how well each distribution fits the data.

The distribution functions discussed in the preceding sections, beta or normal for example, are called families. An individual distribution is specified by estimating values for its parameters. There are two possibilities for selecting one or more distribution function families as candidates for modeling a random quantity.

1. Make the selection based on the correspondence between the situation being modeled and the theoretical properties of the distribution family as presented in the previous sections.

   For example, a large client buys a particular product from a supplier. The client supplies numerous stores from each purchase. The time between purchases is a random variable. Based on the theoretical properties of the distributions previously discussed, the time between orders could be modeled as using an exponential distribution and the number of units of product purchased could be modeled using a normal distribution.

2. Make the selection based on the correspondence between summary statistics and plots, such as a histogram, and particular density functions. Software packages such as ExpertFit [Law and McComas 1996, 2001] automatically compute and compare, using a relative measure of fit, candidate probability distributions and their parameters. In ExpertFit, the relative measure of fit is based on a proprietary algorithm that includes statistical methods and heuristics.

   For example, 100 observations of an operation time are collected. A histogram is constructed of this data. The mean and standard deviation are computed. Figure 3-11 shows the histogram on the same graph as a lognormal distribution and a gamma distribution whose mean and standard deviation were estimated from the data set. Note that the gamma distribution (dashed line) seems to fit the data much better than the lognormal distribution (dotted line).

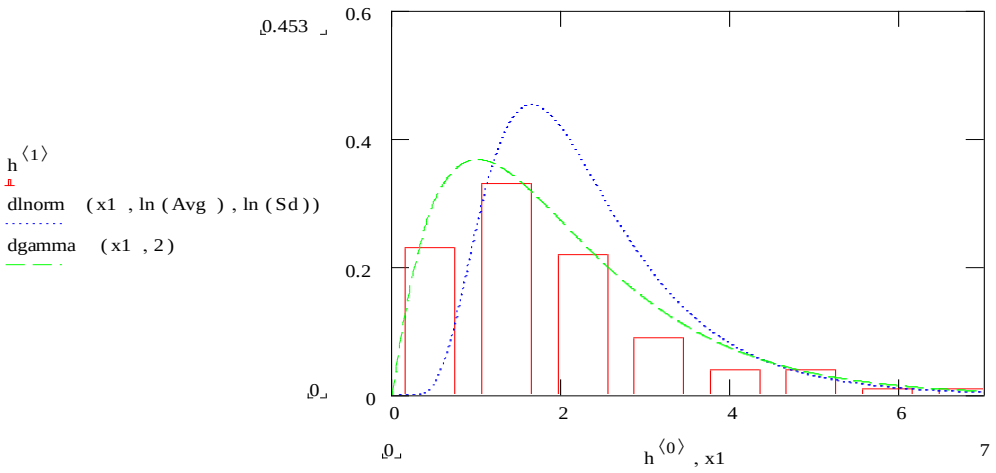


Figure 3-11: Comparison of a Histogram with Gamma and Lognormal Density Functions

For some distributions, the estimation of parameters values is straightforward. For example, the parameters of the normal distribution are the mean and standard deviation that are estimated by the sample mean and sample standard deviation computed from the available data. For other distributions, the estimation of parameters is complex and may require advanced statistical methods. For example, see the discussion of the estimation procedure for the gamma distribution parameters in Law (2007). Fortunately, these methods are implemented in distribution function fitting software.

The third activity is to assess how well each candidate distribution represents the data and then choose the distribution that provides the best fit. This is called determining the "goodness-of-fit". The modeler uses statistical tests assessing goodness of fit, relative and absolute heuristic measures of fit, and subjective judgment based on interactive graphical displays to select a distribution from among several candidates.

Heuristic procedures include the following:

1. Density/Histogram over plots – Plot the histogram of the data set and a candidate distribution function on the same graph as in Figure 3-11. Visually check the correspondence of the density function to the histogram.

2. Frequency comparisons – Compare the frequency histogram of the data with the probability computed from the candidate distribution of being in each cell of the histogram.

   For example, Figure 3-12 shows a frequency comparison plot that displays the sample data set whose histogram is shown in Figure 3-11 as well as the lognormal distribution whose mean and standard deviation were estimated from the data set. Differences between the lognormal distribution (solid bars) and the data set (non-solid bars) are easily seen.

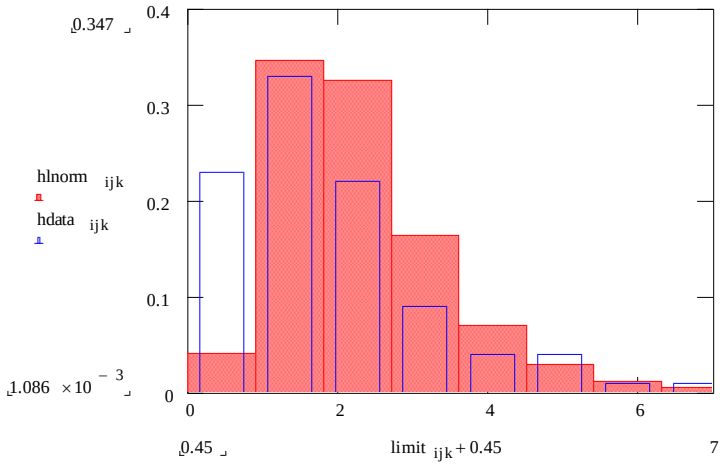


Figure 3-12: Frequency Comparison of a Data Set with a Lognormal Distribution

3. Distribution function difference plots – Plot the difference of the cumulative candidate distribution and the fraction of data values that are less than x for each x-axis value in the plot. The closer the plot tracks the 0 line on the vertical axis the better.

   For example, Figure 3-13 shows a distribution function difference plot comparing the sample data set whose histogram is displayed in Figure 3-11 to both the gamma and lognormal distributions whose mean and standard deviations were estimated from the data. The gamma distribution (solid line) appears to fit the data set much more closely that the lognormal distribution (dotted line).

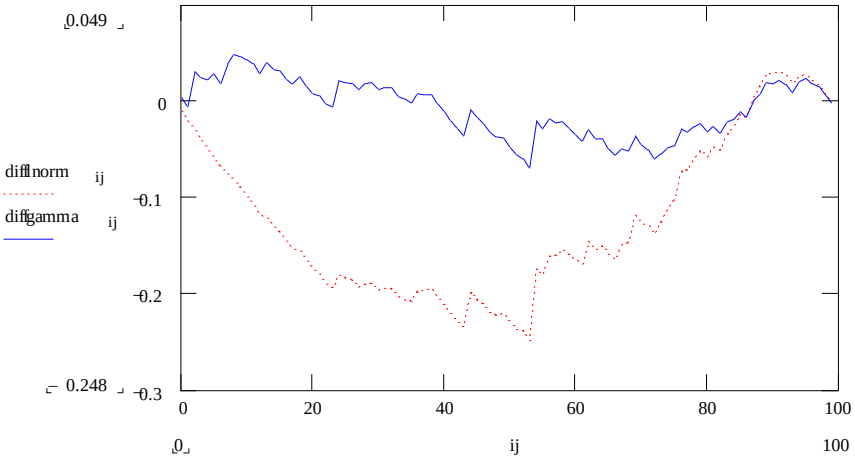


Figure 3-13: Distribution Function Difference Plot Comparison of a Data Set with a Gamma and a Lognormal Distribution

4. Probability plots – Use one of the many types of probability plots to compare the data set and the candidate distribution. One such type is as follows. Suppose there are n values in the data set. The following points, n in number, are plotted: (i / n th percent point of the candidate distribution, the i th smallest value in the data set). These points when

plotted should follow a 45 degree line. Any substantial deviation from this line indicates that the candidate distribution may not fit the data set.

For example, Figure 3-14 shows a probability plot that compares the sample data set whose histogram is displayed in Figure 3-11 to both the gamma and lognormal distributions shown in the same figure. Note that the gamma distribution (solid line) tracks the 45 degree line better than does the lognormal distribution (dotted line) and both deviate from the line more toward the right tail.

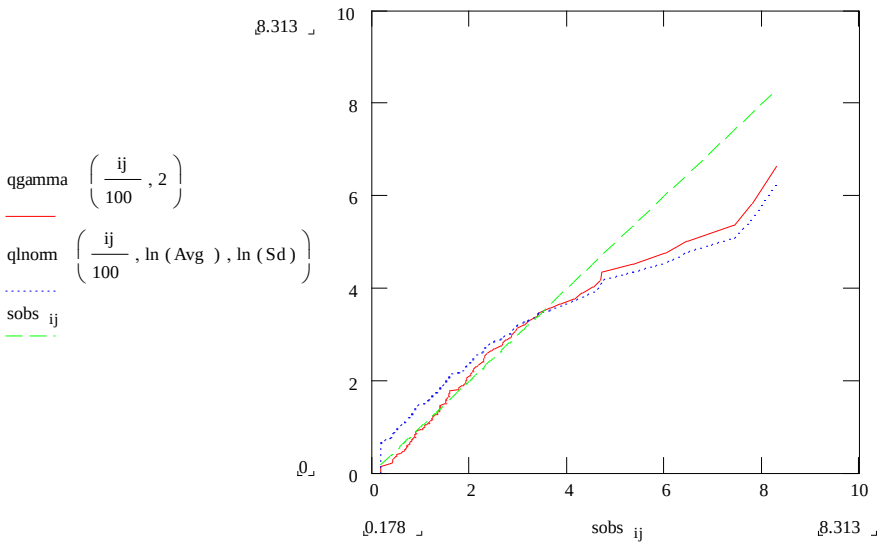


Figure 3-14: Probability Plot Comparison of a Data Set with Gamma and Lognormal Distributions

Statistical tests formally assess whether the data set that consists of independent samples is consistent with a candidate distribution. These tests provide a systematic approach for detecting relatively large differences between a data set and a candidate distribution. If no such differences are found, the best that can be said is that there is no evidence that the candidate distribution does not fit the data set.

The behavior of these tests depends on the number of values in the data set. For large values of n, the tests seem to always detect a significant difference between a candidate distribution and a data set. For smaller values of n, the tests detect only gross differences. This should be kept in mind when interpreting the results of the test.

The following tests are common and are typically performed by distribution function fitting software.

1. Chi-square test – formally compares a histogram of the data set with a candidate distribution as was done visually using a frequency comparison plot.

2. Kolmogorov-Smironv (K-S) test – formally compares an empirical distribution function constructed from the data set with a candidate cumulative distribution, which is analogous to the distribution function difference plot.

3. Anderson-Darling test – formally compares an empirical distribution function constructed from the data set with a candidate cumulative distribution function but is better at detecting differences in the tails of the distribution than the K-S test.

3.4 Summary

This chapter discusses how to determine the distribution function to use in modeling a random quantity. How this choice can affect the results of a simulation study has been illustrated. Some issues with obtaining and using data have been discussed. Selecting a distribution both using a data set and in the absence of data has been presented.

Problems

1. List the distributions that have a lower bound.

2. List the distributions that have an upper bound.

3. List the distributions that are continuous.

4. List the distributions that are discrete.

5. Suppose X is a random variable that follows a beta distribution with range [0,1]. A random variable, Y, is needed that follows a beta distribution with range [10, 100]. Give an equation for Y as a function of X.

6. Suppose data are not available when a simulation project starts.

   a. What three parameters are commonly estimated without data?

   b. An operation time is specified giving only two parameters: minimum and maximum. However, it is to be modeled using a triangular distribution. What would you do?

7. Consider the following data set: 1, 2, 2, 3, 4, 5, 7, 8, 9, 10, 11, 13, 15, 16, 17, 17, 18, 18, 18, 20, 20, 21, 21, 24, 27, 29, 30, 37, 40, 40. What distribution family appears to fit the data best? Use summary statistics and a histogram to assist you.

8. Hypothesize one or more families of distributions for each of the following cases:

   a. Time between customers arriving at a fast food restaurant during the evening dinner hour.
   b. The time till the next failure of a machine whose failure rate is constant.
   c. The time till the next failure of a machine whose failure rate increases in time.
   d. The time to manually load a truck based on the operational design of a system. You ask the system designers for the minimum, average, and maximum times.
   e. The time to perform a task with long task times possible.
   f. The distribution of job types in a shop.
   g. The number of items each customer demands.

9. What distribution function family appears to fit the following data set best? Use summary statistics and a histogram to assist you. Test your selection using the plots discussed in section 3.3.2.

8.39	3.49	3.17	15.34	4.68	4.38	0.02	1.21
3.56	0.50	4.38	2.53	20.61	2.78	2.66	32.88
22.49	5.10	4.58	3.07	22.64	34.86	9.59	0.67
12.24	3.25	34.07	5.43	14.72	5.84	15.37	21.20
0.21	3.20	25.12	3.18	3.60	11.45	1.07	8.69
0.46	9.16	10.71	3.75	1.54	0.65	3.68	10.46
20.11	5.81	4.63	3.13	8.99	2.82	0.87	13.45
10.10	12.57	22.67	3.55	5.68	29.07	0.62	25.23
17.97	35.76	17.05	4.61	12.36	14.02	24.33	11.05
1.10	4.56	9.51	7.31	23.33	5.81	3.48	3.23

10. What distribution function family appears to fit the following data set best? Use summary statistics and a histogram to assist you. Test your selection using the plots discussed in section 3.3.2.

2373	2361	2390	2377	2333
2327	2380	2373	2360	2382

11. Use the distribution function fitting software to solve problems 7, 9, and 10.

Chapter 4

Conducting Simulation Experiments

4.1 Introduction

This chapter provides the information necessary to design, carry out, and analyze the results of a **simulation experiment**. Experimentation with a simulation model, as opposed to an exact analytic solution obtained using mathematics, is required. Principle 2 states that simulation models conform both to system structure and to available system data. Conditional logic is employed. Thus, these models usually cannot be solved by analytic methods. Simulation experiments must be properly designed and conducted as would any field or laboratory experiment. The design of simulation experiments leads to the benefits of simulation described by principle 3: lower cost and more flexibility than physical prototypes as well as less risk of negative consequence on day-to-day operations than direct experimentation with existing, operating systems as the plan-do-check-act (PDCA) cycle of lean would do.

The design of a simulation experiment specifies how model processing generates the information needed to address the issues and to meet the solution objectives identified in the first phase of the simulation process. An approach to the analysis of results is presented, including ways of examining simulation results to help understand system behavior as well as the use of statistical methods such as confidence interval estimation to help obtain evidence about performance, including the comparison of scenarios.

Prerequisite issues to the design and analysis of any simulation experiment are discussed. These include verification and validation as well as the need to construct independent observations of simulation performance measures and to distinguish between probability and degree of confidence.

Verification and validation are discussed first followed by a discussion of the need to construct independent observations of performance measures. The design elements of simulation experiments are explained. Finally, an approach to the analysis of terminating simulation results is given along with an explanation of how probability and degree of confidence are differentiated.

The discussion in this chapter assumes some prior knowledge of data summarization, probability, and confidence interval estimation.

4.2 Verification and Validation

This section discusses the verification and validation of simulation models. Verification and validation, first described in principle 6 of chapter 1, have to do with building a high level of confidence among the members of a project team that the model can fulfill is objectives. Verification and validation are an important part of the simulation process particularly with respect to increasing model credibility among managers and system experts.

Verification has to do with developing confidence that the computer implementation of a model is in agreement with the conceptual model as discussed in chapter 1. In other works, the computer implementation of the model agrees with the specifications given in the conceptual model. Verification includes debugging the computer implementation of the model.

Validation has to do with developing confidence that the conceptual model and the implemented model represent the actual system with sufficient accuracy to support making decision about project issues and to meet the solution objectives. In other works, the computer implementation of the model and the conceptual model faithfully represent the actual system.

As described by many authors (Balci, 1994; Balci, 1996; Banks, Carson, Nelson, and Nicol, 2009; Carson, 2002; Law, 2007; Sargent, 2009), verification and validation require gathering evidence that the model and its computer implementation accurately represent the system under study with respect to project issues and solution objectives. Verification and validation are a matter of degree. As more evidence is obtained, the greater the degree of confidence that the model is verified and valid increases. It should be remember however that absolute confidence (100%) cannot be achieved. There will always be some doubt as to whether a model is verified and validated.

How to obtain verification and validation evidence and what evidence to obtain is case specific and requires knowledge of the problem and solution objectives. Some generally applicable strategies are discussed and illustrated in the following sections. The application studies, starting in chapter 6, provide additional examples. Application problems in the same chapters give students the opportunity to practice verification and validation.

Verification and validation strategies are presented separately for clarity of discussion. However in practice, verification and validation tasks often are intermixed with little effort to distinguish verification from validation. The focus of both verification and validation is on building confidence that the model can be used to meet the objectives of the project.

A pre-requisite to a proper simulation experiment is verifying and validating the model.

Throughout this chapter, including the discussion of verification and validation, illustrations and examples will make use of a model of two stations in a series with a large buffer between the stations as well as the industrial example presented in section 1.2. A diagram of the former is shown in Figure 4-1. A part enters the system, waits in the buffer of workstation A until the machine at this workstation is available. After processing at workstation A, a part moves to workstation B where it waits in the buffer until the workstation B machine is available. After processing at workstation B, a part leaves the system. Note that because it is large, the buffer between the stations is not modeled.

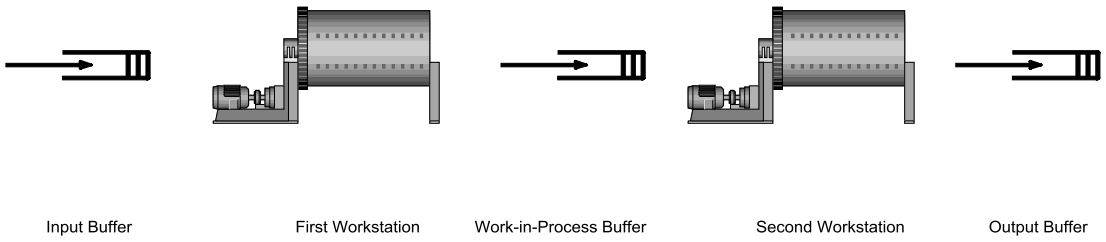


Figure 4-1: Example Two Workstations in Sequence, Repeated.

4.2.1 Verification Procedures

Some generally applicable techniques for looking for verification evidence follow.

1. What goes into a model must come out or be consumed.

For example, in the two workstations in a series model, the following “entity balance” equation should hold:

Number of entities entering the system = the number of entities departing the system + the number of entities still in the system at the end of the simulation

The latter quantity consists of the number of entities in each workstation buffer (A and B) plus the number of entities being processed at workstations A and B. If the entity balance equality is not true, there is likely an error in the model that should be found and corrected.

The number of entities entering the system consists of the number of entities initially there at the beginning of the simulation plus the number of entities arriving during the simulation.

For example, for one simulation of the two workstations in a series model, there were 14359 entities arriving to the model of which 6 were there initially. There were 14357 entities that departed and two entities in the system at the end of the simulation. One of the two entities was in the workstation A operation and the other was in the workstation B operation.

2. Compare the process steps of the computer model and the conceptual model.

The process steps in the model implemented in the computer version of the model and the conceptual model should correspond and any differences should be corrected or justified.

The process steps in the two workstations in a series model are as follows:

1. Arrive to the system.
2. Enter the input buffer of workstation A.
3. Be transformed by the workstation A operation.
4. Be moved to and enter the input buffer of workstation B.
5. Be transformed by the workstation B operation.
6. Depart the system.

3. Check all model parameter values input to the model.

The model implementation should include the checking required to assure that input parameter values are correctly input and used.

For example in the industrial application discussed in section 1.2, customer demand volume is input. The volume of product shipped is output. Enough information is included in the reports generated by the model to easily determine if all of the input volume is shipped or is awaiting shipment at the end of the simulation.

4. Check all entity creations.

The time between arrivals is specified as part of the model. The average number of arrivals can be computed given the ending time of the simulation. In addition, the number of arrivals during the simulation run is usually automatically reported. These two quantities can be compared to assure that entities are being created as was intended.

For example, suppose model of the two stations in a series was simulated for 40 hours with an average time between arrivals of 10 seconds. The expected number of arrivals would be 14400 (= 40 hours / 10 seconds). Suppose 20 independent simulations were made and the number of arrivals ranged from 14128 to 14722. Since this range includes 14400, verification evidence would be obtained. How to do the independent simulations is discussed in section 4.3 and following.

Alternatively a confidence interval for the true mean number of arrivals could be computed. If this confidence interval includes the expected number of arrivals verification evidence is obtained. In the same example, the 95% confidence interval for the mean number of arrivals is 14319 to 14457. Again, verification evidence is obtained.

5. Check the results of all logical decisions.

Sufficient checking should be built into the simulation model to assure that all logical decisions are correctly made that is all conditional logic is correctly implemented.

For example in the industrial problem discussed in section 1.2, each product could be shipped from one of a specified set of loads spots. Output reports showed the volume of shipments by product and load spot combination. Thus, it could be easily seen if a product was shipped from the wrong load spot.

6. Implement the simplest possible version of the model first and verify it. Add additional capabilities to the model one at a time. Perform verification after each capability is added.

Verifying that any complex computer program was implemented as intended can be difficult. Implementing the smallest possible model helps simplify the verification task, and perhaps more importantly, results in a running model in relatively little time. Verifying one capability added to an already verified model is relatively straightforward.

For example, the model of the industrial problem presented in section 1.2, has been developed over a number of years with new capabilities added to support addressing new issues and solution objectives.

7. For models developed by multiple individuals, used structured walk-throughs.

Each individual implements an assigned portion of the model, or sub-model. Each individual presents the implementation to all of the other team members. The team as a whole must agree that the implementation faithfully represents the conceptual model.

For example, one strategy is to build and implement an initial model as quickly as possible from the specifications in the conceptual model. If the conceptual model is incomplete, assumptions are made to complete model construction and implementation. The assumptions may be gross or inaccurate. The entire team reviews the initial model, especially the assumptions, and compares it to the conceptual model. The assumptions are corrected as necessary. This may require team members to gather additional information about how certain aspects of the system under study work.

8. Use the model builders available in most simulation environments.

Model builders implement the standard modeling constructs available in a simulation language. They provide a standard structure for model building and help guard against model building errors such as inconsistent or incomplete specification of modeling constructs.

9. Output and examine voluminous simulation results.

Sufficient information should be output from the simulation to verify that the different components of the system are operating consistently with each other in the model.

For example in the industrial problem of section 1.2, both the utilization of each load spot and summary statistics concerning the time to load each product are reported. If the load spots assigned to a product have high utilization, the average product loading time should be relatively long.

10. Re-verify the model for each set of model parameter values tested.

A model implementation can be verified only with respect to the particular set of model parameter values tested. Each new set of parameter values requires re-verification. However after many sets of parameter values have been tested, confidence is gained that the implementation is correct for all sets of parameter values in the same range.

For example for the industrial problem of section 1.2, verification information is carefully examined after each simulation experiment.

4.2.2 Validation Procedures

Some generally applicable techniques for looking for validation evidence follow.

1. Compare simulation model results to those obtained from analytic models.

This is a restatement of principle 11 of chapter 1. For example, the mean number of busy units of a resource can be computed easily as discussed in chapter 6.

In the two workstations in a series model, suppose the operation time at the second workstation is a constant 8.5 seconds and the mean time between arrivals is 10 seconds. The percentage busy time for workstation B is equal to 8.5 / 10 seconds or 85%. The simulation of the workstation provides data from which to estimate the percent busy time. The range of workstation B utilization over multiple independent simulations is 83% to 87%. A confidence interval for the true mean utilization could be computed as well. The 95% confidence interval is 84.4 to 85.4. Thus, validation evidence is obtained.

2. Use animation to view simulation model dynamics, especially those involving complex conditional logic.

Reviewing all the implications of complex decisions using voluminous information in a static medium, such as a report, or even in an interactive debugger, is difficult and possibly overwhelming. Animation serves to condense and simplify the viewing of such information.

Consider the following illustration. In the early 1980's, a particular simulation company was developing its first commercial animator product. Having completed the implementation and testing, the development team asked an application consultant for an industrial model to animate. The consultant supplied a model that included a complex control system for a robot.

The developers completed the animation and presented it to the consultant. The response of the consultant was that there must be something wrong with the new animation software as the robot could not engage in the sequence of behavior displayed.

Try as they might, the development team could not find any software error in the animator. To aid them, the team asked the consultant to simulate the model, printing out all of the information about the robot's behavior. The error was found not in the animator, but in the model. The disallowed behavior pattern occurred in the simulation!

This is not a criticism of the consultant. Rather it points out how easy it was to see invalid behavior in an animation though it was infeasible to detect it through a careful examination of the model and output information.

3. Involve system experts and managers

System experts should review the model, parameter values and simulation results for consistency with system designs and expectations. Reports of simulation results should be presented in a format that is understandable to system experts without further explanation from the modelers. Animation can help in answering questions such as: How was the system represented in the model? Inconsistencies and unmet expectations must be resolved as either evidence of an invalid model or unexpected, but valid, system behavior.

For example the development process for the industrial model discussed in section 1.2 was as follows. A first cut model was developed as quickly after the start of the project as possible. It was clear during the development of this model that some components of the system had not been identified or had incomplete specifications that is the first draft conceptual model was incomplete. The modelers made gross assumptions about how these components operated. The first cut model was reviewed by the entire project team including system experts and managers. Based on this review, tasks for completing the conceptual model were assigned. When these tasked were completed, the conceptual model was updated and the implemented model was revised accordingly.

4. If some quantities are estimated by system experts and managers, test their effect on system outputs.

As discussed in chapter 3, there may be a lack of data available for estimating time delays or other quantities needed in a model. This is common when the simulation model is being used to assist in the design of a new system. For such quantities, it is essential to perform a sensitivity analysis. This involves running the model with a variety of values of each estimated quantity and observing the effect on performance measures. Estimated quantities that greatly effect system performance should be identified. Further study may be necessary to obtain a more precise estimate of their value.

For example, there was little data concerning shipping times, the time between when a product left the plant and when it arrived at a customer, for the industrial model discussed in section 1.2. These times were believed to be directly related to some of the key performance measures estimated by the model. Thus, it was thought to be wise to refine them over time. Initially, shipping times were specified as triangularly distributed with estimates of the minimum, maximum, and modal times for all products in general supplied by logistics experts. Later additional data was available so that shipment times were available for each group of products. Still later, an analysis of data in the corporate information system was done to provide product specific shipping times. The simulation model was modified to allow any of the three shipping time options to be used for any product.

5. Carefully review a trace of a simulation run.

A model specific report of the step-by-step actions taken during a run can be generated by the simulation in a format that can be read by system experts and managers. A careful examination of such a report, though tedious, can help assure that the process steps included in the simulation model are complete and correctly interact with each other.

For example, the sponsors of an industrial inventory management simulation required such a trace to assure that the model correctly captured the response of the actual system to certain disturbances. The trace was carefully examined by the sponsors and other system experts to gain confidence that the model was valid.

6. Compare performance measure values to system data to see if any operationally significant differences can be observed.

The same performance measures computed in the model may be estimated from data collected from an existing system. Summary statistics, such as the average, computed from performance measure values may be compared by inspection to summary statistics computed from the data collected from an existing system. If no operationally significant differences are observed, then validation evidence is obtained.

Law (2007) discusses the difficulty of using statistical tests to compare performance measure values and real world data as well as making some recommendations in this regard.

For example, in the industrial model of section 1.2, system experts believed that empty rail cars spent 6 to 7 days in the plant. Simulation results estimated that empty rail cars spent an average of 6.6 days in the plant. Thus, validation evidence was obtained.

4.3 The Problem of Correlated Observations

Most statistical analysis procedures require independent (and identically distributed) observations of performance measure values. However, the observations in a simulation experiment are typically dependent (correlated). This section illustrates why a simulation experiment generates correlated observations. Approaches to dealing with this issue are presented later in this chapter.

Consider the time the nth part arriving to workstation A in the two stations in a series model would spend at the workstation:

Time at workstation A_n = Time in buffer$_n$ + Operation time$_n$

The time in the buffer for the nth part is composed of the operation times for the parts that preceded it in processing while the nth part was in the buffer. For example, suppose the fourth part to arrive does so while the second part to arrive is being processed. So the time the fourth part spends in the buffer is equal to a portion of the operation time for the second part and the entire operation time for the third part:

Time at workstation$_4$ = f(operation time$_2$, operation time$_{3)}$ + Operation time$_4$

Thus, the time spent at the workstation by the fourth part is correlated with the time spent by the second and the third parts.

Rather than using correlated performance measure observations directly in statistical analysis computations, independent observations are constructed. How to do this is discussed later in this chapter.

The statistical analysis of simulation results is greatly aided by the construction of independent observations of the performance measures.

4.4 Common Design Elements

The elements common to all simulation experiments are discussed in the following sections. These include model parameters and their values, performance measures, and random number streams.

4.4.1 Model Parameters and Their Values

Model parameters correspond to system control variables or operational rules whose values can be changed to meet the solution objectives defined in the first step of the simulation process. Values of model parameters can be quantitative such as the size of a buffer or qualitative such as which routing policy to use.

Often in traditional experimental design and analysis, time and cost constraints result in the use of only two or three values of each model parameter. Simulation affords the opportunity to test as many values as time and computing resources allow. For example, various sizes of an inter-station buffer could be simulated. A very large size could represent an infinite buffer. A buffer size of one or two would be minimal. Intermediate buffer sizes such as five and ten could be evaluated.

Which values are used may depend on the results of preceding simulations. For example, results of the initial simulations may indicate that a buffer size in the range 10 to 20 is needed. Additional simulations would be run with for buffer sizes between 10 and 20.

Model parameters must be defined and their values specified.

4.4.2 Performance Measures

Performance measures are quantities used to evaluate system behavior. They are defined in accordance with principle 9 of chapter 1: “Simulation experimental results conform to unique system requirements for information.” Thus, each simulation experiment could have different performance measures.

Possible performance measures for experiments with the two stations in a series model could be as follows:

1. The number of items waiting in each buffer.
2. The percentage of time each workstation is busy.
3. The percentage of time each workstation is idle.
4. The time an item spends in the system (lead time).
5. The total number of items processed by the workstation.

Note that state variable values are used as performance measures along with the time taken by entities in one, more than one, or all of the processing steps. A count of the number of entities completing processing is desired as well. These kinds of performance measures are typical of many simulation experiments.

Performance measures must be defined, including how each is computed.

4.4.3 Streams of Random Samples

One purpose of a simulation experiment is comparing scenarios. Suppose that no statistically significant difference between two scenarios is found. This could occur because the scenarios do not cause distinct differences in system performance. A second and undesirable possibility is that the variance of the observations made during the simulation is too high to permit true differences in system observations to be confirmed statistically.

Suppose we wished to assess a change in the operation of workstation A in the two stations in a series model where the range of the operation time is reduced to uniformly distributed between 7 and 11 seconds from uniformly distributed between 5 and 13 seconds. The same arrivals could be used in simulating both scenarios. Thus, the comparison could be made with respect to the same set of entities processed by the workstation. In general, this approach is referred to as the method of **common random numbers** since simulations of the two scenarios have the same pattern of arrivals in common. Each time between arrivals was determined by taking a random sample from the exponential distribution modeling this quantity. How this is done will be discussed in the next chapter.

To better understand the effect of common random numbers, consider what happens when they are not used. There would be a different set of arrivals in the simulation of the first scenario than in the simulation of the second scenario. Observed differences in performance measure values between the two scenarios could be due to the differences in the arrivals or true differences between the scenarios. Thus, the variance associated with summary statistics of differences in values, such as the mean lead time, would likely be higher than if common random numbers were used. This higher variance might result in a failure to detect a true difference between the scenarios with respect to a given performance measure such as lead time even if such a difference existed.

The method of common random numbers requires distinct streams of samples for each quantity modeled by a probability distribution. While this does not guarantee a reduction in the variance of the difference, experience has shown that a reduction often occurs. In practice for most simulation languages, this means that the stream of samples associated with each quantity modeled by a probability distribution must be given a distinct name.

Law (2007) more details concerning the common random number approach as well as other experiment design techniques to control the variance. Banks, Carson, Nelson, and Nicol (2009) discuss these techniques as well.

The quantities modeled by probability distributions in a model must be identified and uniquely named the method of common random numbers may be employed.

4.5 Design Elements Specific to Terminating Simulation Experiments

A **terminating simulation experiment** ends at a specified simulation time or event that is derived from the characteristics of the system under study and is stated as a part of the experiment design. This is the distinguishing characteristic of such as an experiment.

This section presents the design elements that are specific to terminating simulation experiments. These include setting initial conditions, specifying the number of replications of the experiment, and specifying the ending time or event of the simulation.

4.5.1 Initial Conditions

To begin a simulation, the initial values of the state variables and the initial location in the model of any entities, along with their attribute values, must be specified. Together, these are called the **initial conditions**.

In a terminating simulation, the initial conditions should be the same as conditions that occur in the actual system (Law, 2007). The work of Wilson and Pritsker (1978) leads toward using the modal or, at least, frequently occurring conditions. This must be done to ensure there is not a statistically significant greater portion of performance measure values in any given range gathered from the simulation than would occur in the actual system. Thus, **statistical bias** is

collecting performance measure values that could not have occurred, or in greater (lesser) proportion in one range than could have occurred, in the actual system.

Consider again the two work stations in a series model. For example, suppose it is known that parts are almost always in the buffer of workstation A and of workstation B. Thus possible initial conditions are:

1. The workstation A resource is in the BUSY state processing one part.
2. The workstation B resource is in the BUSY state processing one part.
3. Two parts are in the buffer of workstation A.
4. Two parts are in the buffer of workstation B.

Note that the time spent at either workstation by a part will consist of the time spent in the input buffer plus the operation time. If the simulation experiment begins with no parts in either input buffer, the time the first part spends at each workstation is equal to the operation time because the time spent in the input buffer will be zero. The observed lead time for this part will be less than for any part processed by the actual system.

Statistical bias is illustrated in Figure 4-2 that shows example histograms of part time in the system collected from the actual system and from a simulation. The simulation has improper initial conditions of no parts at the workstation. Some of the simulation observations are biased low. Calculations of statistics based on statistically biased observations may also be biased and inaccurate conclusions about problem root causes or the performance of proposed solutions drawn.

The initial conditions must be specified as a part of the experimental design and must be actual conditions that occur in the system.

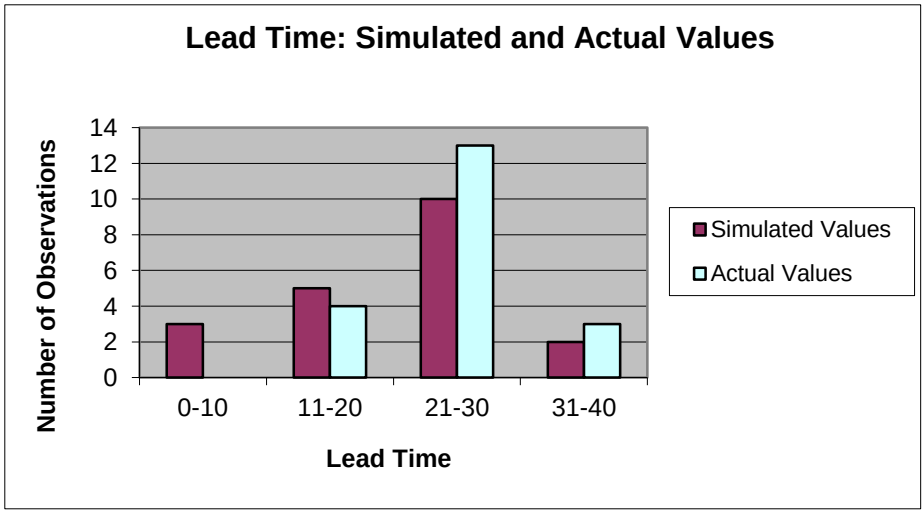


Figure 4-2: Illustration of Statistical Bias

4.5.2 Replicates

This section discusses the idea of replication to construct independent observations of simulation performance measures. **Replicates** of a simulation experiment differ from each other only in the sample values of the quantities modeled by probability distributions. Replicates are treated as independent of each other since the sample values exhibit no statistical correlation.

Each replicate is one possibility of how the random behavior of the system actually occurred. Multiple possibilities for system behavior should be examined in detail to draw conclusions about the effectiveness of system scenarios in meeting solution objectives.

Consider again the two work stations in a series model. There is a stream of sample values for the time between arrivals and another stream for operation times at workstation A. A replicate is defined by the particular samples taken in these two streams. Examining system performance for other streams of the time between arrivals and of service times is necessary. These other streams define additional replicates.

Observations of the same performance measure from different replicates are statistically independent of each other. In addition, performance measure observations from different replicates are identically distributed for the same reason. Thus replication is one way of constructing independent observations for statistical analysis. However, since performance measures may be arbitrarily defined, the underlying probability distribution of the performance measure observations cannot be determined in general.

During each replicate, one or more observations of the values of a performance measure are made. For example, the number of entities that complete processing in the two work stations in series model is incremented each time processing is finished, the lead time is recorded each time an entity completes processing, and the number of entities in either workstation buffer is updated each time an entity arrives at a workstation as well as each time an entity begins processing.

For the reasons discussed in section 4.3, each replicate can produce only one independent sample, x_i. This independent sample is often a statistic computed from the observations of a performance measure, usually the average, minimum, or maximum. For example, one average of the number in the buffer at a workstation A is computed from all of observations made during one replicate. This average is one independent sample of the average number in the buffer.

Statistical summaries are estimated from the x_i's. These summaries typically include the average, standard deviation, minimum, and maximum. Confidence intervals are also of interest.

In summary, each simulation experiment consists of n replicates. Within each replicate and for each performance measure, one or more observations are made. From the observations, one or more statistics are typically computed. Each such statistic is the independent observation, x_i, produced by the replicate.

For example, a simulation experiment concerning the two work stations in a series model could consist of 20 replicates. The number of entities in the workstation A buffer could be observed. Each time the number in the buffer changes an observation is made. The average number in the buffer as well as the maximum number in the buffer is computed. There are 20 independent observations of the average number in the buffer as well as 20 independent observations of the maximum number in the buffer.

The number of replicates initially made is generally determined by experience and the total amount of real ("clock") time needed to compute the simulation. Most of the time, this number is in the range 10-30. More replicates may be needed if the width of a confidence interval computed from the performance measure observations is considered to be too wide. Confidence interval estimation is discussed later in this chapter.

The number of replications of the simulation experiment must be specified.

4.5.3 Ending the Simulation

This section discusses the time or condition that determines when to end a replicate.

An ending time for a replicate arises naturally from an examination of most systems. A manufacturer wants to know if its logistic equipment will suffice for the next budget period of one year. So the end of the budget year becomes the simulation ending time. A fast food restaurant does most of its business from 11:30 A.M to 12:30 P.M. Thus the simulation ending time is one hour. The experiment for a production facility model could cover the next planning period of three months. After that time, new levels of demand may occur and perhaps new production strategies implemented. The simulation experiment for a production facility could end when 100 parts are produced.

4.5.4 Design Summary

The specification of design elements for a terminating simulation experiment can be accomplished by completing the template shown in Table 4-1.

Table 4-1: Terminating Simulation Experiment Design

Element of the Experiment	Values for a Particular Experiment
Model Parameters and Their Values	
Performance Measures	
Random Number Streams	
Initial Conditions	
Number of Replicates	
Simulation End Time / Event	

Consider a terminating simulation experiment for the two workstations in a series model. The time between arrivals and the operation time at workstation A are modeled using probability distributions. Performance measures include the number in the buffer at each workstation, the state of the each workstation (BUSY or IDLE), and entity lead time. The model parameter is the machine used at workstation A, either the current machine with operation time uniformly distributed between 5 and 13 seconds or a new machine with operation time uniformly distributed between 7 and 11 seconds. The initial conditions are two items in each buffer and both workstations busy. Twenty replicates will be made for the planning horizon of one work week. The experimental design is shown in Table 4-2.

Table 4-2: Simulation Experiment Design for the Two Workstations in Series Model

Element of the Experiment	Values for a Particular Experiment
Model Parameters and Their Values	Workstation A Machine: Current vs. New
Performance Measures	Number in the buffer at each workstation State of each workstation Lead Time
Random Number Streams	Time between arrivals Operation Time
Initial Conditions	Two entities in each buffer One entity in service at each workstation (State of each workstation resource is BUSY)
Number of Replicates	20
Simulation End Time / Event	1 week (40 hours)

4.6 Examining the Results for a Single Scenario

This section presents a strategy for examining the simulation results of a single system scenario as defined by one set of model parameter values. Results are examined to gain an understanding of system behavior. Statistical evidence in the form of confidence intervals is used to confirm that what is observed is not just due to the random nature of the simulation model and experiment and thus provides a valid basis for understanding system behavior.

Simulation results are displayed and examined using graphs and histograms as well as summary statistics such as the mean, standard deviation, minimum, and maximum. Patterns of system behavior are identified if possible. Animation is used to display the time dynamics of the simulation. This is in accordance with principle 8: Looking at all the simulated values of performance measures helps.

How the examination of simulation results is successfully accomplished is an art as stated in principle 1. Thus, this topic will be further discussed and illustrated in the context of each application study.

The discussion in this session is presented in the context of the two work stations in a series model.

4.6.1 Graphs, Histograms, and Summary Statistics

Observed values for each performance measure can be examined via plots, histograms, and summary statistics. To illustrate, each of these will be shown for the number of entities in the buffer of workstation A in the two workstations in a series model.

A plot of the observed values of the number in this buffer from replicate one of the simulation experiment defined in Table 4-2 is shown in Figure 4-3. The x-axis is simulated time and the y-axis is the number in the buffer of workstation A. Note from the plot that most of the time the number in the buffer varies between 0 and 10. However, there are several occasions that the number in the buffer exceeds 20. This shows high variability at workstation A.

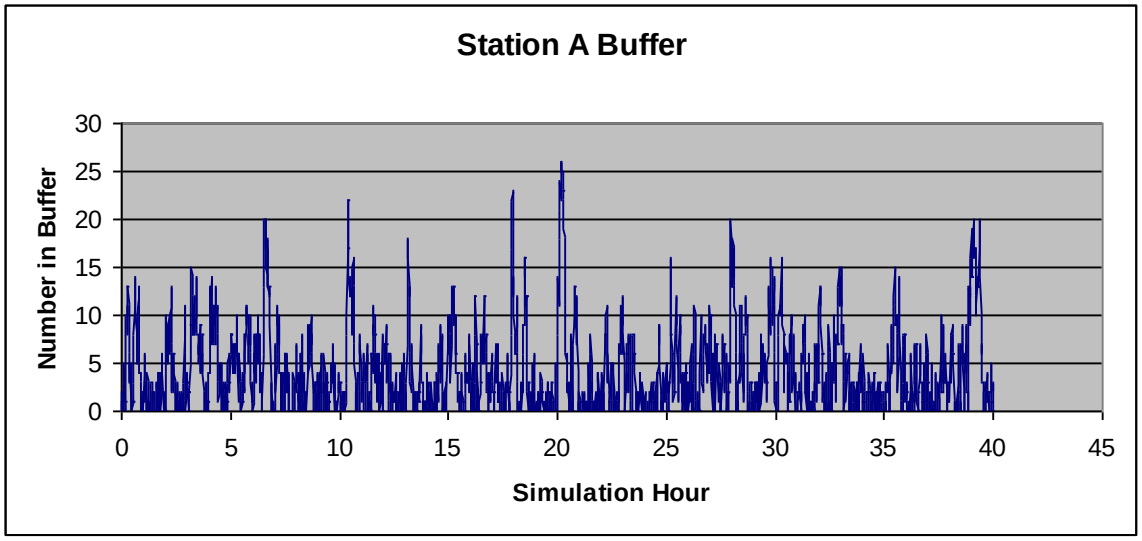


Figure 4-3: Plot of the Number of Entities in the Workstation A Buffer

A histogram of the same observations is shown in Figure 4-4. The percent of time that a certain number of entities is in the buffer is shown on the y-axis. The number of entities is shown on the

x-axis. Note that about 91% of the time there are 10 or less entities in the buffer of workstation A. However about 9% of the time there are more than 10 entities in the buffer.

It would be wise to examine these same graphs from other replicates to see if the same pattern of behavior is observed. If the software capability is available, a histogram combining the observations from all of the replicates would be of value.

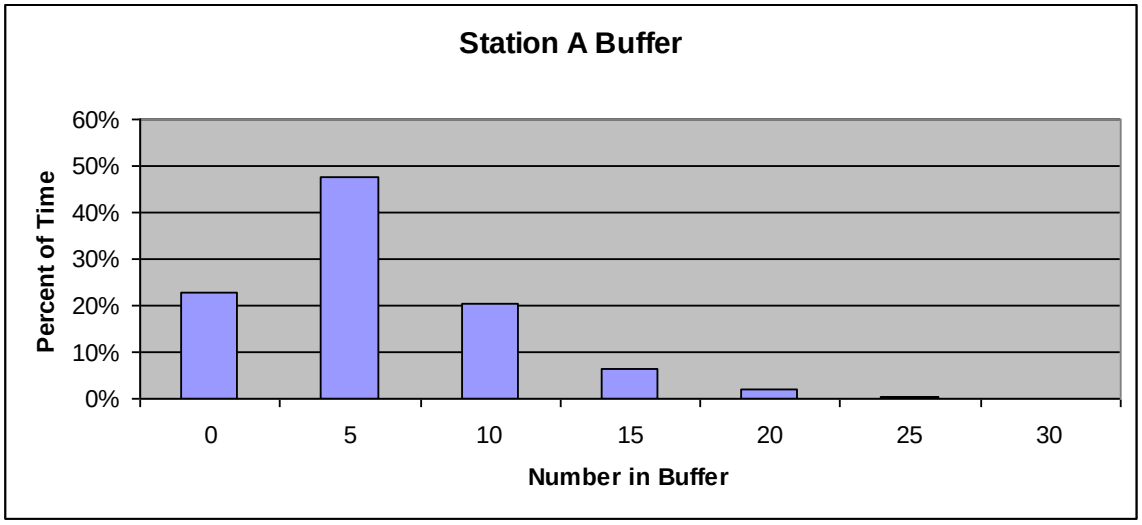


Figure 4-4: Histogram of the Number of Entities in the Workstation A Buffer

Summary statistics can be computed from the observations collected in each replicate. However, these observations are likely not independent, so their standard deviation is not very useful. The average, minimum, and maximum of the observations of the number in the buffer of workstation A from replicate 1 are given in Table 4-3. The average number of entities is relatively low but the maximum again shows the high variability in the number in the buffer.

Table 4-3: Summary Statistics for the Number of Entities in the Buffer of Workstation A – Replicate 1

Statistic	Value
Average	4.1
Minimum	0
Maximum	26

As was previously discussed, one independent observation each of the average, minimum, and maximum is generated by each replicate. Suppose the average and maximum number in the buffer of workstation A are of interest. The average corresponds to the average work-in-process (WIP) at the workstation and the maximum to the buffer capacity needed at the workstation. Table 4-4 summarizes the results for 20 replicates. The average ranges from 3.1 to 6.6 with an overall average of 4.4. This shows that the average number in the buffer has little variability. The maximum shows significant variability ranging from 21 to 43 with an average of 31.

Table 4-4: Summary Statistics for the Number of Entities in the Buffer of Workstation A – Replicate 1 through 20

Replicate	Average Number in the Workstation A Buffer	Maximum Number in the Workstation A Buffer
1	4.1	28
2	4.6	27
3	4.1	30
4	3.2	24
5	3.8	24
6	4.3	29
7	4.0	25
8	4.4	34
9	4.3	40
10	4.1	28
11	4.1	26
12	4.5	38
13	4.5	31
14	4.3	30
15	4.8	37
16	4.2	28
17	5.2	40
18	4.3	38
19	4.3	26
20	4.4	36
Average	4.3	31.0
Std. Dev.	0.39	5.4
Minimum	3.2	24
Maximum	5.2	40

4.6.2 Confidence Intervals

One purpose of a simulation experiment is to estimate the value of a parameter or characteristic of the system of interest such as the average or maximum number in the buffer of workstation A. The actual value of such a parameter or characteristic is most likely unknown. Both a point estimator and an interval estimator are needed. The point estimator should be the center point of the interval.

The average of the set of independent and identically distributed observations, one from each replicate, serves as a point estimator. For example, the values in the "average" row of Table 4-4 are point estimators, the first of the average WIP in the buffer of workstation A and the second of the needed buffer capacity.

The confidence interval estimation procedures recommend by Law (2007) will be used to provide an interval estimator. The **t-confidence interval** given by equation 4-1 is recommended.

$$P\left(\overline{X} - t_{1-\alpha/2,n-1} * \frac{s}{\sqrt{n}} \leq \mu \leq \overline{X} + t_{1-\alpha/2,n-1} * \frac{s}{\sqrt{n}} \right) \approx 1-\alpha \qquad (4\text{-}1)$$

where $t_{1-\alpha/2,n-1}$ is the 1-$\alpha/2$ percentage point of the Student's t distribution with n-1 degrees of freedom, n is the number of replicates, $\overline{X}$ is the average (the values on the "average" row of Table 4-4 for example), and s is the standard deviation (the values on the "std. dev." row of Table

4-4 for example). The $\approx$ sign means approximately. The symbol μ represents the actual but unknown value of the system parameter or characteristic of interest.

The result of the computations using equation 4-1 is the interval shown in equation 4-2:

(lower bound $\leq \mu \leq$ upper bound) with 1-α confidence (4-2)

where

lower bound = $\overline{X} - t_{1-\alpha/2,n-1} * \frac{s}{\sqrt{n}}$ (4-3)

upper bound = $\overline{X} + t_{1-\alpha/2,n-1} * \frac{s}{\sqrt{n}}$ (4-4)

Equations 4-1 and 4-2 show the need to distinguish between probability and confidence. Understanding this difference may require some reflection since in everyday, non-technical language the two ideas are often used interchangeably and both are expressed as a percentage.

A probability statement concerns a random variable. Equation 4-1 contains the random variables $\overline{X}$ and s and thus is a valid probability statement. The interpretation of equation 4-1 relies on the long run frequency interpretation of probability and is as follows: If a very large number of confidence intervals are constructed using equation 4-1, the percentage of them that include the actual but unknown value of μ is approximately 1-α. This percentage is called the **coverage**.

The interval expressed in equation 4-2 contains two numeric values: lower bound and upper bound plus the constant μ whose value is unknown. Since there are no random variables in equation 4-2, it cannot be a probability statement. Instead, equation 4-2 is interpreted as a statement of the degree of confidence (1-α) that the interval contains the value of the system parameter or characteristic of interest. Typical values for (1-α) are 90%, 95%, and 99%. A higher level of confidence implies more evidence that the interval contains the value of μ.

Some thoughts on how to interpret the level of confidence with respect to the kind of evidence provided is worthwhile. Keller (2001) suggests the following, which will be used in this text.

Table 4-5. Interpretation of Confidence Values

Confidence (1-α) Range	Interpretation
(1-α) $\geq$ 99%	Overwhelming evidence
95% $\geq$ (1-α) > 99%	Strong evidence
90% $\geq$ (1-α) > 95%	Weak evidence
90% > (1-α)	No evidence

Note that the higher the level of confidence the greater the value of $t_{1-\alpha/2,n-1}$ and thus the wider the confidence interval. A narrow confidence interval is preferred so that the value of μ is more precisely bounded. However, it is clear that a high level of confidence must be balanced with the desire for a narrow confidence interval.

Why equation 4-1 is approximate and not exact is worthy of discussion. For equation 4-1 to be exact, the observations on which the confidence interval computations are based must come from a normal distribution as well as being independent and identically distributed. As was previously discussed, the latter two conditions are met by the definition of a replicate while the first condition cannot be guaranteed since the performance measures in a simulation are arbitrarily defined.

Thus, equation 4-1 is approximate. Approximate means that the coverage produced using equation 4-1 will likely be less than 1-α.

Given that equation 4-2 provides only an approximate (not exact) level of confidence (not a probability), it is natural to ask why it should be used. Law (2007) concludes that experience has shown that many real-world simulations produce observations of the type for which equation 4-1 works well, that is the coverage produced using equation 4-1 is close enough to 1-α to be useful in conducting simulation studies. In the same way, Vardeman and Jobe (2001) state that confidence intervals in general have great practical use, even though no probability statement can be made as to whether a particular interval contains the actual value of the system characteristic or parameter of interest. Since confidence intervals seem to work well in general and in simulation studies, they will be used throughout this text.

As an example, Table 4-6 contains the 99% confidence intervals computed from equation 4-2 for the average and maximum number of entities in the buffer of workstation A based on the results shown in Table 4-4.

Table 4-6: 99% Confidence Intervals for the Number of Entities in the Buffer of Workstation A Based on 20 Replicates

	Average Number in the Workstation A Buffer	Maximum Number in the Workstation A Buffer
Average	4.3	31.0
Std. Dev.	0.39	5.4
99% CI – Lower Bound	4.0	27.5
99% CI – Upper Bound	4.5	34.4

The confidence interval for the average is small. It would be safe to conclude that the average number in the buffer of workstation A was 4 (in whole numbers). The confidence interval for the maximum number in the buffer ranges from 27 to 34 (in whole numbers). If this range is deemed too wide to establish a buffer size additional replicates, say another 20, could be made.

4.6.3 Animating Model Dynamics

As discussed in chapter 1, simulation models and experiments capture the temporal dynamics of systems. However, reports of models and experimental behavior are often confined to static mediums such as reports and presentations like those shown in the preceding sections. The simulation process includes system experts and managers who may not be knowledgeable about modeling methods and may be skeptical that a computer model can represent the dynamics of a complex system. In addition, complex systems may include complex decision rules. All behavioral consequences resulting from these rules may be difficult to predict.

Addressing these concerns involves answering the question: What system behavior was captured in the model? One very effective way of meeting this requirement is seeing the behavior graphically. This is accomplished using animation.

Typical ways of showing simulated behavior using animation follow:

1. State of a resource with one unit: The resource is represented as a graphical object that physically resembles what the resource models. For example, if the resource models a lathe, then the object looks like a lathe. Each state of the resource corresponds to a different color. For example, yellow corresponds to IDLE, green to BUSY, and red to BROKEN. Color changes during the animation indicate changes in the state of the resource in the simulation.

2. Entities: An entity is represented in the frame as a graphical object that physically resembles what the entity models. Different colors may be used to differentiate entities with different characteristics. For example if there are two types of parts, graphical objects representing part type 1 may be blue and those representing part type 2 may be white.

3. Number of entities in a buffer: A graphical object, which may be visually transparent, represents the buffer. An entity graphical object is placed in the same location as the buffer graphical object whenever an entity joins the buffer in the simulation. The buffer graphical object accommodates multiple entity graphical objects.

4. Material transportation: Any movement, such as between workstations, of entities in the simulation can be shown on the animation. The location of an entity graphical object can be changed at a rate proportional to the speed or time duration of the movement. Movement of material handling equipment can be shown in a similar fashion. As for other resources, a piece of material handling equipment is represented by a graphical object that resembles that piece of equipment. For example, a forklift is represented by a graphical object that looks like a forklift.

An animation of the two-stations in a series system should be viewed at this time.

4.7 Comparing Scenarios

This section presents a strategy for determining if simulation results provide evidence that one scenario is better than another. Often one scenario represents current system operations for an existing system or a baseline design for a proposed system. Improvements to the current operations or to a baseline design are proposed. Simulation results are used see if these improvements are significant or not. In addition, it may be necessary to compare one proposed improvement to another. This is an important part of step 3 Identify Root Causes and Assess Initial Scenarios as well as step 4 Review and Extend Previous Work of the simulation project process.

Often, pair-wise comparisons are made. This will be the scope of our discussion. Law (2007) provides a summary of methods for ranking and selecting the best from among all of the scenarios that are considered.

The job of comparing scenario A to scenario B is an effort to find evidence that scenario A is better than scenario B. This evidence is found first by examining observations of performance measures to see if any operationally significant differences or unexpected differences can be seen. If such differences are seen, an appropriate statistical analysis is done to confirm them. Confirm means to determine that the differences are not due to random variation in the simulation experiment.

Many times a scenario is better with respect to one performance measure and the same or worse with respect to others. Evaluating such tradeoffs between scenarios is a part of the art of simulation.

Each of the ways of comparing scenarios will be discussed in the context of the simulation experiment concerning the two stations in a series model. This experiment is presented in Table 4-2. The primary performance measure of interest will be entity lead time.

4.7.1 Comparison by Examination

Some ways of comparing two scenarios by examination of performance measure observations follow.

1. For each replicate (or at least several replicates), graph all observations of a performance measure.

For example, the graph of the number in the buffer of workstation A for the scenario for the current machine in use at workstation A is shown in Figure 4-3. This could be compared to the graph of the same quantity for the scenario where the new machine is used at workstation A. If the latter graph consistently showed fewer entities in the buffer, then there would be evidence that that using the new machine at workstation A is an improvement: less WIP.

Graphing lead time observations is not usually done since lead time is not a state variable and does not have a value at every moment in simulation time.

2. For each replicate or over all replicates, compare the histograms of the observations of a performance measure.

For example, histograms of lead time can be compared. If the histogram for the new machine at workstation A scenario clearly shows a greater percentage of entities requiring less time on the line versus the current machine scenario, then there would be evidence that using the new machine at workstation A lowers cycle time.

3. Compare the averages of the sample values, x_i, gathered from the replicates. Note whether the difference in the averages is operationally significant.

For example, the average lead time for the current machine scenario is 62.7 seconds and for the new machine scenario is 58.5 seconds. These values are for all replicates of the experiment. Thus, the new machine reduces cycle time by about 6%, which is operationally significant.

4. Compare the range [minimum, maximum] of the sample values, x_i. Note whether the ranges overlap.

For example, the range of cycle time averages over the replicates of the experiment for the current machine scenario is (52.5, 71.7) and for the new machine scenario is (48.8, 68.9). The ranges overlap and thus provide no evidence that the new machine reduces cycle time verses the existing machine at workstation A.

4.7.2 Comparison by Statistical Analysis

This section discusses the use of confidence intervals to confirm that perceived differences in the simulation results for two scenarios are not just due to random variation in the experiment.

Note that the experiment design assures that scenarios share random number streams in common. Thus, the scenarios are not statistically independent. Furthermore, the same number of replicates is made for each scenario. Thus, an approach that compares the simulation results

on a replicate by replicate basis is required and helpful. This approach is called the paired-t method.[1]

Table 4-7 provides the organization to support the paired-t method. Each row corresponds to a replicate. The difference between the performance measure values for each replicate is shown in the fourth column. These differences are independent observations. A 1-α confidence interval for the population mean of the difference in the fourth column is computed. If this confidence interval does not contain zero, it will be concluded that there is a statistically significant difference between the scenarios with confidence 1-α. This confidence interval is constructed and interpreted using the same reasoning as was given in section 4.6.2.

To illustrate, Table 4-8 compares, based on entity lead time, the use of the new machine at workstation A versus the current machine using the paired-t method. A 99% confidence interval for the mean difference is constructed: (3.7, 4.7) with 99% confidence. Thus, with 99% confidence the new machine at workstation A reduces mean cycle time in the range (3.7, 4.7) seconds.

It is also helpful to examine the data in Table 4-8 on a replicate-by-replicate basis. Notice that in all of the replicates, cycle time was less using the new machine at workstation A. It should be noted however that it is still quite possible that in any particular 40 hour period, the two stations in a series line would perform better with respect to cycle time using the current machine at workstation A instead of the new machine. The simulation results show that on the average over many 40 hour periods the line will perform better with respect to cycle time using the new machine at workstation A.

Table 4-7: Format of the Paired-t Method

Replicate	Scenario A	Scenario B	Difference (Scenario A – Scenario B)
1			
2			
3			
4			
. . .			
n			
Average			
Std. Dev.			
1-α C. I. Lower Bound			
1-α C.I. Upper Bound			

[1] Law (2007) provides a more in depth discussion of the comparison of alternatives using confidence intervals, including the generation of confidence intervals when common random numbers are not used.

Table 4-8: Comparison of Scenarios Using the Paired-t Method (1-α = 99%)

Replicate	Current Machine	New Machine	Difference (Current – New)
1	61.1	57.3	3.8
2	66.0	62.2	3.9
3	60.6	57.6	3.0
4	52.5	48.8	3.7
5	58.3	55.0	3.3
6	63.4	59.3	4.0
7	59.7	55.0	4.8
8	63.9	59.2	4.7
9	62.7	58.5	4.2
10	61.1	56.7	4.4
11	60.7	56.6	4.1
12	65.2	59.8	5.4
13	64.7	58.3	6.4
14	63.6	59.5	4.1
15	67.3	63.5	3.8
16	61.7	57.2	4.5
17	71.7	68.9	2.8
18	63.3	59.0	4.3
19	62.3	58.1	4.2
20	64.6	59.9	4.7
Average	62.7	58.5	4.2
Std. Dev.	3.82	3.8	0.8
99% C. I. Lower Bound	60.9	56.7	3.7
99% C.I. Upper Bound	64.5	60.3	4.7

4.7.2.1 A Word of Caution about Comparing Scenarios

In comparing scenarios, many confidence intervals may be constructed. For each pair of scenarios, several performance measures may be compared. Many scenarios may be tested as well.

The question arises as to the resulting α level for all confidence intervals together, $\alpha_{overall}$. This $\alpha_{overall}$ level is the probability that all confidence intervals simultaneously cover the actual difference in value between the scenarios of the system parameter or characteristic each estimates.

A lower bound on $\alpha_{overall}$ is computed using the Bonferroni inequality where a total of k confidence intervals are conducted:

P(all confidence intervals cover the actual value) >= $1 - \sum_{j}^{k} \alpha_j$ (4-5)

and thus:

$$\alpha_{overall} \leq \sum_{j=1}^{k} \alpha_j \quad (4\text{-}6)$$

Suppose we compare two scenarios using two performance measures with α = 0.05. A confidence interval of the difference between the scenarios is computed for each performance measure. The lower bound on the probability that both confidence intervals cover the actual difference in the performance measures is given by equation 4-5: $\alpha_{overall}$ <= 0.05 + 0.05 = 0.10.

Consider comparing two scenarios with respective to 10 performance measures. Each confidence interval is computed using $\alpha = 0.05$. Then the probability all confidence intervals cover the actual difference in the performance measures might be as low as 0.05*10 = 0.50. That is the α error associated with all our work would be 0.5. Thus, when making many comparisons, a small value of α_j for each confidence interval is necessary. For example with all $\alpha_j = 0.01$, the overall α error associated with ten comparisons is 0.1, which is acceptably low.

Unfortunately if a large number of performance measures are used or many scenarios are compared, $\alpha_{overall}$ will always be large. Thus, it is likely that for at least one confidence interval that the true difference between the performance measure values will not be covered. So a difference between two scenarios will not be detected.

4.8 Summary

This chapter discusses the design and analysis of simulation experiments. Elements are defined and organized into a design. A method to construct statistically independent observations to avoid correlation difficulties is described.

The need to gather evidence that a model is valid and verified is presented. Possible strategies in this regard are given. Ways to compare scenarios, both through statistical analysis and the examination of data, are discussed.

Problems (Similar problems are associated with each of the case studies for further practice).

1. Suppose 4 scenarios were compared in a pair-wise fashion with respect to one performance measure. How many comparisons are made? If $\alpha = 0.01$ is used for all comparisons, what is the upper bound on the α for all the comparisons made? What if $\alpha = 0.05$ is used? Which of the two values for α should be used?

2. Consider the following table of simulation results.

Replicate	**Workstation % Busy Time – Scenario One**	**Workstation % Busy Time – Scenario Two**
1	87	78
2	80	72
3	79	71
4	80	72
5	78	71

   a. Construct 95% confidence intervals for the workstation % busy time for each scenario.

   b. Construct a paired-t confidence interval, $\alpha = 0.05$, to compare the percent busy time of a workstation for two scenarios.

3. Consider the following table of simulation results.

Replicate	Maximum Time – Scenario One	Maximum Time – Scenario Two
1	241.8	122.0
2	61.1	62.6
3	122.1	94.7
4	111.6	73.1
5	154.4	105.2

   a. Construct 95% confidence intervals for the maximum time for each scenario.

   b. Construct a paired-t confidence interval, $\alpha = 0.05$, to compare the maximum time at the workstation for the two scenarios.

   c. Are the confidence intervals constructed in a. and b. approximate or exact? Defend your answer.

4. Develop the design of a terminating simulation experiment for problem 2-10.

5. Defend the use of approximate confidence intervals.

6. Consider the simulation of a single workstation consisting of a single machine with an operation time uniformly distributed between 5 and 10 minutes. The time between part arrivals is exponentially distributed with a mean of 9 minutes.

   a. What verification evidence could be sought?

   b. What validation evidence could be sought?

7. Conduct a complete analysis of a simulation experiment regarding a single workstation with one machine based on the data that follow. The mean time between arrivals is 10 minutes and the operation time is 8 minutes. The simulation was run for 168 hours. Management wishes to achieve a production quota of 1000 items per 168 hours.

   a. Provide evidence for the verification and validation of the simulation based only on the data in the following table and the problem statement.

Replicate	Workstation % Busy Time	Number of Entities Arriving	Number of Entities Departing	Number of Entities in Processing at the End of the Simulation	Number of Entities in the Buffer at the End of the Simulation
1	87	1044	1044	0	0
2	80	961	960	1	0
3	79	944	943	1	0
4	80	965	959	1	5
5	78	942	942	0	0

b. Compare the two scenarios using first the average number in the buffer and then the maximum as described below. Use only the data that follows and items i-iv.

i. Compute appropriate statistical summaries (average, standard deviation, minimum, maximum, range, and confidence intervals) and state any evidence found from this information.

ii. Compute and display appropriate histograms and state any evidence seen in them.

iii. In how many replicates is the new case better than the current operations? What evidence does this information provide?

iv. Perform the appropriate statistical analysis to compare the scenarios.

	Number in Buffer			
	Current Operations		New Case	
Replicate	Average	Maximum	Average	Maximum
1	12.8	28	4.3	15
2	1.2	8	1.1	7
3	4.3	16	2.6	16
4	2.9	10	1.9	8
5	3.6	17	2.1	12
6	3.7	10	2.0	8
7	2.1	12	1.2	7
8	3.5	17	1.6	11
9	2.7	13	1.4	9
10	2.0	10	1.2	9
11	1.4	8	1.3	10
12	2.0	12	1.4	10
13	1.4	7	1.4	9
14	2.7	17	2.0	12
15	1.7	16	1.2	9
16	1.5	7	0.9	8
17	5.2	26	4.2	17
18	3.2	15	2.0	9
19	3.1	14	2.0	9
20	2.2	11	1.2	8

Chapter 5
The Simulation Engine

5.1 Introduction

This chapter discusses the computations necessary to simulate a model on a computer. These computational tasks are performed by software that can be referred to as a **simulation engine.** The engine produces performance measure values as output. It does this transparently to the simulation user whose primary concern is performing the steps of the simulation process including model building and experiment design as well as the statistical analysis of performance measure values and drawing conclusions about system behavior. Nevertheless, a basic understanding of how a simulation engine does its computation tasks is fundamental.

All models are mapped, transparently to the modeler, into a set of events within the simulation engine. The mapping may be complex and not straightforward. An event is a point in simulation time when the value of one or more the state variables changes. In addition an event is used to specific when in simulated time, or under what conditions, other events, including itself, next occur.

The basic operations that a simulation engine must perform are presented in the context of the two workstation example model that was presented in previous chapters. Fundamentally, the engine must conduct the simulation step by step from start to finish. This requires

1. Sequencing the events.
2. Processing each event.
3. Organizing entities waiting for resources.
4. Generating individual samples from probability distributions to obtain values for entity attributes and times between entity arrivals as well as operation and transportation times.

A discussion of the events in the two workstation example will precede a discussion of each of the activities of the simulation engine.

5.2 Events and Event Graphs

Event graphs (Schruben, 1983; 1995) are a diagramming technique for showing the events comprising a model. An event graph consists of nodes and arcs. Nodes correspond to events. Arcs tell the relationships between events: the other events, including itself, that an event can cause to occur and the logical conditions that may restrict such occurrences. The logical conditions make use of the state variables. An arc also tells the time from now when an event will take place.

The event graph for the two station serial system is shown in Figure 5-1. There are four state variables: the number in the buffer of each station and the state (busy, idle) of each station. Three events are associated with each station: *Entity arrives, Start service,* and *End service*.

The entity arrives event associated with station A causes itself to occur again, that is the next entity to arrive, after a time interval specified by the time between arrivals. The number in the buffer of workstation A is incremented by 1.

The entity arrives event causes the start service event to begin processing the arriving entity immediately if the machine is IDLE. The start service event decreases the number in the buffer of workstation A by 1 and makes the workstation BUSY.

The end service event follows the start service event and occurs after a time interval that is the item processing time. The end service event will initiate processing of the first entity waiting in the buffer if there is one by scheduling the start service event at the current time. The end service event makes the workstation IDLE.

The time between arrivals to station A and the item processing time could be random variables.

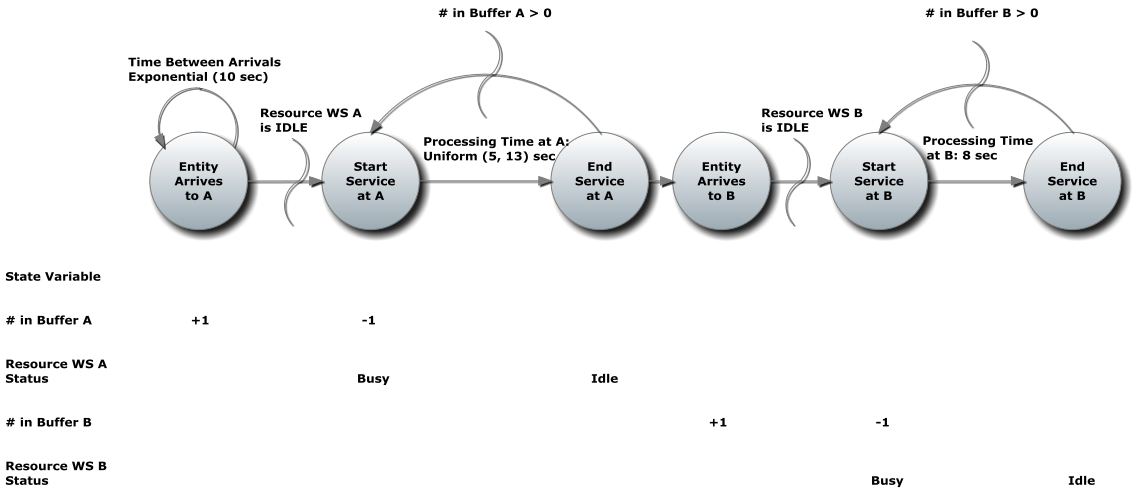


Figure 5-1: Event Graph for Two Workstations in a Series Model

5.3 Time Advance and Event Lists

This section discusses how the simulation proceeds through time by scheduling event occurrences and processing each of them in turn. In general, a model is simulated as a time ordered sequence of the occurrences of the events. Event occurrences are processed one at a time. Each event occurrence changes the value of one or more state variables and may schedule other events. This simulation approach is illustrated by one possible simulation of the two workstations in sequence model.

The **event list** is the time ordered list of all event occurrences scheduled at the current time and in the future. The simulation proceeds by removing the first event occurrence on the list and processing it. This processing may result in one or more event occurrences being added to the list to be processed at the current time or in the future. Note that only one event occurrence at a time is removed from the list. All others remain on the list. After the processing of the event occurrence, the list will consist of the event occurrences already on the list when the first event occurrence was removed plus those added by processing this event occurrence.

For the two workstation model, the simulation engine must deal with six events that change the values of the state variables. Each of these events must be scheduled in time and processed. These events are the arrival of an entity (part) to each station as well as the start and end of processing.

At any point in time, the event list could contain the following event occurrences at future points in time:

- Entity arrives to workstation A event.
- Entity ends service at workstation A event.
- Entity ends service at workstation B event.

Other events can occur only at the same point in time as another event.

- The start of service at workstation A event that can occur either when a entity arrives to workstation A (arrives to workstation A event) or when a entity completes processing at workstation A (ends service at workstation A event).

- The entity arrives to workstation B event that occurs every time an entity ends service at workstation A event occurs. (Recall there is no time delay for moving between workstations.)
- The start of service at workstation B event that can occur either when a entity arrives to workstation B (arrives to workstation B event) or when an entity completes processing at workstation B (ends service at workstation B event).

To illustrate, consider one possibility for the event list at the start of the simulation.

Current Simulation Time: 0
Next Simulation Time = Time of first event occurrence in list = 0.0

Event	Time of Occurrence	Entity ID
Entity Arrives to A	0.0	1

The simulation will begin with the arrival of the first entity at time 0.

Thus, the first task of the simulation engine is to process the Entity Arrives to A event at time 0. This task involves removing the Entity Arrives to A event from the list and performing the actions associated with the event: scheduling the next Entity Arrives to A event and scheduling the Entity Start Service event, if workstation A is idle (which it is initially). After the entity arrives to A event is processed the event list is as follows, assuming the next arrival to workstation A is at time 5.0:

Current Simulation Time: 0
Next Simulation Time = Time of first event occurrence in list = 0.0

Event	Time of Occurrence	Entity ID
Start Service at A	0.0	1
Entity Arrives to A	5.0	2

Next, the simulation engine removes the Start Service at A event from the list. The Entity Arrives to A event remains on the list. Processing the event removed from the list results in scheduling the End Service at A event as shown in the following.

Current Simulation Time: 0
Next Simulation Time = Time of first event occurrence in list = 5.0

Event	Time of Occurrence	Entity ID
Entity Arrives to A	5.0	2
End Service at A	8.0	1

The entity with ID number 2 will arrive at time 5.0 and the End of Service at workstation A for the entity with ID number 1 will occur at time 8.0.

The simulation engine advances time to the next event occurrence at time 5.0 and processes the Entity Arrives to A event for the entity with ID 2. This means that the Entity Arrives to A event will be removed from the list and End Service at A event will remain on the list.

At time 5.0, the workstation A resource is in the busy state. Thus, the entity with ID 2 enters the queue for the workstation A resource. No entry for this entity is placed on the event list. In addition, processing this event causes the Entity Arrives to A event to be scheduled at time 32.5 for the entity with ID 5. This means that the next occurrence of the Entity Arrives to A event is placed on the event calendar at time 32.5.

Thus, after processing the Entity Arrives at A event occurrence at time 5.0, the event list consists of the End Service at A event which was previously on the list plus the next Entity Arrives to A event that was newly placed on the list as shown in the following.

Current Simulation Time: 5.0
Next Simulation Time = Time of first event occurrence in list = 8.0

Event	Time of Occurrence	Entity ID
End Service at A	8.0	1
Entity Arrives to A	32.5	3

Next, the simulation engine advances time to the 8.0 to process the end of service at A event for entity 1. The entity with ID number 1 will arrive at workstation B at time 8.0 since there is no movement delay. The entity with ID number 2 will leave the queue of the workstation A resource and start processing using the workstation A resource that has just become idle. Thus, the workstation A resource becomes busy.

Thus after processing the End Service at A event, the Entity Arrives to A event remains on the list and the Entity Arrives to B event as well as the Start Service at A event are added.

Current Simulation Time: 8.0
Next Simulation Time = Time of first event occurrence in list = 8.0

Event	Time of Occurrence	Entity ID
Entity Arrives to B	8.0	1
Start Service at A	8.0	2
Entity Arrives to A	32.5	3

Simulation engines typically use the strategy that all possible processing of one entity at the current simulation time will be done before any processing of any other entity. Another way of saying this is that the entity will proceed as far as possible until obstructed by a time delay or by waiting for a currently unavailable resource. This implies that new events at the current simulation time for this entity are placed first on the event list. Thus in the above list, the entity arrives to B event for the entity with ID 1 at time 8.0 precedes the start service at A event for the entity with ID number 2.

The remainder of the simulation is processed in a similar fashion.

5.4 Simulating the Two Workstation Model

This section discusses and illustrates the record of the time ordered sequence of events that are processed by a simulation engine for a particular model. This record is called a **trace** and includes the changes in state variable values that occur as well as other relevant information such as entity attributes. All simulation engines provide a trace that the modeler can examine to determine the step-by-step behavior of a simulation for verification and validation.

Consider one possible simulation of the two workstations in sequence model. Let's follow the sequence of events processed in time order when only one entity moves through the two workstations, assuming that no other entities arrive in the meantime.

The trace for the simulation with one entity is shown Table 5-1. Only the new values of state variables whose values are changed by an event are shown. At the start of the simulation there are no entities in the model, the buffers are empty and the workstation resources are in the IDLE state. The entity with ID number 1 arrives at time 0 and enters the buffer of workstation A. Since the workstation A resource is IDLE, the start service at A event occurs at time 0. This event removes the entity from the buffer of workstation A and makes the workstation A resource BUSY.

Table 5-1: Simulation Trace for One Entity

Current Simulation Time	Event	Entity ID	Number in Buffer – Station A	State of Workstation A Resource	Number in Buffer – Station B	State of Workstation B Resource
0.0	Initial Conditions		0	IDLE	0	IDLE
0.0	Entity Arrives to A	1	1			
0.0	Start Service at A	1	0	BUSY		
8.0	End Service at A	1		IDLE		
8.0	Entity Arrives to B	1			1	
8.0	Start Service at B	1			0	BUSY
16.5	End Service at B	1				IDLE

The simulation engine must determine the duration of processing at workstation A for this particular entity. This is done by computing a random sample from the processing time distribution: uniform (5,13). Suppose the value turns out to be 8.0. Thus, the end of service at A event is placed at time 8.0. At this time, the workstation A resource becomes IDLE.

The entity arrives at B event occurs at time 8.0 as well since there is no time delay for movement between the workstations. The entity enters the buffer of workstation B. Since the workstation B resource is IDLE, the start service at B event occurs at time 8.0. The duration of processing at workstation B is a constant 8.5. Thus, the end of service at B event is placed at time 16.5.

Now, suppose a second entity arrives in the simulation at time 5.0. This time is determined by computing a value from the time between arrivals distribution: exponential (10) when the entity arrives at A event is processed for entity 1 at time 0. Suppose further that the simulation engine computes the service time at workstation A to be 7.0. Table 5-2 shows the trace of the simulation for this situation.

Note the events involving entity 2. Since the workstation A resource is BUSY when entity 2 arrives at time 5.0, it remains in the buffer of station A. At time 8.0, all events concerning entity 1 are processed first. After these events are processed, the start service at A event is processed for entity 2. Since the processing time is computed to be 7.0, the end of service at A event is placed at time 15.0.

At time 15.0, the end of service at A event occurs for entity 2 as well as the entity arrives to B event. Since workstation B resource is busy, entity 2 waits in the buffer of station B.

At time 16.5, the end of service at B event occurs for entity 1. Since the workstation B resource becomes IDLE, start of service at B event occurs for entity 2. At time 25.0, entity 2 completes processing at workstation B.

Table 5-2: Simulation Trace for Two Entities

Current Simulation Time	Event	Entity ID	Number in Buffer – Station A	State of Workstation A Resource	Number in Buffer – Station B	State of Workstation B Resource
0.0	Initial Conditions	--	0	IDLE	0	IDLE
0.0	Entity Arrives to A	1	1			
0.0	Start Service at A	1	0	BUSY		
5.0	Entity Arrives to A	2	1			
8.0	End Service at A	1		IDLE		
8.0	Entity Arrives to B	1			1	
8.0	Start Service at B	1			0	BUSY
8.0	Start Service at A	2	0	BUSY		
15.0	End Service at A	2		IDLE		
15.0	Entity Arrives to B	2			1	
16.5	End Service at B	1				IDLE
16.5	Start Service at B	2			0	BUSY
25.0	End Service at B	2				IDLE

5.5 Organizing Entities Waiting for a Resource

Notice from the discussion in section 5.2 that there is either zero or one event occurrence on the event list corresponding to each entity. If there is no event occurrence, the entity is waiting usually for a resource to become available. Multiple entities may be waiting for the same resource. Thus, it is necessary to maintain lists of waiting entities as well as lists of event occurrences.

Entities wait for a resource that is currently not in the idle state in an ordered list similar to the event list. When a unit of the resource completes its current task or otherwise becomes idle, it will process the first entity in the list. The list is sequenced either by order of entity entry in the list (first-in-first-out or last-in-first-out) or by an entity attribute value (high-value-first or low-value-first).

Suppose entities in the two workstation model have the following attributes:
1. Time of arrival to the system
2. Estimated processing time at workstation A

Suppose that at a particular moment in simulation time there are three entities waiting for the workstation A resource. The waiting entities are ordered first-in-first-out as follows.

Entity	Time of Arrival	Estimated Processing Time
101	100.0	15.0
102	110.5	9.8
103	120.5	21.0

Alternatively, suppose the entities were sorted by the lowest value of estimated processing time first as follows.

Entity	Time of Arrival	Estimated Processing Time
102	110.5	9.8
101	100.0	15.0
103	120.5	21.0

Note that the sequence in which entities are processed at workstation A is the same as their order in the queue of workstation A.

5.6 Random Sampling from Distribution Functions

In chapter 2, entity attribute values and the time between entity arrivals as well as operation and transportation times were modeled using probability distributions. Furthermore, values for these variables need to be assigned to each entity. To accomplish this assignment, a random sample must be taken from the corresponding probability distribution. This subject is worthy of a lengthy and thorough discussion such as that provided in Law (2007) as well as Carson, Banks, Nelson, and Nicol (2009). Here one approach for taking random samples is presented to illustrate how this issue is addressed.

Consider the time between entity arrivals in the two workstations in a series model: exponential (10) seconds, where 10 is the average time between arrivals, TBA. This quantity follows the cumulative distribution function:

$$y = F(x) = 1 - e^{-x / TBA} = 1 - e^{-x / 10}$$

and therefore

$$x = -TBA \ln (1 - y) = -10 \ln (1-y) \qquad (5\text{-}1)$$

In the same way, the service time at workstation A is uniformly distributed between a minimum and a maximum value (5 and 13 seconds) and therefore follows the cumulative distribution function:

$$y = F(x) = (x\text{-minimum}) / (\text{maximum} - \text{minimum}) = (x - 5) / (13 - 5)$$

and therefore

$$x = y * (\text{maximum} - \text{minimum}) + \text{minimum} = y * (13 - 5) + 5 \qquad (5\text{-}2)$$

Notice that taking the inverse of the cumulative distribution reduces each case to the same problem, determining the value of y. Thus, this approach for taking a random sample is called the **inverse-transformation method.**

Figure 5-2 shows how this method works for the service time at workstation A. Any value of y in the range 0-1 is equally likely. (This is because y is a cumulative distribution.) Good experimental procedure requires a random sample and so a random sample of y must be chosen. Once a random value is selected for y, the random sample of x is straightforward to compute.

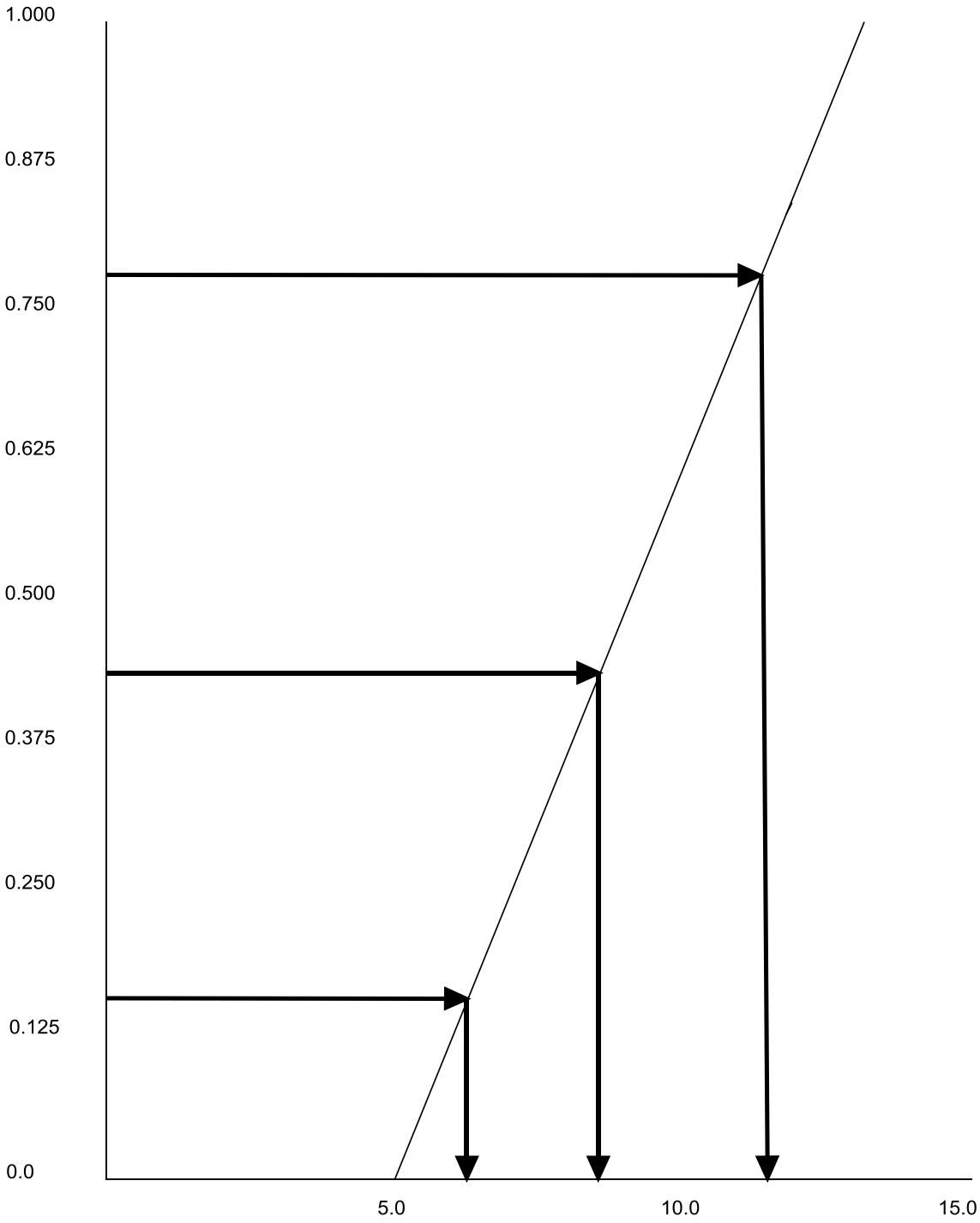


Figure 5-2: Illustration of the Inverse Transformation Method

The inverse-transformation method is summarized as follows:

1. Determine the inverse of the cumulative distribution function, $F^{-1}(x)$.
2. Each time a sample is needed:
   a. Generate a random sample, r, uniformly distributed in the range 0 to 1.
   b. $x = F^{-1}(r)$.

Using the inverse-transformation method requires that the inverse of the cumulative distribution function exists. This is true for the following distributions commonly used in simulation models: uniform, triangular, exponential, weibull, and any discrete distribution where the mass function is enumerated as well as a heuristic distribution in histogram form.

As an example, consider the use of the inverse-transformation method with equation 5-2. Suppose r is selected to be 0.45. Then x = -10 ln (1 – 0.43) = 5.62. Next the inverse-transformation method is applied to equation 5-5. Suppose r is selected to be 0.88. Then x = 0.88 * (13 - 5) + 5 = 12.04.

5.7 Pseudo-random Number Generation

All of the random sampling strategies discussed in the previous section require a random sample uniformly distributed in the range (0,1). Fortunately, there are several well known algorithms for generating such samples, called **pseudo - random** numbers. These algorithms are deterministic. However, the properties of the sequence of pseudo-random numbers make them look random. These properties include the following:

1. The numbers do not exhibit any statistical correlation with each other.
2. The numbers appear to be uniformly distributed in the range (0,1).
3. There are many, many (at least 1,000,000) numbers in sequence.
4. All possible numbers in the sequence are generated before any number repeats.

Because the pseudo-random number generation algorithms are deterministic, a sequence of numbers can be regenerated whenever necessary. This is important in simulation both for debugging and experimentation using common random numbers. Imagine the difficulty of removing a bug from a model if the results were randomly different each time the model was executed!

A sequence of pseudo-random numbers is called a stream. Having multiple streams of random numbers allows sampling from each particular probability distribution used in a model to be associated with a particular stream. For example in the two stations in a series model, the time between arrivals and the operation time at station A would be assigned different streams. This means for example that if the probability distribution modeling the operation time at station A were changed, the times between arrivals would remain the same.

As in the previous section, one approach to pseudo-random number generation will be presented. Other approaches for generating pseudo-random numbers are given in Banks, Carson, Nelson, and Nicol (2009) as well as Law (2007). Schmeiser (1980) provides a comprehensive survey.

Perhaps the most common type of pseudo-random number generation algorithm, with respect to use in simulation languages, is the linear congruential generator (Lehmer, 1951). The linear congruential generator (LCG) has the form:

$$Z_i = (a*Z_{i-1} + c) \bmod(m) \qquad (5\text{-}3)$$

$$r_i = Z_i / m \qquad (5\text{-}4)$$

The Z_i's are a set of integers that range from 0 to m-1. The integer Z_i is a remainder and m is the divisor. Other parameters of the generator are a multiplier a, an increment c, and the first integer Z_0. The pseudo-random number r_i is obtained by dividing Z_i by m. Fortunately for our purposes, values for the parameters (a, c, m, and Z_0) that result in the desirable properties listed above are used by commercial simulation languages.

The generator is recursive that is Z_i is a function of Z_{i-1}. Note that at most, m distinct Z_i's and thus r_i's (pseudo-random numbers) can be obtained. Once Z_0 is generated a second time, the entire sequence of Z_i's, and thus r_i's, will be repeated and in the same sequence as the first time.

Consider the example LCG shown in Table 5-3. The LCG parameter values are shown in the table. Note that the Z_i' s range from 0-8. All nine of the Z_i's are generated before any value repeats. Thus, the r_i's appear to be as uniformly distributed in the range (0,1) as nine numbers can be. The statistical correlation between the r_i's is low, 0.030. Since the number of values generated is only 9, the value of m is too small for an effective LCG. However, it suffices for an example.

Table 5-3: Example LCG

		i	Z_i	r_i
M	9	0	8	0.889
A	4	1	1	0.111
C	5	2	0	0.000
		3	5	0.556
		4	7	0.778
		5	6	0.667
		6	2	0.222
		7	4	0.444
		8	3	0.333
		9	8	0.889
		10	1	0.111
		11	0	0.000
		12	5	0.556

5.8 Summary

This chapter discusses the basic operations of a simulation engine. While these operations are performed transparently to the modeler, an understanding of them helps clarify how simulation experiments work. Events are organized and processed in time sequence. Entities waiting for resources are sorted and maintained. Random samples from distribution functions are generated and pseudo-random number streams are managed.

Problems

1. State a procedure for generating a random sample from each of the following distributions using the inverse transformation method. Use the procedure in section 5.6 as a guide.

   a. Uniform distribution: $F(x) = \dfrac{x - minimum}{maximum - minimum}$

   b. Exponential distribution: $F(x) = 1 - e^{x / mean}$

   c. Weibull distribution: $F(x) = 1 - e^{(-x/c)^m}$ where c and m are the scale and shape parameters of the distribution respectively.

   d. Triangular distribution:

$$F(x) = \begin{cases} \dfrac{(x - minimum)^2}{(mode - minimum) * (maximum - minimum)}, & minimum \le x \le mode \\ 1 - \dfrac{(maximum - x)^2}{(maximum - mode) * (maximum - minimum)}, & mode < x \le maximum \end{cases}$$

   e. Discrete distribution:
   F(x) = 0.1, x = 1
   = 0.4, x = 2
   = 0.6, x = 3
   = 0.9, x = 4
   = 1.0, x = 5

2. Create a new trace based on the one shown in Table 5-2 by adding a entity with ID number 4 that arrives at time 2.0 with a processing time at workstation A of 6.4.

3. Consider the properties of pseudo-random number generators presented in section 5-8. Does property four imply property two?

4. Consider the two workstation in a series model and the last event list shown in section 5-5.

Current Simulation Time: 8.0
Next Simulation Time = Time of first event occurrence in list = 8.0

Event	Time of Occurrence	Entity ID
Entity Arrives to B	8.0	1
Start Service at A	8.0	2
Entity Arrives to A	32.5	3

Use the event graph shown in Figure 5-1 as well as the trace shown in Table 5-2 as a guide.

   a. Show the event list after the processing of the entity arrives to B event for the entity with ID number 1. What single event occurrence was removed from the list? What event occurrences remain on the list? What event occurrences are added to the list?
   b. Show the event list after the first event on the list resulting from 2a is processed. What single event occurrence was removed from the list? What event occurrences remain on the list? What event occurrences are added to the list?

5. Implement a (bad) LCG generator in Excel with the following parameters:

   $a = 5$; $m = 16$; $c = 3$; $Z_0 = 0$.

   Generate the first 20 samples from the generator. Assess its behavior using the four properties in section 5.7.

6. Compute the following table using a spreadsheet.

   a. Generate the two random number streams, corresponding to interarrival time and operation time at station A, for the first ten arriving entities in the two workstation in a series model. Do this by using the random number generator built into your spreadsheet program. In Excel, this would be accomplished by entering the function rand() into each cell of the Pseudo-random Number / Bet. Arrivals and the Pseudo-random Number / Service Time columns.

   b. Use the inverse-transformation method to generate the time between arrivals and service time samples. This means entering equation 5-1 into each cell in the Sample / Bet. Arrivals column and entering equation 5-2 into each cell in the Sample / Service Time column. The corresponding pseudo-random number in the columns should be referenced for each cell.

Table for Problem 6

Entity ID	Pseudo-random Number		Sample	
	Bet. Arrivals	Service Time	Bet. Arrivals	Service Time
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				

7. Considering only the first two entities from the data generated in the solution to number 6, create a trace similar to Table 5-2 for the two workstations in a series model.

Part II
Basic Organizations for Systems

Traditionally, there have been two basic approaches to organizing systems. A serial system processes one, or at most a few, types of customers, parts, or anything else. Each item visits each of the workstations in a predefined sequence. There are many contexts in which serial systems occur. An assembly line may be used to manufacture an automobile. Customers using a fast food restaurant drive through first order food via a microphone, drive to one window to pay, and then proceed to a second window where the food is delivered. Serial systems are discussed in chapter 7.

A job shop processes many different types of jobs. Each job type has a unique route through a set of workstations. Like serial systems, job shops occur in a wide variety of contexts. A job enters the shop and is routed through one or more of the stations of the shop for processing. A customer enters the cafeteria and proceeds in whatever order seems to make sense to the customer through the stations where the various types of food can be acquired.

A simulation based analysis of a job shop evolves through chapter 8 as well as chapter 10 in part III. Chapter 8 discusses how the number of machines at each station in the job shop can be determined. Chapter 10 examines the conversion of the job shop from a push to a pull operating strategy.

Serial lines and job shops consist of multiple workstations. Thus before studying these, a study of a single workstation is presented in Chapter 6. Mathematical models as well as simulation models are employed.

The terminating simulation experiment design is applied in all three application studies. The iterative nature of the simulation process is demonstrated.

The application studies in this part of the book are straightforward. The applications problems follow directly from the corresponding application studies. These provide basic practice in using the simulation process for problem solving including model building, experimentation and analysis as well as the use of simulation environment software. The application studies and problems in the subsequent parts of the book are intended to be more challenging and to show the breadth of the use of simulation.

The application study chapters are intended to be used in the following way. Modeling and experimentation information needed to perform the application problem at the end of the chapter is introduced in and illustrated by the application study. Thus, the application study is presented in a straightforward way. Questions at the end of the chapter aid in the understanding of the application study. Some questions require extensions of the model or the simulation experiment and are labeled as laboratory exercises.

Additional questions at the end of the chapter concern the application problems. These questions raise issues concerning modeling, experimentation, and the interpretation of results that must be resolved in completing the study. The student may perform all of the simulation process steps, including model building, verification and validation, experimentation, analysis of simulation results, and conclusion drawing. Alternatively, the questions may be used simply to discuss how the study should be performed.

Chapter 6

The Workstation

6.1 Introduction

Previously, the effect on part lead time when the current machine at a workstation was replaced with a new machine with less variance in operating time was studied. In this chapter, a more extensive study of a single workstation is performed using both analytic and simulation models in a complementary fashion. The effects of operational detractors, batching with setup, machine downtime, and part reworking are analyzed. Again, such a study can be done before a new machine is acquired to help validate the future state in a lean transformation. The workstation operates on one independently identifiable part at a time. The part resides in an input buffer while waiting for processing.

6.2 Points Made in the Case Study

This case shows how a new workstation, that is a part of a lean transformation, can be studied and its operation validated before implementation by using a combination of analytic and simulation models.

The results of the analytic models are compared to the results of the simulation models to provide validation evidence for the simulation models.

The use of models to quantify performance using both average and maximum values is shown.

The average time between arrivals to a workstation should be equal to the takt time in order to produce the quantity of product demanded by a customer. Variation could result in the time between arrivals being greater than the takt time for any particular part. This could have a short term negative impact on the ability to meet customer demands.

The average processing time at a workstation must be less than the takt time in order to produce the quantity of product demanded by a customer. Variation could result in the processing time being greater than the takt time for some items. This could have a short term negative impact on the ability to meet customer demands.

The iterative nature of the simulation process is shown. A review of the initial study of the workstation leads to a request to include the three detractors in the simulation study, both individually and in combination. These detractors are batching with setup, breakdowns, and rejection and rework of a completed part.

Statistical analysis is used to determine if the detractors have a significant effect on the lead time at the workstation when the detractors are present.

6.3 The Case Study

The case study shows the process of using simulation and analytic models together to address issues concerning a new workstation before acquisition and implementation.

6.3.1 Define the Issues and Solution Objective

A lean team has been studying the operation of a particular workstation. A replacement machine with less variation in processing time, but the same average processing time, has been proposed as a part of a future state definition. Management is requiring a study to determine the average and maximum lead times at the workstation if the replacement machine is acquired.

The workstation operates 168 hours per month. Customer demand per month is 1680 parts or 10 parts per hour, resulting in a takt time of 6 minutes. The processing time for the new machine is triangularly distributed with a mean of 5 minutes, a minimum of 3 minutes and a maximum of 8 minutes. Thus, the mode is 4 minutes. (See the discussion in chapter 3 for the computation of the mode.) Inbound arrival of parts is not well controlled, which will be modeled using the practical worst case: Exponentially distributed with a mean equal to the takt time of 6 minutes.

6.3.2 Build Models

The model in pseudo-English is shown below.

```
Define Arrivals:                    // mean must equal takt time
        Time of first arrival:      0
        Time between arrivals:      Exponentially distributed with a mean of 6 minutes
                                    Exponential (6) minutes
        Number of arrivals:         Infinite // Note: The average number of arrivals is 1680

Define Resources:
        WS/1 with states (Busy, Idle)

Define Entity Attributes:
        ArrivalTime                 // part tagged with its arrival time; each part has its own tag

Process Workstation
Begin
        Set ArrivalTime = Clock                         // record time part arrives on tag
        Wait until WS/1 is Idle in Queue QWS            // part waits for its turn on the machine
        Make WS/1 Busy                                  // part starts turn on machine; machine is busy
        Wait for Triangular (3, 4, 8) minutes           // part is processed
        Make WS/1 Idle                                  // part is finished; machine is idle
        Tabulate (Clock-ArrivalTime) in LeadTime        // keep track of part time on machine
End
```

The definitions tell about arrivals, the machine including its states, and the entity (part) attributes. The comments (denoted by //) describe the steps the part goes through for processing on the machine as well as recording the arrival time and tabulating its individual lead time just before departure.

6.3.3 Identify Root Causes and Assess Initial Alternatives

In this section, the simulation experimental design and results are presented. First, an analytic model of the single work station is discussed in the next section.

6.3.3.1 Analytic Model of a Single Workstation

The time between arrivals is characterized by both a mean, T_a, and by a standard deviation, σ_a and the processing time is characterized by both its mean CT and by its standard deviation, σ_T. The coefficient of variation of the time between arrivals is $c_a = \sigma_a / T_a$. The coefficient of variation of the processing time is $c_T = \sigma_T / CT$.

The average number of parts in the buffer is WIP_q and WIP_q plus the utilization of the machine (equal the average number of parts in processing) is the WIP.

The average time spent at the workstation can be broken into two parts: the average time in processing, CT, and the average time in the buffer, CT_q. Their sum is the total is the lead time LT.

The relationship between WIP, lead time and throughput is known as Little's Law.

Work in process (WIP) = Throughput (TH) X Lead Time (LT) (6-1)

Examples:

Number of parts at a workstation = Parts completed per hour at the work station X
Total time at the workstation

Number of customers at Burger King = Customers served per hour at Burger King X
Time from entry to completion of service at BK

Number of pallets on a holding conveyor = Pallets entering the main line conveyor per hour
X
Time till entry to the holding conveyor to entry to the main line conveyor

Number of units in a transfer center = Number of units entering the transfer center per hour X
Average processing time in the transfer center

Number of students enrolled at GVSU = Number of students entering per year X
Average number of years enrolled at GVSU

Here are some ideas that can be extracted from Little's Law.

1. In order for the WIP (a bad thing to have lots of) to decrease, either the throughput must decrease (for a constant lead time) or the cycle time must decrease for a constant throughput. Since throughput often depends on requirements for finished goods and is the reciprocal of the takt time, decreasing lead time is most likely necessary to decrease WIP.

2. Another way of writing Little's Law is TH = WIP / LT. This means that increasing throughput can be achieved by increasing WIP or decreasing LT. However, increasing WIP (a bad thing to have lots of) may increase lead time. Thus, increasing throughput most often requires decreasing lead time. Note that the same throughput can be achieved with large WIP and large lead times or small WIP and small lead times.

3. A third way of writing Little's Law is LT = WIP / TH. Decreasing LT can be achieved by decreasing WIP or increasing throughput if the WIP does not increase.

Next we will consider all of the information that can be computed about the behavior of a single workstation that has one machine or one worker. We will include **means and variances** in evaluating **average** behavior. Notice that variation in measures of behavior is not, an often cannot, be determined analytically.

Consider the workstation shown in Figure 6-1, with computed quantities shown in **boldface**. Quantities that are known are the time between arrivals (mean and variance) as well as the processing time (mean and variance). Quantities that can be computed are:

1. time in the input buffer of the station (CT_q)
2. lead time at the station (LT)
3. average number of parts in the input buffer (WIP_q)
4. average number of parts at the station (WIP)
5. utilization of the station, the percent of time the workstation is busy processing a part. (μ)
6. time between departures (mean and standard deviation) (T_d and σ_d)

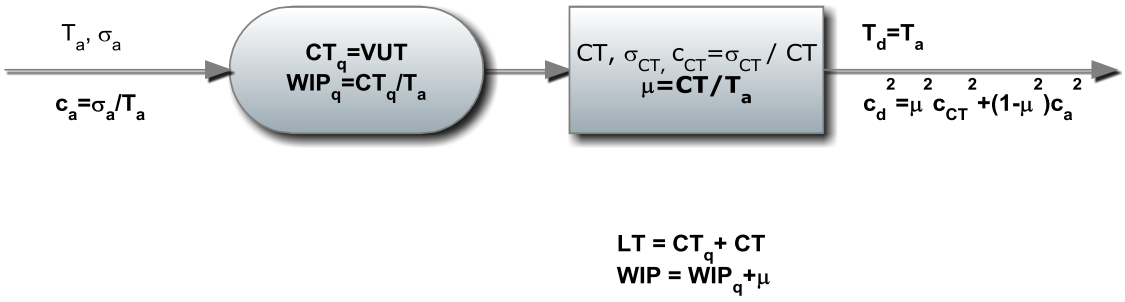


Figure 6-1: Computation of Workstation Quantities

Equation 6-2 is called the VUT equation, for Variance – Utilization – Time and is used to approximate the average cycle time in the queue. This equation is presented and further discussed in Hopp and Spearman (2007).

$$CT_q \approx VUT = \left(\frac{c_a^2 + c_{CT}^2}{2}\right)\left(\frac{\mu}{1-\mu}\right)CT \qquad (6\text{-}2)$$

The following insights can be gained by examining equation 6-2.

1. The cycle time in the buffer depends on the variance of the time between arrivals and the variance of the processing time, expressed as the squared coefficient of variation. As the variance of either increase, the average cycle time in the queue increases. The coefficient of variation is the standard deviation / mean.
2. The cycle time in the buffer increases in a highly non-linear fashion as the utilization increases. The utilization term for a utilization of 90% is 9, for a utilization of 95% is 19, and for a utilization of 99% is 99.
3. The only way to effectively run a workstation with high utilization is to eliminate the variation in the time between arrivals and the processing time.
4. A utilization of 100% cannot be achieved unless the variance in both the processing time and the time between arrivals is zero.
5. The mean and the standard deviation of the exponential distribution are equal. Thus, the coefficient of variation for an exponential distribution is equal to 1. Thus, the "good" range for the V term is 0 to 1.
6. The distributions of the time between arrivals and the processing times are not required, only the mean and the standard deviation.

Once the average cycle time in the buffer is determined, the average number in the buffer can be determined using Little's Law:

$$WIP_q = CT_q * \frac{1}{T_a} \qquad (6\text{-}3)$$

The lead time at the station is simply the cycle time in the buffer plus the processing time:

$$LT = CT_q + T \qquad (6\text{-}4)$$

The number at the station can be obtained from equation 6-4 using Little's Law:

$$WIP = LT * \frac{1}{T_a} = \left(CT_q + T\right) * \frac{1}{T_a} = WIP_q + \mu \qquad (6\text{-}5)$$

The mean of the time between departures should equal the mean of the time between arrivals. This is simply a law of the conservation of parts: All entering parts must depart. The conservation law applies between workstations as well: The mean and variation of the time between departures from one workstation are the same as the mean and variation of the time between arrivals to the next work station.

The squared coefficient of determination of the time between departures is given by equation 6-6:

$$c_d^2 = u^2 c_T^2 + (1 - u^2) c_a^2 \qquad (6\text{-}6)$$

The following insights can be gained by examining equation 6-6.

1. The variation in the departures for a high utilization workstation depends mostly on the variation in the processing time. Thus, a low variation processing time results in a low variation in the departures, which results in a low variation in the arrivals to the next workstation.
2. A workstation with high utilization and low variation in processing time will, to a great extent, eliminate high variation in the time between arrivals.
3. A workstation with high utilization and high variation in processing time will cause high variation in the time between arrivals to the next station. Thus, the cycle time in the buffer at the next station will tend to be high.
4. A workstation with low utilization will tend to result in the variation of the time between arrivals at the next workstation equaling the variation in the time between departures at the current workstation.

The results from the analytic model of the workstation of interest are shown in Table 6-1.

Table 6-1: Analytic Model of Workstation – Results

Inputs	Average time between arrivals	6
	Average processing time	5
	Std. Dev. time between arrivals	6
	Std. Dev. processing time	3
Utilization	Utilization	83.3%
Average Times	c_a -- Time between arrivals	1
	c_T -- Processing time	0.6
	Variance term	0.68
	Utilization term	5.0
	Average time in buffer	17.0
	Average lead time	22.0
Average Number of Parts	Average number in the buffer	2.8
	Average number at the station	3.7
Departure Information	Average time between departures	6
	c_d^2 -- Time between departures	0.56

Note that the inputs to the analysis are the average and standard deviation of the time between arrivals as well as the average and standard deviation of the processing time. The averages are typically obtained through value stream mapping. The standard deviations must typically be obtained through additional data collection and analysis.

6.3.3.2 Simulation Analysis of the Single Workstation

Next, the design for the simulation experiment must be specified. This design will make use of the results of the analytic model.

The experimental design contains the elements discussed in chapter 4. This is a terminating simulation of duration 168 hours, the monthly planning period. There are two streams required, one for the time between arrivals and one for the processing time. The initial conditions are set based the results of the analytic model: 2 parts in the buffer and thus one additional part on the machine. Lead time is the primary performance measure of interest. Some of the other quantities computed by the analytic model are also of interest: utilization of the machine and average number of parts in the buffer. These will be used in obtaining validation evidence for the simulation model and experiment. Twenty replications will be performed. There are no model parameters in the first experiment.

In summary, the experiment design is as follows:

Table 6-2: First Simulation Experiment Design for the Workstation.

Element of the Experiment	**Values for This Experiment**
Type of Experiment	Terminating
Model Parameters and Their Values	None
Random Number Streams	1. Time between arrivals 2. Operation time
Performance Measures	1. Part lead time 2. Utilization of the machine 3. Number of parts waiting for the machine in the buffer
Number of Replicates	20
Initial Conditions	2 parts in the buffer implying one part on the machine
Simulated Time Interval (Beginning time – ending time)	0 – 168 hours

Verification evidence is obtained using the balance equation:

Number of entities entering =
Number of entities leaving + Number of entities remaining at the end

The number of entities entering the model is the sum of those arriving and the initial entities. Thus for the first replicate:

1717 + 3 = 1719 + 1.

Thus, validation evidence is obtained.

Table 6-3 shows the simulation results. These results are consistent with the results from the analytic model. The 99% confidence intervals for the utilization, the average number of parts

waiting at the station, and the average lead time all contain the corresponding values resulting from the analytic model. Thus, model validation evidence is obtained.

Table 6-3: Simulation Results for Base Experiment

Replicate	Average Number at Station	Average Lead Time	Max Lead Time	Utilization
1	3.20	18.80	74.70	0.86
2	2.99	18.11	79.15	0.83
3	3.40	20.36	83.38	0.83
4	3.70	21.15	61.29	0.88
5	2.28	14.13	54.11	0.80
6	2.76	16.79	61.17	0.82
7	3.39	20.17	75.93	0.84
8	2.57	15.66	60.89	0.82
9	3.17	18.52	79.15	0.86
10	3.36	20.34	97.58	0.83
11	2.92	16.99	60.50	0.86
12	4.51	26.07	104.61	0.87
13	3.11	18.97	81.80	0.82
14	2.63	16.46	67.19	0.80
15	3.23	18.89	75.20	0.86
16	2.75	16.62	90.52	0.82
17	2.56	15.49	62.29	0.83
18	2.53	15.66	68.35	0.80
19	5.52	31.55	138.89	0.88
20	4.45	25.75	129.22	0.87
Average	3.25	19.32	80.30	0.84
Std. Dev.	0.79	4.24	22.63	0.02
99% CI Lower Bound	2.74	16.61	65.82	0.82
99% CI Upper Bound	3.76	22.03	94.77	0.85

6.3.4 Review and Extend Previous Work

Management reviewed the simulation model and experiment, concluding that the model was a validated as a tool for assessing the future state before implementation.

The lead time results from the models were of concern with respect to how the workstation would operate. The average lead time was about four times the processing time. The average maximum lead time estimated by the simulation model was about 16 times the processing time. These high values are entirely due to the variation in the arrival of inbound parts or orders (expressed by modeling the time between arrivals as exponentially distributed) as well as the variation in processing time as seen in the coefficient of variation of 0.60. Thus, reducing this variation by identifying and addressing root causes seems fundamental to making the workstation operation leaner.

At the review meeting, a request was made to assess the effects of three detractors on workstation performance. The assessment of each was to be made independently of the others.

The nature of these detractors is discussed in the following section.

6.3.4.1 Detractors to Workstation Performance

The first detractor is breakdowns. Breakdowns reduce the amount of available production time. A period of operation for a single machine ends in a breakdown. The length of this period is highly variable. Some time is needed to repair the machine, which could vary by the type of breakdown. This breakdown-repair cycle repeats as shown in Figure 6-2.

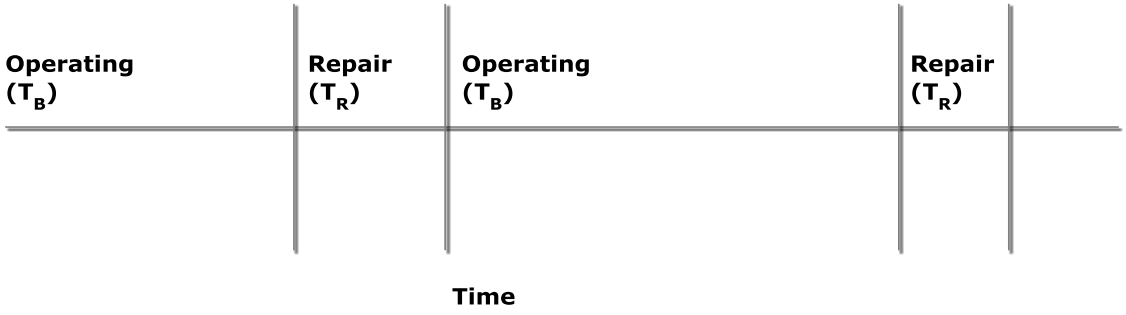


Figure 6-2: Operation and Repair Cycle

Let T_B denote the average time between the end of a repair and the next breakdown and T_R denote the average repair time. Then the quantity $T_B + T_R$ is the length in time of the breakdown-repair cycle. The availability is defined as the percent of time the machine is not broken and is computed as follows.

$$A = \frac{T_B}{T_B + T_R} \qquad (6\text{-}7)$$

The following should be noted concerning availability:

1. The time to complete all work (operations on parts) is reduced to A% of the original time.
2. The lead time for parts waiting for the workstation while it is being repaired will be much longer than for parts that don't wait for a repair. Thus, the average, maximum, and standard deviation of the lead time will increase.

In this case, the machine breaks down on the average once per week (40 hours) and takes between 30 minutes and 2 hours to repair. Since the time between breakdowns is highly variable, it is modeled as exponentially distributed with a mean of 40 hours. The time to repair is modeled as uniformly distributed between 30 minutes and 2 hours (120 minutes).

The second detractor is defective parts. Either additional parts need to be made or the defective parts need to be reworked to meet the demand. This increases the amount of work that needs to be done to produce the number of parts needed to meet the demand. If additional parts need to be made or the average rework time is the same as the average production time, the number of parts that need to be made is given by equation 6-8 where p is the percent of parts that are defective.

$$D_{New} = D_{Old} / (1\text{-}p) \qquad (6\text{-}8)$$

The following should be noted concerning defective parts.

1. The increase in work will increase the utilization of the workstation, which in turn increases the lead time as shown in the VUT equation (6-2).
2. Effectively, there are more arrivals to the workstation which decreases the time between arrivals to TBA * (1-p).

In the case, let p = 5%.

The final detractor is setup and the resulting batching of parts. The setup and batching process is as follows. As they arrive, parts are gathered into a group called a batch until the number of parts in the group equals the predetermined batch size (b). The newly formed batch enters the buffer of the machine to wait processing. Processing the batch means performing a setup operation on the machine and then processing all items in the batch.

The following should be noted concerning setup and batching.

1. Waiting for a batch to form will increase the average, maximum, and standard deviation of lead time.
2. The following must be true: b*takt time >= setup time + b * operation time
3. The minimum feasible batch size may be greater than one, given the preceding item in the list.

This leads to the following question: What is the smallest value of the batch size such that the utilization, which now includes the setup time, is as close as possible to a given value? Decreasing the batch size increases the number of setups and thus the amount of time spent doing setup work, which is not productive. However, decreasing the batch size decreases the work in process and finished goods inventories and supports a more flexible production schedule. These goals are consistent with achieving a lean production environment.

The utilization should be computed as shown in equation 6-9.

μ = (setup time + b * CT) / (b * T_a) (6-9)

Then the smallest batch size for a given value of the utilization is given by equation 6-9a.

$$b = setup\ time/(u * T_a - CT) \quad (6\text{-}9a)$$

This problem also can be formulated and solved using a spreadsheet to facilitate evaluating alternative values of the batch size. These alternatives could include complying with constraints such as the batch size must be a multiple of 10. The evaluation is done by computing the utilization and number of batches as a function of the selected batch size.

The target utilization is entered. The absolute deviation between the actual utilization and the target is minimized. In other words, the batch size n is changed until the actual utilization is as close as possible to the target. This may be done manually or with one of the spreadsheet tools: solver and goal seek. Note: if goal seek is used, start with a very small batch size.

In this case, the target utilization is set to 95% and the setup time is 30 minutes. Table 6-4 shows the how a batch size of 42 was determined. Note the value of the batch size computed using equation 6-9a is 42.9.

Table 6-4: Result of Finding a Target Batch Size

Inputs	Target Utilization	95%
	Average Time Between Arrivals	6
	CT	5
	Setup Time	30
	Demand	1680
Result	Batch size (b)	42
Computations	Numerator	240
	Denominator	252
	Utilization	95.2%
	Deviation	0.2%
	Number of Batches	40

6.4 The Case Study for Detractors

The simulation process is restarted to analyze the effect on lead time of each detractor separately.

6.4.1 Define the Issues and Solution Objective

The effect of random downtimes on part lead time for parts processed by the workstation is to be assessed. As discussed in the previous section, the time between breakdowns is modeled as exponentially distributed with a mean of 40 hours and the time to repair is modeled as uniformly distributed between 30 and 120 minutes.

The effect of defective parts on part lead time at the workstation is to be assessed. Five percent of completed parts are found to be effective and must be reworked. The time for reworking is the same as the original processing time.

The effect of setup and batching on part lead time at the workstation is to be assessed. For a target utilization of 95%, the batch size was determined to be 42 parts.

6.4.2 Build Models

The modeling of random downtimes was discussed in chapter 2 and will not be repeated here. To summarize, recall a distinct process is created which models an ongoing cycle of breakdowns and repairs. After the time between breakdowns, the resource representing the workstation enters the broken state from the idle state. After the time for repair, the resource enters the idle state to be ready to process another part.

The model in section 6.3.2 is modified as follows to include defective parts. After a part has completed processing, it is identified as needing rework with probability 5%. If rework is needed, the part is sent back to start the workstation process over again. Parts now arrive to Process Arrive where the arrival time is set. This avoids resetting the arrival time for defective parts.

```
// Workstation model with defective parts

Define Arrivals:                    // mean must equal takt time
        Time of first arrival:      0
        Time between arrivals:      Exponentially distributed with a mean of 6 minutes
                                    Exponential (6) minutes
        Number of arrivals:         Infinite // Note: The average number of arrivals is 1680

Define Resources:
        WS/1 with states (Busy, Idle)

Define Entity Attributes:
        ArrivalTime                 // part tagged with its arrival time; each part has its own tag

Define State Variable
        PercentDefective = 0.05     // Percent of defective parts

Process Arrive
Begin
        Set ArrivalTime = Clock                 // record time part arrives on tag
        Send to Process Workstation             // start processing
End

Process Workstation
Begin
        Wait until WS/1 is Idle in Queue QWS    // part waits for its turn on the machine
        Make WS/1 Busy                          // part starts turn on machine; machine is busy
        Wait for Triangular (3, 4, 8) minutes   // part is processed
        Make WS/1 Idle                          // part is finished; machine is idle
        If (Uniform(0,1) < 0.05) then
                Send to Process Workstation             // part is defective rework
        Tabulate (Clock-ArrivalTime) in LeadTime        // keep track of part time on machine
End
```

The model in section 6.3.2 is modified as follows to include batching and setup. Entities in the workstation process now represent batches. Thus, an entity is not sent from the arrival process to the workstation process until a batch is formed. In the arrival process, the first 41 entities in the batch wait on a list. The 42nd entity is sent to the workstation process. The time delay in the workstation process is now the setup time plus 42 different samples of the processing time. After the batch is processed, all 42 entities go to the depart process so that each lead time can be recorded.

```
// Workstation model with batching and setup

Define Arrivals:                    // mean must equal takt time
        Time of first arrival:      0
        Time between arrivals:      Exponentially distributed with a mean of 6 minutes
                                    Exponential (6) minutes
        Number of arrivals:         Infinite // Note: The average number of arrivals is 1680

Define Resources:
        WS/1 with states (Busy, Idle)

Define Entity Attributes:
        ArrivalTime                 // part tagged with its arrival time; each part has its own tag

Define State Variable
        BatchSize = 42              // Percent of defective parts
        SetupTime = 30              // Setup time
        Processed                   // Number of parts processed

Define List
        BatchList                   // List of parts waiting for batching

Process Arrive
Begin
        Set ArrivalTime = Clock                         // record time part arrives on tag
        If TotalArrivals(Arrive) %  BatchSize ! = 0 then
                Add entity to list BatchList            // stop entity processing for now
        Send to Process Workstation                     // start processing
End

Process Workstation
// An entity in this process represents a batch
Begin
        Wait until WS/1 is Idle in Queue QWS    // batch waits for its turn on the machine
        Make WS/1 Busy                          // part starts turn on machine; machine is busy
        Wait for SetupTime                      // Setup machine

        // Processing time for batch as sum of processing times for each part
        Processed = 0
        do while Processed < Batchsize
        Begin
                Wait for Triangular (3, 4, 8) minutes       // part is processed
                Processed++
        End

        Make WS/1 Idle                          // part is finished; machine is idle
        // Send all entities in batch to record lead time
        Send Batchsize – 1 from BatchList to Process Depart
        Send to Process Depart
End

Process Depart
        Tabulate (Clock-ArrivalTime) in LeadTime        // keep track of part time on machine
End
```

6.4.3 Assessment of the Impact of the Detractors on Part Lead Time

The experimental design shown in Table 6-2 can be used with additions as follows:

1. For the case of breakdowns, add two random streams: one for the time between breakdowns and the other for the time till repair.
2. For the case of defective parts, add a random stream for determining whether or not parts are defective.
3. For setup and batching, no additions are needed.

Table 6-5 shows the simulation results assessing the effect on lead time of breakdowns.

Table 6-5: Simulation Results for Breakdown Experiment

Replicate	Average Number at Station	Average Lead Time	Maximum Lead Time	Utilization
1	4.33	25.41	94.28	0.86
2	5.28	32.03	186.20	0.83
3	3.49	20.90	83.38	0.83
4	8.48	48.58	270.39	0.87
5	3.91	24.29	138.08	0.80
6	3.67	22.36	116.80	0.82
7	3.89	23.15	141.18	0.84
8	3.51	21.39	121.25	0.82
9	6.07	35.46	124.02	0.86
10	5.40	32.74	164.23	0.83
11	4.30	25.06	149.75	0.86
12	14.79	85.83	264.19	0.86
13	3.57	21.78	91.78	0.82
14	4.52	28.31	134.16	0.80
15	4.14	24.25	148.00	0.86
16	2.76	16.71	90.52	0.82
17	7.06	42.84	213.73	0.83
18	3.41	21.14	128.25	0.80
19	10.59	60.52	218.28	0.88
20	7.17	41.47	221.30	0.86
Average	5.52	32.71	154.99	0.84
Std. Dev.	2.94	16.68	56.37	0.02
Lower Bound	3.64	22.04	118.93	0.82
Upper Bound	7.40	43.38	191.05	0.85

Note that there is an operational significant increase in the average number at the station, the average lead time, and the maximum lead time versus the base experiment. Notice the increase in the standard deviation of these quantities as well.

This statistical significance of this difference is confirmed for the average lead time using the paired-t test shown in Table 6-6.

Table 6-6: Paired-t test for Difference in Average Lead Time

	Average Lead Time		
Replicate	Base	Breakdowns	Increase
1	18.80	25.41	6.62
2	18.11	32.03	13.92
3	20.36	20.90	0.54
4	21.15	48.58	27.42
5	14.13	24.29	10.15
6	16.79	22.36	5.56
7	20.17	23.15	2.98
8	15.66	21.39	5.73
9	18.52	35.46	16.94
10	20.34	32.74	12.40
11	16.99	25.06	8.07
12	26.07	85.83	59.76
13	18.97	21.78	2.81
14	16.46	28.31	11.85
15	18.89	24.25	5.36
16	16.62	16.71	0.09
17	15.49	42.84	27.36
18	15.66	21.14	5.48
19	31.55	60.52	28.97
20	25.75	41.47	15.71
Average	19.32	32.71	13.39
Std. Dev.	4.24	16.68	13.96
Lower Bound	16.61	22.04	4.46
Upper Bound	22.03	43.38	22.31

The application of the paired-t test to the other simulation results is left as an exercise for the reader.

Table 6-7 shows the simulation results for the case of defective parts.

Table 6-7: Simulation Results for Defects Experiment

Replicate	Average Number at Station	Average Lead Time	Max Lead Time	Utilization
1	4.42	25.96	92.01	0.90
2	3.93	23.78	109.29	0.86
3	4.95	29.68	115.86	0.89
4	6.25	35.80	116.02	0.92
5	2.76	17.08	57.05	0.83
6	3.71	22.61	89.94	0.87
7	4.34	25.81	86.87	0.88
8	3.23	19.67	69.43	0.86
9	4.56	26.64	96.35	0.90
10	4.46	27.02	105.34	0.87
11	3.51	20.49	70.55	0.89
12	6.45	37.28	151.03	0.91
13	4.59	27.99	120.01	0.87
14	3.77	23.62	110.85	0.84
15	5.55	32.52	97.14	0.91
16	3.44	20.82	109.64	0.87
17	4.22	25.56	107.08	0.87
18	3.08	19.13	74.60	0.84
19	9.26	52.91	210.05	0.92
20	7.47	43.21	202.32	0.91
Average	4.70	27.88	109.57	0.88
Std. Dev.	1.61	8.83	39.26	0.03
Lower Bound	3.67	22.23	84.46	0.86
Upper Bound	5.73	33.53	134.69	0.90

The average and maximum lead times have increased noticeably. The utilization has increased as would be expected as has the average number at the station. Recall from the VUT equation that average time in the buffer increases in a highly non-linear fashion as the utilization increases.

Table 6-8 shows the simulation results for the case of setting up the machine and batching parts.

Table 6-8: Simulation Results for the Batching and Setup Experiment

Replicate	Average Number at Station	Average Lead Time	Max Lead Time	Utilization
1	1.07	385.04	569.01	0.95
2	1.04	396.15	600.01	0.92
3	1.02	387.67	574.46	0.92
4	1.34	447.70	725.87	0.96
5	0.92	372.75	576.90	0.89
6	1.02	388.70	601.50	0.92
7	1.07	394.96	584.17	0.94
8	0.99	380.84	570.58	0.92
9	1.07	388.28	610.56	0.96
10	1.05	396.50	587.92	0.92
11	1.19	416.94	610.85	0.95
12	1.29	443.08	720.79	0.96
13	1.01	387.51	609.96	0.90
14	0.96	385.28	576.22	0.89
15	1.22	429.32	685.80	0.95
16	0.98	380.15	591.23	0.92
17	0.99	385.66	581.59	0.92
18	0.98	391.68	586.99	0.89
19	1.86	579.34	950.53	0.97
20	1.24	425.28	696.60	0.97
Average	1.12	408.14	630.58	0.93
Std. Dev.	0.21	45.65	90.67	0.03
Lower Bound	0.98	378.94	572.58	0.91
Upper Bound	1.25	437.35	688.58	0.95

The following effects of setup and batching can be noted in table 6-8.

1. The average number at the station now represents batches instead of individual parts. The average value of 1.12 indicates that on the average a new batch forms in a shorter time than it takes to process the preceding batch.
2. The average and maximum lead times of a part increase greatly versus the case with no detractors reflecting the time to form a batch and setup time.
3. The utilization is consistent with that specified to find the best batch size, 95%.

6.5 Summary

This chapter discusses a beyond lean analysis of the operation of a single workstation, both with and without operations detractors: breakdowns, part reworking, as well as setup and batching. An analytic model is used to compute the workstation utilization as well as the average time and number of parts in the buffer of the workstation for the case of no detractors. This model provides validation evidence for a simulation model of the workstation which estimates the same quantities plus the maximum lead time. The different replications of the simulation experiment show a wide range of different system behavior possibilities and the corresponding performance measure values. Details of system behavior could be extracted from the simulation as well.

Simulation models and experiments were conducted individually for each detractor. Results were compared to the no detractors case. An analytic model was used to set the best batch size given a utilization of 95%.

Problems

1. Perform a complete comparison of the breakdowns case to the no detractors case using paired-t statistical tests.
2. Perform a complete comparison of the part reworking case to the no detractors case using paired-t statistical tests.
3. Perform a complete comparison of the setup and batching case to the no detractors case using paired-t statistical tests.
4. Find the best batch size for a target utilization of 95% for a workstation with average time between arrivals of 10 minutes, cycle time of 9 minutes, and setup time of 1 hour. Production is 1000 parts.
5. Based on the simulation results that follow, provide validation evidence for a model of a single workstation with utilization of 80%

Replicate	**Utilization**
1	80.2%
2	79.5%
3	80.4%
4	80.6%
5	79.2%

6. Based on the simulation results that follow, provide verification evidence for a model of a single workstation.

Initial items:	10
Items remaining at the end of the simulation:	15
Arriving items:	150
Departing items:	145

7. Consider a single server workstation for which the average time between arrivals is 10 minutes and the average processing time is 9 minutes. Suppose a group modeling the workstation is trying to determine the distributions for the time between arrivals and the processing time in absence of data. Use the VUT equation to determine the average waiting time in the queue for the following possibilities.

	Time Between Arrivals	**Processing Time**
a.	Exponential	Exponential
b.	Constant	Exponential
c.	Exponential	Uniform (6, 12)
d.	Constant	Uniform (6, 12)
e.	Exponential	Triangular (6, 9, 12)
g.	Constant	Triangular (6, 9, 12)
h.	Exponential	Triangular (6, 7, 14)
i.	Constant	Triangular (6, 7, 14)

Case Problem

A new workstation is being designed and a complete analysis is needed as described in this chapter. The workstation operates 168 hours per month. Parts are modeled as arriving according to an exponential distribution with mean 10 minutes. Processing time is uniformly distributed between 6 and 9 minutes.

Detractors are as follows.

Breakdowns: The average time between breakdowns is 40 hours. Repair time is uniformly distributed between 30 and 150 minutes.

Defective parts: Five percent of parts are defective and require rework.

Setup and batching: The setup time is 45 minutes. A utilization of 95% is targeted. The best batch size should be determined.

First perform a complete study of the new workstation with no detractors. Use an analytic model as well as a simulation model and experiment. Part lead time is the primary performance measure. Verification and validation evidence for the simulation model must be obtained.

Second, use a simulation model and experiment to assess the joint effect of all three detractors. Verification and validation evidence should be obtained.

How to do this case study will be described in tutorial style for the simulation environment that you are using in a separate document.

Chapter 7

Serial Lines

7.1 Introduction

A workstation performs one, or one set of, operations on an item. Multiple workstations are required to perform all operations necessary to produce a finished product or perform a required service. Suppose one, or at most a few, types of items need to be processed and that this can be accomplished by processing all items in the same sequence. In this case, a set of single workstations organized into a serial line is appropriate, possibly with a material handling device moving items between the workstations.

In Figure 7-1, a material handling device such as a conveyor lies between seven workstations that comprise a serial line. Items enter at the far left and depart at the far right. Each workstation in left to right sequence performs particular operations on the item. A finished product leaves the system after completing the operation at the right most workstation.

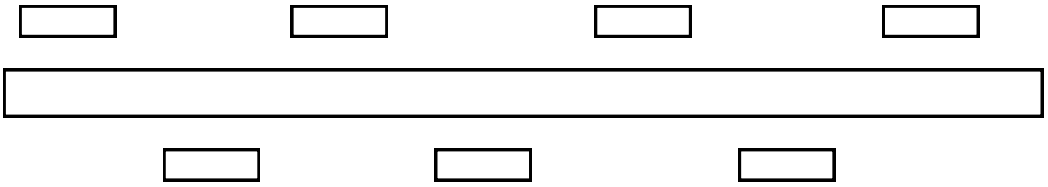


Figure 7-1: Typical Serial Line

An assembly line is one kind of serial system that typically employ human workers. A transfer line is another type of serial system whose workstations consist of automatic machines with a common control system.

In a lean system, the work to be done in processing an item would be balanced between the workstations. The the average operating time at each workstation should be as close to the same as possible. However, it is possible that these average times vary or due to random variation the processing time for an individual part varies between the workstations and the operation time for different parts vary at any particular workstation.

Thus when a workstation completes its operation on an item, the following workstation may still be processing another item. By placing the completed item in a buffer between the stations, the preceding workstation may begin processing another item. If the buffer is full, the preceding station has no where to place the completed item and cannot begin working on another. In this case, the workstation is in the BLOCKED state. Time in the BLOCKED state is unproductive and may increase lead time. Preventing blocking requires buffer space that may be scarce (or a reduction in variability). Thus, minimizing lead time and minimizing buffer between workstations space trade off against each other.

7.2 Points Made in the Case Study

This application study shows how the trade-off between competing factors such as minimizing lead time and minimizing buffer space can be assessed using simulation.

Generally accepted analytic and experimental results are applied in the design of a first simulation experiment. Justification for using equally sized buffers is based on these results. Performance measure values from this first experiment are used in designing additional experiments.

Sometimes a statistically significant improvement in system performance is not operationally meaningful. Doubling the buffer size leads to a small reduction in lead time. Though statistically significant, the amount of the increase is not operationally meaningful.

The benefit of evolving new models from existing models is shown. Models of serial systems are evolved from the model of the single workstation presented in chapter 6.

7.3 The Case Study

In this application study, we consider a serial line where it is essential to minimize buffer space between workstations without compromising lead time.

7.3.1 Define the Issues and Solution Objective

A new three workstation serial line process is being designed for an electronics assembly manufacturing system. The line produces one type of circuit card with some small design differences between cards allowed. The line must meet a lead time requirement that is yet undetermined. Circuit cards are batched into relatively large trays for processing before entering the line.

Only a small amount of space is available in the plant for the new line, so inter-station buffer space must be kept to a minimum. A general storage area with sufficient space can be used for circuit card trays prior to processing at the first workstation. Engineers are worried that the small inter-station buffer space will result in severe blocking that will prevent the lead time target from being reached. The objective is to determine the relationship between buffer size and lead time noting the amount of blocking that occurs. This knowledge will aid management in determining the final amount of buffer space to allocate. Management demands that production requirements be met weekly. Overtime will be allowed if necessary.

The layout of the line is shown in Figure 7-2. A single circuit card is used to represent a tray. All cards in the tray are processed simultaneously by each machine. Three circuit card trays are shown waiting in the general storage area for the solder deposition station, which is busy. The component placement station is idle and its buffer is empty. The solder reflow station is busy with one circuit card tray waiting.

The time between the arrival of circuit card trays to the first workstation is exponentially distributed with mean 3.0 minutes. The processing time distribution is the same at each workstation: uniformly distributed between 0.7 and 4.7 minutes.[1] This indicates that the line is balanced, as it should be.

[1] Alternatively, some simulation languages such as Automod express the uniform distribution as mean and half range instead of minimum and maximum. Thus, a uniform distribution between 0.7 and 4.7 can be equivalently expressed as 2.7 ± 2.0.

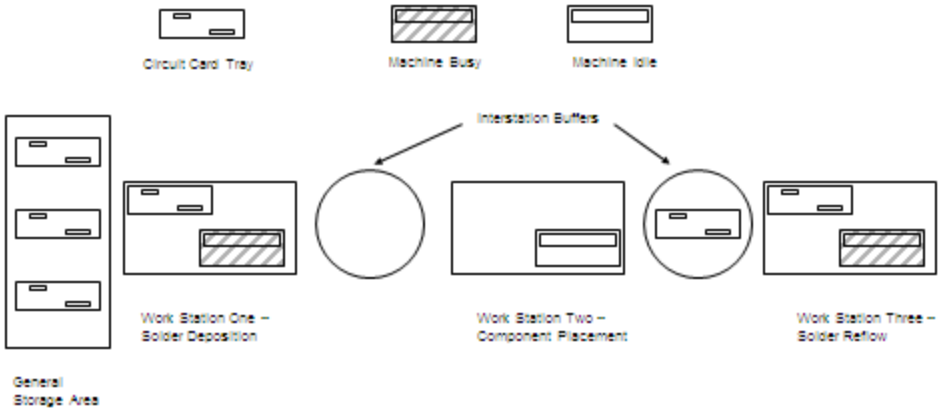


Figure 7-2: Three Station Serial System for Electronics Assembly Manufacturing

7.3.2 Build Models

The model of the serial system includes the arrival process for circuit cards, operations at the three stations, and tray movement into and out of the two inter-station buffers.

There are four entity attributes. One entity attribute is arrival time to the system, ArrivalTime. The other three entity attributes store the operation times for that particular entity at each station.

Assigning all operating times when the entity arrives assures that the first entity has the first operating time sample at each station, the second the second and so forth. This assignment is the best way to ensure that the use of common random numbers is most effective in reducing the variation between system alternative scenarios, which aids in finding statistical differences between the cases. In general, the nth arriving entity may not be the nth entity processed at a particular station. In the serial line model, this will be the case since entities can't "pass" each other between stations.

The processing of a circuit card tray at the deposition station is done in the following way. A circuit card tray uses the station when the latter becomes idle. The operation is performed. The circuit card tray then must find a place in the inter-station buffer before leaving the deposition station. Thus, the deposition station enters the blocked state until the circuit card tray that has just been processed enters the inter-station buffer. Then the station enters the idle state. This logic is repeated for the placement station. The tray departs the line following processing at the reflow station.

The pseudo-English for the model follows. Note that there is one process for arriving entities, one for departing entities and one for each of the three workstations.

```
// Serial Line Model

Define Arrivals:
        Time of first arrival:      0
        Time between arrivals:      Exponentially distributed with a mean of 3 minutes
                                    Exponential (3) minutes
        Number of arrivals:         Infinite

Define Resources:
        WS_Deposition/1             with states (Busy, Idle, Blocked)
        WS_Placement/1              with states (Busy, Idle, Blocked)
        WS_Reflow/1                 with states (Busy, Idle)
        WS_BufferDP/?               with states (Busy, Idle)
        WS_BufferPR/?               with states (Busy, Idle)

Define Entity Attributes:
        ArrivalTime                 // part tagged with its arrival time; each part has its own tag
        OpTimeDes                   // operation time at deposition station
        OpTimePlace                 // operation time at placement station
        OpTimeReflow                // operation time at reflow station

Process Arrive
Begin
        Set ArrivalTime = Clock                         // record time part arrives on tag
        Set OpTimeDes       = uniform(0.7, 4.7)[2]      // set operations times at each station
        Set OpTimePlace     = uniform(0.7, 4.7)
        Set OpTimeReflow    = uniform(0.7, 4.7)
        Send to Process Deposition                      // start processing
End

Process Deposition
// Deposition Station
Begin
        Wait until WS_Deposition/1 is Idle in Queue Q_Deposition  // wait its turn on the machine
        Make WS_Depostion/1 Busy            // tray starts turn on machine; machine is busy
        Wait for OpTimeDes minutes          // tray is processed
        Make WS_Deposition/1 Blocked        // tray is finished; machine is Blocked

        Wait until WS_BufferDP/1 is Idle    // wait for interstation buffer space
        Make WS_Deposition/1 Idle           // free deposition machine
        Send to Process Placement           // Send to placement machine
End

Process Placement
// Placement Station
Begin
        Wait until WS_Placement/1 is Idle   // wait its turn on the machine
        Make WS_BufferDP/1  Idle            // leave interstation buffer
        Make WS_Placement/1 Busy            // tray starts turn on machine; machine is busy
        Wait for OpTimePlace minutes        // tray is processed
        Make WS_Placement/1 Blocked         // tray is finished; machine is Blocked
        Wait until WS_BufferPR/1 is Idle    // wait for interstation buffer space
        Make WS_Placement/1 Idle            // free placement machine
```

[2] In Automod this would be written using the midpoint, half-range style: uniform 2.7, 2.0

```
        Send to Process Reflow                  // Send to reflow machine
End
Process Reflow
// Reflow Station
Begin
        Wait until WS_Reflow1 is Idle           // wait its turn on the machine
        Make WS_BufferPR/1  Idle                // leave interstation buffer
        Make WS_Reflow/1 Busy                   // tray starts turn on machine; machine is busy
        Wait for OpTimeReflow minutes           // tray is processed
        Make WS_Reflow/1 Idle                   // free placement machine
        Send to Process Depart                  // Send to reflow machine
End

Process Depart
        Tabulate (Clock-ArrivalTime) in LeadTime        // keep track of part time on machine
End
```

7.3.3 Identify Root Causes and Assess Initial Alternatives

The experiment design is summarized in Table 7-1.

Table 7-1: Simulation Experiment Design for the Serial System

Element of the Experiment	Values for This Experiment
Type of Experiment	Terminating
Model Parameters and Their Values	Size of each buffer -- (1, 2, 4, 8, 16)
Performance Measures[3]	1. Lead Time 2. Percent blocked time depostion station 3. Percent blocked time placement station
Random Number Streams	1. Time between arrivals 2. Operation time deposition station 3. Operation time placement station 4. Operation time reflow station
Initial Conditions	1 circuit card tray in each inter-station buffer with the following station busy
Number of Replicates	20
Simulation End Time	2400 minutes (one week)

Since management assesses production weekly, a terminating experiment with a simulation time interval of one week was chosen. The size of each of the two buffers is of interest. Note that the workstations in this study have the same operation time distribution, indicating that the line is balanced as it should be. Analytic and empirical research have shown that for serial systems whose workstations have the same operation time distribution that buffers of equal size are preferred (Askin and Standridge, 1993; Conway et al., 1988).

There was some debate as to whether throughput, WIP, or lead time should be the primary performance measure. Since all circuit card trays that arrive also depart eventually the throughput rate in-bound to the system is the same as the throughput rate out-bound from the system, at least in the long term. Note that by Little's Law, the ratio of the WIP to the lead time (LT) is equal to the (known) throughput rate. Thus, either the total WIP, including that preceding the first workstation, or the lead time could be measured.

[3] The problems at the end of the chapter reference the performance measures not discussed in the text.

The lead time is easy to observe since computing it only requires recording the time of arrivals as a entity attribute. Thus, the lead time can be computed when a entity leaves the system. Computing the WIP requires counting the number of entities within the line. This also is straightforward. We will choose to compute the lead time for this study.

Buffer size is the model parameter of interest. The relationship between buffer size and lead time is needed. Thus, various values of buffer size will be simulated and the lead time observed. Average lead time is one performance measure statistic of interest along with the percent blocked time of each station.

There are four random variables in the model, the time between arrivals and three operation times, with one stream for each. The simulated time interval is the same as the management review period for production, one week. Twenty replicates will be made. Since the utilization of the stations is high, a busy station with one circuit card tray in the preceding inter-station buffer seems like a typical system state. Thus, the initial conditions were determined.

Verification and validation evidence were obtained from a simulation experiment with the inter-station buffer size set to the maximum value in the experiment, 16. Results show almost no blocking in this case. Verification evidence consists of balancing the number of arrivals with the number of departures and the number remaining in the simulation at the end of the run for one replicate. These values are shown in Table 7-2.

Table 7-2: Verification Evidence for Replicate 1

	Circuit Trays Arriving	**Circuit Trays Departing or Remaining at the end of the Simulation**
Initial conditions	6	
Arrival process	794	
Completed processing		785
In buffers		12
In processing		3
Total	800	800

Validation evidence is obtained by comparing the percent busy time of the deposition station as estimated from the simulation results with the expected value computed as follows. The average operation time is (0.7 + 4.7)/2 = 2.7 minutes. The average time between arrivals is 3 minutes. Thus, the expected percent busy time is 2.7 / 3.0 = 90%. The approximate 99% confidence interval for the true percent busy time computed from the simulation results is 86.7 % - 91.7%. Since this interval contains the analytical determined busy time value, validation evidence is obtained.

Since the other stations can be blocked, their percent busy time is not as straightforward to compute analytically. Thus, validation evidence with regard to the deposition and placement stations is more difficult to obtain.

Simulation results for the various values of inter-station buffer capacity will be compared and differences noted. Statistical analysis will be performed to confirm differences that affect what buffer capacity is chosen. Table 7-3 shows the average lead time for each buffer capacity for each replicate as well as summary statistics.

Table 7-3: Average Lead Time Summary (minutes)

	Buffer Capacity				
Replicate	1	2	4	8	16
1	75.4	46.3	36.6	33.9	33.2
2	121.3	65.7	40.9	38.2	38.1
3	97.7	66.8	48.6	39.6	39.4
4	41.5	29.9	25.1	25.0	25.0
5	45.8	27.5	23.6	23.5	23.5
6	93.7	45.8	28.9	27.9	27.9
7	47.1	35.6	29.1	28.2	28.2
8	36.5	24.6	21.8	21.7	21.7
9	45.2	28.5	25.3	25.0	25.0
10	52.1	26.7	23.6	23.1	23.1
11	137.2	87.4	57.6	48.8	48.8
12	102.6	44.9	36.1	34.1	34.1
13	70.6	41.3	28.8	26.9	26.9
14	44.7	33.9	26.9	25.5	25.5
15	97.4	54.6	35.0	30.9	30.4
16	46.9	29.9	27.1	26.4	26.4
17	39.1	31.6	28.5	27.2	27.2
18	95.9	69.7	51.6	43.4	43.4
19	28.7	23.5	21.9	21.9	21.9
20	80.2	44.5	35.0	34.0	34.0
Average	70.0	42.9	32.6	30.3	30.2
Std. Dev.	31.5	17.7	10.2	7.5	7.5
99% CI Lower Bound	49.8	31.6	26.1	25.5	25.4
99% CI Upper Bound	90.2	54.3	39.1	35.1	35.0

Figure 7-3 shows a graph of the average time in the system versus the buffer capacity. From Table 7-3 and Figure 7-3 it is easily seen that the average time in the system decreases significantly when the buffer capacity is increased from 1 to 2 as well as from 2 to 4. Smaller decreases are seen when the buffer capacity is increased further. The statistical significance of these latter differences will be determined.

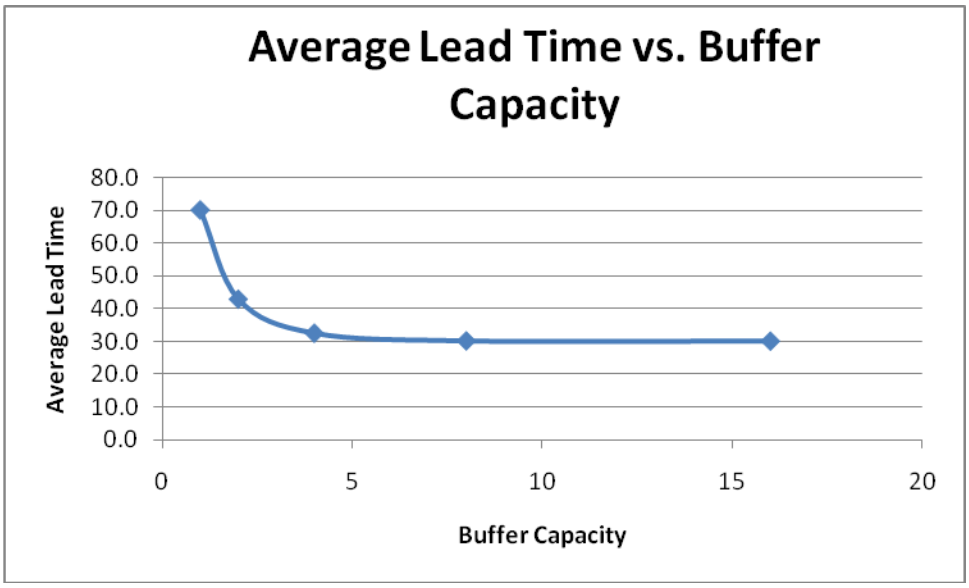


Figure 7-3: Average Time in the System versus Buffer Capacity

The minimum possible average time in the system is the sum of the average processing times at each station 2.7 + 2.7 + 2.7 = 8.1 minutes. Note that for buffer sizes of 4, 8, and 16, the average time in the system is 4 to 5 times the minimum average cycle time. This is due to the high degree of variability in the system: exponential arrival times and uniformly distributed service times with a wide range as well as the high utilization of the work stations. This result is consistent with the VUT equation that shows that the lead time increases as the variability of the time between arrivals, the variability of the service time, and the utilization increase.

The paired-t method is used to compute approximate 99% confidence intervals for the average difference in lead time for selected buffer sizes. These results along with the approximate 99% confidence intervals for the average lead time for each buffer size are shown in Table 7-4. Note that average lead time reduction using 16 buffers instead of 8, is not statistically significant since the approximate 99% confidence interval for the difference in average lead time contains 0.

Table 7-5 summarizes the percent blocked time for the deposition station as well as the differences in percent block time including paired-t confidence intervals for the mean difference for neighboring values of buffer sizes of interest.

The percentage of time that the deposition station is blocked decreases as the buffer size increases as would be expected. The paired-t confidence interval for the difference in percentage of blocked time for 16 buffers versus 8 buffers does not contain 0. Thus, the reduction in percent blocked time due to the larger buffer size is not statistically significant. Further note that the 99% confidence intervals for the percent of time blocked for the case of 8 and 16 buffers both contain 0. Thus, the percent of blocked time for these cases is not significantly different from zero.

Table 7-4: Paired t Test for Average Lead Time (minutes)

	Buffer Capacity				
Replicate	4	8	Diff 4 - 8	16	Diff 8 -16
1	36.6	33.9	2.7	33.2	0.7
2	40.9	38.2	2.7	38.1	0.1
3	48.6	39.6	9.0	39.4	0.3
4	25.1	25.0	0.2	25.0	0.0
5	23.6	23.5	0.1	23.5	0.0
6	28.9	27.9	0.9	27.9	0.0
7	29.1	28.2	0.9	28.2	0.0
8	21.8	21.7	0.1	21.7	0.0
9	25.3	25.0	0.4	25.0	0.0
10	23.6	23.1	0.5	23.1	0.0
11	57.6	48.8	8.9	48.8	0.0
12	36.1	34.1	2.0	34.1	0.0
13	28.8	26.9	1.9	26.9	0.0
14	26.9	25.5	1.4	25.5	0.0
15	35.0	30.9	4.1	30.4	0.5
16	27.1	26.4	0.7	26.4	0.0
17	28.5	27.2	1.3	27.2	0.0
18	51.6	43.4	8.2	43.4	0.0
19	21.9	21.9	0.0	21.9	0.0
20	35.0	34.0	1.0	34.0	0.0
Average	32.6	30.3	2.34	30.2	0.08
Std. Dev.	10.2	7.5	2.92	7.5	0.19
99% CI Lower Bound	26.1	25.5	0.47	25.4	-0.04
99% CI Upper Bound	39.1	35.1	4.21	35.0	0.21

Table 7-5: Paired-t Confidence Intervals for Deposition Percent Blocked Time

	Buffer Capacity				
Replicate	4	8	Diff 4 - 8	16	Diff 8 -16
1	2.51%	1.11%	1.41%	0.00%	1.11%
2	1.57%	0.00%	1.57%	0.00%	0.00%
3	1.12%	0.29%	0.83%	0.00%	0.29%
4	0.84%	0.00%	0.84%	0.00%	0.00%
5	0.53%	0.00%	0.53%	0.00%	0.00%
6	1.48%	0.11%	1.37%	0.00%	0.11%
7	2.19%	0.00%	2.19%	0.00%	0.00%
8	0.69%	0.00%	0.69%	0.00%	0.00%
9	0.66%	0.10%	0.56%	0.00%	0.10%
10	0.49%	0.00%	0.49%	0.00%	0.00%
11	3.62%	1.65%	1.96%	0.20%	1.45%
12	0.90%	0.00%	0.90%	0.00%	0.00%
13	0.61%	0.00%	0.61%	0.00%	0.00%
14	1.33%	0.12%	1.21%	0.00%	0.12%
15	1.46%	0.12%	1.33%	0.00%	0.12%
16	1.30%	0.29%	1.01%	0.00%	0.29%
17	0.69%	0.16%	0.53%	0.00%	0.16%
18	2.74%	1.26%	1.47%	0.17%	1.09%
19	0.66%	0.10%	0.57%	0.00%	0.10%
20	2.35%	0.42%	1.92%	0.00%	0.42%
Average	1.39%	0.29%	1.10%	0.02%	0.27%
Std. Dev.	0.87%	0.48%	0.53%	0.06%	0.43%
99% CI Lower Bound		-0.02%	0.76%	-0.02%	-0.01%
99% CI Upper Bound		0.59%	1.44%	0.06%	0.54%

7.3.4 Review and Extend Previous Work

System experts reviewed the results developed in the previous section. They concluded that a buffer size of four should be used in the system. The small, approximately 10%, decrease in average lead time obtained with a buffer size of 8 did not justify the extra space. The difference was statistically significant in the simulation experiment. The average percent blocked time for the deposition station is about 1.4% percent for a buffer size of 4.

It was noted that for some replicates the average lead time was near an hour. This was of some concern.

7.3.5 Implement the Selected Solution and Evaluate

The system was implemented with 4 buffers. Average lead will be monitored. Causes of long average lead time will be identified and corrected.

7.5 Summary

The model of a serial line is evolved from the model of a single workstation. Analytic results are employed in designing simulation experiments. Simulation results help in selecting the interstation buffer sizes to use in the serial system and thus help ensure that the system is as lean as possible upon implementation.

Problems

1. Based on the simulation experiment results that follow for one replicate of the simulation in section 7.3.1 with a buffer size of 4, verify that the number of entities entering the system are all accounted for at the end of the simulation.

Number of entities created through the arrival process:	807
Number of entities created through initial conditioins:	6
Number of entities completing processing at the solder deposition station:	808
Number of entities completing processing at the component placement station:	804
Number of entities completing processing at the solder reflow station:	799
Number of entities waiting for the deposition station resource at the end of the simulation:	2
State of the deposition station resource at the end of the simulation:	Busy
Number of entities waiting for the placement station resource at the end of the simulation:	5
State of the placement station resource at the end of the simulation:	Busy
Number of entities waiting for the reflow station resource at the end of the simulation:	4
State of the reflow station resource at the end of the simulation:	Busy

2. Based on the simulation results presented in this chapter, provide an argument for using 8 buffers instead of 4. Without simulation would a lean team have been more inclined to use a larger buffer size due to a lack of information?

3. Complete the following table of the percentage blocked times for the placement station for the simulation in section 7.3. What statistically significant results are obtained? How do these compare to the lead time and percent blocked time for the deposition station results?

Percent Blocked Time the Placement Station

	Buffer Size				
Replication	**4**	**8**	**8-4**	**16**	**16-8**
1	2.78	1.32		0.00	
2	1.85	0.19		0.00	
3	1.90	0.00		0.00	
4	2.12	0.52		0.00	
5	1.28	0.00		0.00	
6	1.48	0.09		0.00	
7	1.88	0.40		0.00	
8	1.57	0.14		0.00	
9	1.01	0.00		0.00	
10	2.29	0.00		0.00	
Average					
Std. Dev.					
99% CI Lower Bound					
99% CI Upper Bound					

4. Develop and compute the results from an analytic model of the serial line using the equations in chapter 6. Note that the equation for c_d^2 gives the squared co-efficient of variation of the time between arrivals to the next workstation. Assume no blocking of workstations.

5. Develop and implement in a simulation environment a model of a the system described in the manufacturing case problem below but assuming that the system will have all the space that is needed between stations. This means that modeling the interstation buffers is not necessary. Note that a single workstation is simply a serial line with one station. Verify the model.

6. Develop a model of a fast food restaurant that works as follows. The time between customer arrivals is exponentially distributed with mean 0.5 minutes. If the line is longer than 20, an arriving customer goes somewhere else to eat. A customer places an order and pays the cashier. The time to order and pay is uniformly distributed between 0.25 and 0.45 minutes. Next the customer waits for the food gatherer to prepare the order. This time is exponentially distributed with mean 0.4 minutes. There is space for only two people who have ordered and paid to wait for the food gatherer. Finally, the customer waits for the drink dispensing machine. The time to get a drink is 0.4 minutes. Develop a process model of this situation.

7. Design and perform an experiment to determine if reallocating the 8 total buffer spaces so that more were in front of the reflow station and less were in front of the placement station would decrease part time in the system.

Case Problems

Manufacturing
A new serial system consists of three workstations in the following sequence: mill, deburr, and wash. There are buffers between the mill and the deburr stations and between the deburr and the wash stations. It is assumed that sufficient storage exists preceding the mill station. In addition, the wash station jams frequently and must be fixed. The line will serve two part types. The production requirements change from week to week. The data below reflect a typical week. You have been assigned the task of finding the minimum buffer space that doesn't significantly effect lead time.

Relevant data are as follows with all times in minutes:

Time between arrivals -	Part type 1:	Exponentially distributed with mean 2.0
	Part type 2:	Exponentially distributed with mean 3.0
Time at the mill station -	Part type 1:	0.9
	Part type 2:	1.4
Time at the deburr station -		Uniform (0.9, 1.3) for each part type
Time at wash station -		1.0 for each part type
Time between wash station jams -		Exponentially distributed with mean 30.0
Time to fix a wash station jam -		Exponentially distributed with mean 3.0

Embellishment: In the meeting to review your work, it is suggested that the two buffers may be of different sizes but the total amount of buffer space used should not increase. Perform the appropriate simulation experiment.

Case Problem Issues

1. Discuss how arrivals to the system will be modeled.

2. Discuss how verification evidence will be obtained.

3. Discuss how validation evidence can be obtained.
   a. Compute the expected number of arrivals of each type.
   b. Compute the expected number of arrivals per hour of both types together.
   c. Compute the utilization of each workstation.

4. List the important performance measures.

5. What initial conditions should be used?

6. For the simulation language that you are using, discuss how to implement the initial conditions.

7. List the buffer sizes to consider in the simulation experiment and tell why these sizes were chosen.

8. (Embellishment) Discuss whether it is better to have more buffer space between the mill and the deburr station to avoid blocking at the front of the line or to have buffer space between the deburr and the wash stations to avoid blocking at the deburr station when the wash station jams.

Terminating experiment: Use an end time of 168 hours.

Service System

Consider the design of the drive through service area of a fast food restaurant. There are three stations: place order, pay cashier, and pick up food. There is sufficient space for cars preceding the place order station. Your job is to determine the amount of space between the place order and pay cashier stations as well as the pay cashier and pick up food stations in terms of the number of cars. Serving the maximum number of customers possible during the lunch period, 11:00 A.M. to 1:00 P.M., is management's objective. Thus, minimal customer lead time must be achieved. Based on previous experience, customers are classified into types, depending on the size of their order.

Relevant data are as follows with all times in minutes:

Time between arrivals --	Customer type 1:	Exponentially distributed with mean 1.0
	Customer type 2:	Exponentially distributed with mean 1.5
Time at the order station --	Customer type 1:	Exponentially distributed with mean 0.45
	Customer type 2:	Exponentially distributed with mean 0.70

Time at the pay station - Uniform (0.45, 0.65) for each customer type

Time at pickup station -- 0.5 for each customer type

Embellishment: In the meeting to review your work, it is suggested that the two buffers may be of different sizes but the total amount of buffer space used cannot increase due to physical constraints. Perform the appropriate simulation experiments.

Case Problem Issues

1. Discuss how arrivals to the system will be modeled.

2. Discuss how verification evidence will be obtained.

3. Discuss how validation evidence can be obtained.
   a. Compute the expected number of arrivals of each type.
   b. Compute the expected number of arrivals per hour of both types together.
   c. Compute the utilization of each workstation.

4. List the important performance measures.

5. What initial conditions should be used?

6. For the simulation language that you are using, discuss how to implement the initial conditions.

7. List the buffer sizes to consider in the simulation experiment and tell why these sizes were chosen.

Chapter 8

Job Shops

8.1 Introduction

In this chapter, we consider another possible configuration of workstations called a job shop. A serial system processes one or at most a few types of parts that visit all of the system's workstations in sequence. Thus, serial systems are organized around required production operations. A job shop serves a wide variety of parts or jobs. Each type of job may use a different subset of the workstations and visit those stations in a unique sequence. Taken all together, the movement of jobs of all types between workstations appears to be random. Workstations are designed around common operations. For example, all of the milling machines may be gathered together at one workstation.

Several unique aspects of job shops must be taken into account. When a part using a machine is of a different type than its predecessor, the machine may require a setup task to change its operating parameter values or tooling. Different jobs may require significantly different amounts of processing time at a workstation. The routing of each type of job through the shop must be specified.

8.2 Points Made in the Case Study

This case study illustrates principle 2 of chapter 1. Modeler defined performance measures are extracted from standard simulation results to help assess system behavior. In this case, the performance measure of interest is the service level to customers, the percent of jobs completed on time. The shop has defined on time as a lead time of 8 hours or less.

This case study illustrates how analytic computations, such as those defined in principle 11 of chapter 1, can be used to set simulation experiment parameter values. The average number of busy machines of each type is determined using Little’s law. Cycle time is a function of machine utilization, as seen in the VUT equation. Thus, increasing the number of machines of each type would lower utilization and reduce cycle time. Therefore additional machines may be necessary to achieve an acceptable service level.

The experimental design is sequential. The results of initial simulations are used to set the parameter values for subsequent simulations. Additional machines are added at the bottleneck station identified by initial simulations. Subsequent simulations are run to assess the effect on the service level of the additional machines.

The model adapts to the information that is available about the shop in accordance with principle 2 of chapter 1. Jobs are classified into three types with the arrival rate and distribution known for each type only. Each job within a type will be modeled as having the same route through the shop. Processing times are known only by station, independent of job type. Thus, processing times are modeled as random variables with a large variance.

Simulation results illustrate how relieving one bottleneck in a system can create and expose other bottlenecks in the system. As the number of machines of one type is increased, another type of machine becomes the bottleneck.

The job shop model includes several components. The arrival process for each of the three job types is modeled in the same manor. The operation process for each workstation is modeled in the same way. Routing of jobs is included.

8.3 The Case Study

This case study deals with determining the number of machines needed at each workstation to meet a particular level of demand with a satisfactory service level. The average number of busy machines can be determines using Little's law. However, due to waiting time for busy machines, lead time may exceed management's target. In other words, the service level is too low. Additional machines at a station reduce utilization and thus reduce lead time.

8.3.1 Define the Issues and Solution Objective

A job shop consists of four workstations each having one kind of machine: lathe, planer, shaper, and polisher as shown in Figure 8-1. There is one route through the job shop for each of the three job types. Machines may be either in a busy or idle state.

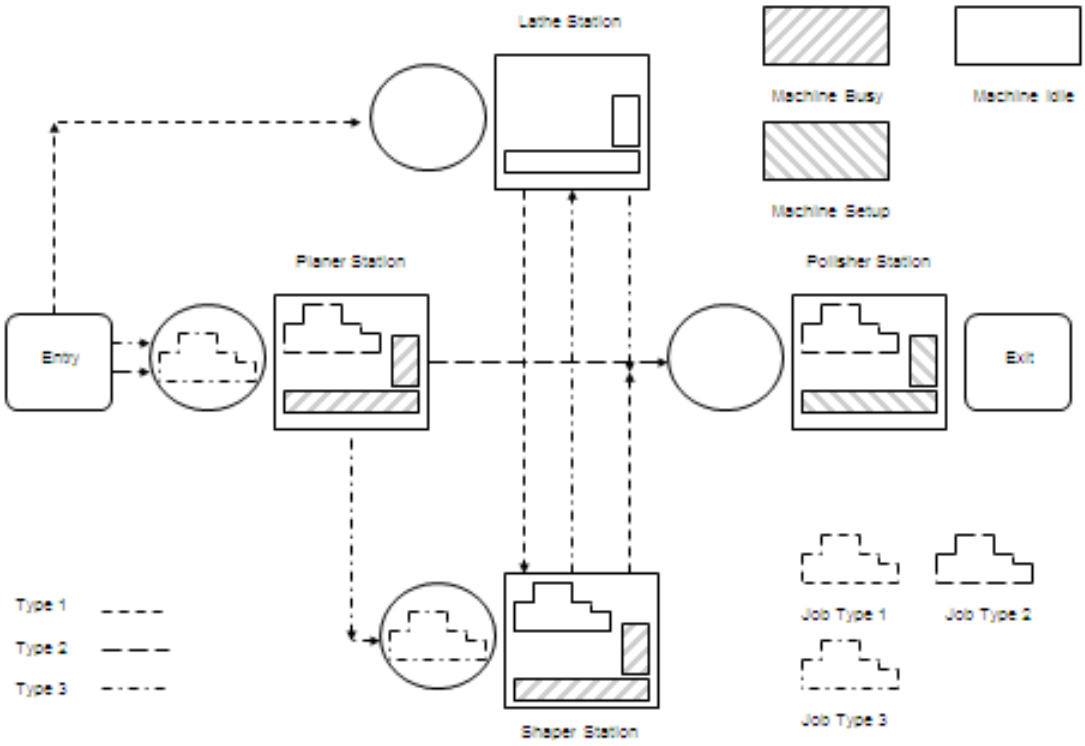


Figure 8-1: Job Shop Layout

Management desires that each job spends less than 8 hours in the shop. The service level is the percent of jobs that meet this target. The objective is to find the minimum number of machines, and thus capital equipment cost, that allows the shop to reach a high, but yet unspecified service level. Management reviews shop performance each month.

The shop processes three types of jobs that have the following routes through the shop:

Type 1: lathe, shaper, polisher
Type 2: planer, polisher
Type 3: planer, shaper, lathe, polisher

Each type of job has its own arrival process, that is its own distribution of time between arrivals. Job processing time data was not available by job type. Thus, a single distribution is used to model operation time at a station. Setup issues can be ignored for now.

Relevant data are as follows. The time between arrivals for each job type is exponentially distributed. The mean time between arrivals for job type 1 is 2.0 hours, for type 2 jobs 2.0 hours, and for type 3 jobs 0.95 hours. Processing times are triangularly distributed with the following parameters (minimum, mode, maximum), given in hours.

Planer:	(1.0,	1.2,	2.4)
Shaper:	(0.75,	1.0,	2.0)
Lathe:	(0.40,	0.80,	1.25)
Polisher:	(0.5,	0.75,	1.5)

8.3.2 Build Models

The model of the job shop uses the following logic:

1. A job arrives as modeled in the arrival process.
2. The job is routed according to the routing process.
   A. If the job needs more operations, it is sent to the station corresponding to its next operation.
   B. If the job has completed all operations, its lead time is computed and the service level for this job recorded. Then the job leaves the shop. Note that the service level is either 100 for acceptable (less than 8 hours) or 0 for not acceptable (greater than 8 hours).
3. The job is processed at the selected workstation as modeled by an operation process.
4. The job returns to step 2.

Job routing corresponds to the routing matrix shown in Figure 8-2, with Depart indicating that the end of the route has been reached.

Job Type	First Operation	Second Operation	Third Operation	Fourth Operation	Last
1	Lathe	Shaper	Polisher	Depart	
2	Planer	Polisher	Depart		
3	Planer	Shaper	Lathe	Polisher	Depart

Figure 8-2. Job Shop Routing Matrix.

Enitities represent jobs and have the following attributes:

JobType =	type of job
ArriveTime =	simulation time of arrival
Location =	location of a job relative to the start of its route: 1st..4th
$OpTime_i$ =	operation time at the ith station on the route of a job: i = 1..4
$Route_i$ =	station at the ith location on the route of a job
ArriveStation =	time of arrival to a station, used in computing the waiting time at a station

There is an arrival process for each job type. The arrival processes for the job types are similar. All of the attributes are assigned including each operation time. The operation time is assigned to assure that the ith arriving job (entity) will be assigned the ith sample from each random number stream for each alternative tested. This is the best implementation of the common random numbers method discussed in chapter 4. The values assigned to the Route attribute are the names of the stations that comprise the route in the order visited. The last station is called Depart to indicate that the entity has completed processing. Whatever performance measure

observations are needed are made in the Depart process. At the end of the arrival process, the entity is sent to the routing process.

The routing process is as follows. The Location relative to the start of the route is incremented by 1. The entity is sent to the process whose name is given by $Route_{Location}$. The routing process requires zero simulation time.

The process as each station is like that of the single workstation discussed in chapter 6. An arriving entity waits for the planer resource. The operation is performed and the resource is made idle. The entity is sent to the routing process.

Upon departure from the shop (Process Depart), the lead time is computed. Thus, the service level can be computed and collected.

The pseudo-English form of the model including the arrival process for type 1 jobs, the operation process for the planer, the routing process and the depart process follows.

Define Arrivals:

Type1

Time of first arrival: 0
Time between arrivals: Exponentially distributed with a mean of 2 hours
Number of arrivals: Infinite

Type2

Time of first arrival: 0
Time between arrivals: Exponentially distributed with a mean of 2 hours
Number of arrivals: Infinite

Type3

Time of first arrival: 0
Time between arrivals: Exponentially distributed with a mean of 0.95 hours
Number of arrivals: Infinite

Define Resources:

Lathe/? with states (Busy, Idle)
Planer/? with states (Busy, Idle)
Polisher/? with states (Busy, Idle)
Shaper/? with states (Busy, Idle)

```
Define Entity Attributes:
        ArrivalTime         // part tagged with its arrival time; each part has its own tag
        JobType             // type of job
        Location            // location of a job relative to the start of its route: 1..4
        OpTime(4)           // operation time at the ith station on the route of a job
        Route(5)            // station at the ith location on the route of a job
        ArriveStation       // time of arrival to a station, used in computing the waiting time

Process ArriveType1
Begin
        Set ArrivalTime = Clock             // record time job arrives on tag
        Set Location    = 0                 // job at start of route
        Set JobType     = 1

        // Set route and processing times
        Set Route(1) to Lathe
        Set OpTime(1) to triangular 0.40, 0.80, 1.25 hr

        Set Route(2) to Shaper
        Set OpTime(2) to triangular 0.75, 1.00, 2.00 hr

        Set Route(3) to Polisher
        Set OpTime(3) to triangular 0.50, 0.75, 1.50 hr

        Set Route(4) to End
        Send to P_Route
End

Process Planer
Begin
        Set ArriveStation = Clock                       // record time job arrives at station
        Wait until Planer/1 is Idle in Queue QPlaner    // job waits for its turn on the machine
        Make Planer/1 Busy                              // job starts on machine; machine is busy
        Tabulate (Clock-ArriveStation) in WaitTimePlaner// keep track of job waiting time

        Wait for OpTime(Location) hours                 // job is processed
        Make Planer/1 Idle                              // job is finished; machine is idle
        Send to P_Route
End

Process Route
Begin
        Location++                                      // Increment location on route
        Send to Route(Location)                         // Send to next station or depart
End

Process Depart
Begin
        //Lead time in hours by job type and for all job types
        if type = Job1 then tabulate (Clock-ArrivalTime) in LeadTime(1)
        if type = Job2 then tabulate (Clock-ArrivalTime) in LeadTime(2)
        if type = Job3 then tabulate (Clock-ArrivalTime) in LeadTime(3)
        tabulate ((Clock-ArrivalTime) in LeadTimeAll
        if ((Clock-ArrivalTime) <= 8 tabulate 100 in Service        // Service level recorded
           else tabulate 0 in Service
End
```

8.3.3 Identify Root Causes and Assess Initial Alternatives

Management reviews the system monthly. Thus, a terminating experiment with an ending time of one month was chosen. Furthermore, management is interested in the percent of jobs that spend less than 8 hours in the shop, the service level, as well as job waiting time at each station. These quantities are the performance measures of interest.

There are seven random number streams, one for the arrival process for each of three types of jobs and one for each of four operation times. Twenty replicates will be made. The initial conditions reflect a typical state of the shop: two jobs of each type at each station.

The model parameters are the number of machines at each station. The expected number of busy machines at each station will be used as the parameter value for the first simulation. Management is able to provide more machines at workstations where the maximum waiting time is excessive in order to meet the service level target.

Table 8-1 summarizes the simulation experiment.

Table 8-1: Simulation Experiment Design for the Job Shop

Element of the Experiment	**Values for This Experiment**
Type of Experiment	Terminating
Model Parameters and Their Values	Number of machines at each station: 1. Average number busy as shown in Table 8-2 following
Performance Measures	1. Percent of jobs whose cycle time is less than 8 hours (Service Level) 2. Waiting time at each station
Random Number Streams	1. Time between arrivals - job type 1 2. Time between arrivals - job type 2 3. Time between arrivals - job type 3 4. Operation time station 1 5. Operation time station 2 6. Operation time station 3 7. Operation time station 4
Initial Conditions	2 parts of each type that can be at a station in the buffer of each station
Number of Replicates	20
Simulation End Time	184 hours (one month)

The expected number of machines needed by each part type at each station is computed as shown in Table 8-2.

1. The mean service time is the mean of the triangular distribution of the service time at each station. This quantity is the arithmetic average of the minimum, mode, and maximum.
2. The expected number of machines is the quotient of the mean operation time divided by the mean time between arrivals (Little's Law).
3. The total expected number of machines at a station is the sum over the three part types. This value is rounded to the next higher whole number to yield the number of machines at each station.
4. Raw processing time is the sum of the mean processing times at each station on the route of a job.

Table 8-2: Expected Number of Machines Needed at Each Workstation

	Planer	Shaper	Lathe	Polisher	Raw Processing Time
Job Type 1					
Mean Time Between Arrivals (TBA = 1/TH)		2	2	2	
Mean Operation Time (CT)		1.25	0.82	0.92	2.99
Expected Number of Machines (= CT / TBA)		0.63	0.41	0.46	
Job Type 2					
Mean Time Between Arrivals (TBA)	2			2	
Mean Operation Time (CT)	1.53			0.92	2.45
Expected Number of Machines (= CT / TBA)	0.77			0.46	
Job Type 3					
Mean Time Between Arrivals (TBA)	0.95	0.95	0.95	0.95	
Mean Operation Time (CT)	1.53	1.25	0.82	0.92	4.52
Expected Number of Machines (= ST / TBA)	1.61	1.32	0.86	0.97	
Total Expected Number of Machines	2.38	1.94	1.27	1.89	
Number of Machines to Use	3	2	2	2	

The mean raw processing time for the job types are 1.45 hours, 2.99 hours, and 4.52 hours. Thus, a cycle time in the shop criteria of one day (8 hours) represents approximate 2 to 5 times the raw processing time which seems reasonable.

Table 8-3 gives the service level for the shop and the maximum waiting time at each station. Notice that the service level is highly variable, ranging from 19.8% to 97.3%. Maximum waiting times are much larger for the shaper than any of the other three machines. The maximum waiting times at the polisher and the planer are also long.

Table 8-3: Simulation Results - Expected Number of Machines Case.

Replicate	Service Level	Maximum Waiting Time at the Lathe (Hours)	Maximum Waiting Time at the Planer (Hours)	Maximum Waiting Time at the Polisher (Hours)	Maximum Waiting Time at the Shaper (Hours)
1	21.2	1.1	4.4	4.6	23.0
2	42.7	1.6	3.3	4.0	10.7
3	22.8	1.2	8.6	9.8	17.1
4	40.1	1.4	5.1	3.4	11.2
5	97.3	1.0	2.6	3.4	2.8
6	73.4	0.8	5.0	4.4	4.4
7	62.4	2.2	2.9	2.8	13.9
8	74.0	1.5	3.9	4.0	9.1
9	24.7	2.2	4.1	4.9	14.0
10	37.6	1.3	4.8	3.3	13.0
11	57.8	1.5	3.6	3.4	7.8
12	19.8	1.1	8.8	9.2	13.3
13	38.0	1.1	7.0	3.7	10.0
14	70.8	1.0	3.1	4.2	9.0
15	36.9	1.1	4.1	6.5	11.5
16	76.2	1.1	3.1	3.5	8.3
17	59.3	1.0	3.0	6.0	7.9
18	60.6	1.5	5.3	4.5	6.3
19	31.1	1.4	3.8	9.1	7.0
20	24.4	1.1	5.3	4.5	16.8
Average	48.6	1.3	4.6	5.0	10.9
Std. Dev.	22.5	0.4	1.8	2.1	4.7
99% CI Lower Bound	34.2	0.1	0.4	0.5	1.1
99% CI Upper Bound	62.9	2.9	2.9	2.9	2.9

8.3.4 Review and Extend Previous Work

System experts reviewed the results developed in the previous section. The average service level of 48.6% was thought to be too low. A service level of at least 95% is needed. A machine will be added to the the shaper station to reduce the maximum waiting time. Additional machines will be added one at a time to the station with the greatest maximum waiting time until the 95% service level is achieved.

8.4 The Case Study with Additional Machines

The need to add one or more machines causes a re-start of the simulation process at the third step, Identify Root Causes and Assess Initial Alternatives.

8.4.1 Identify Root Causes and Assess Initial Alternatives

The simulation experiment uses the same design as in Table 8-1. However, the number of machines at the shaper station is increased by one. Table 8-4 shows the results. The average service level of 56.4% is less than the required 95%. The maximum waiting time at the polisher is much higher than at any of the other stations. Thus, an additional polisher will be added.

Table 8-4: Simulation Results –Additional Shaper Case

Replicate	Service Level	Maximum Waiting Time at the Lathe (Hours)	Maximum Waiting Time at the Planer (Hours)	Maximum Waiting Time at the Polisher (Hours)	Maximum Waiting Time at the Shaper (Hours)
1	17.6	1.9	4.4	10.2	1.5
2	65.6	1.7	3.3	7.3	1.4
3	22.1	1.9	8.6	20.3	1.2
4	56.7	1.1	5.1	7.5	1.3
5	98.0	1.2	2.6	3.8	1.0
6	81.1	1.2	5.0	5.1	1.1
7	72.1	1.7	2.9	6.7	1.9
8	76.7	1.9	3.9	7.2	1.2
9	1.3	1.3	4.1	13.6	1.4
10	39.1	1.2	4.8	8.6	1.8
11	78.6	1.7	3.6	5.9	1.1
12	47.5	1.1	8.8	7.7	1.1
13	61.7	1.3	7.0	7.0	1.9
14	78.4	1.3	3.1	5.9	1.8
15	43.8	1.5	4.1	12.3	2.0
16	83.2	1.6	3.1	6.7	1.6
17	81.7	1.7	3.0	7.0	1.2
18	67.6	2.0	5.3	7.8	1.4
19	43.5	1.4	3.8	10.5	1.6
20	12.1	1.6	5.3	13.8	1.8
Average	56.4	1.5	4.6	8.7	1.5
Std. Dev.	27.1	0.3	1.8	3.8	0.3
99% CI Lower Bound	39.1	1.3	3.5	6.3	1.3
99% CI Upper Bound	73.8	1.7	5.7	11.2	1.7

The simulation results for the case of an additional polisher and shaper are shown in Table 8-5. The average service level, 95.1%, now exceeds 95%. The approximate 99% confidence interval for the service level is 91.1% to 99.2%. Thus with approximately 99% confidence, it can be concluded that the true service level is in a range that includes 95%.

In two of the replicates the service level was less than 80%. In these replicates, the maximum waiting time at the planer exceeded 8 hours. In addition, the average maximum waiting time at the planer was 4.6 hours.

Table 8-5: Simulation Results –Additional Shaper and Polisher Case

Replicate	Service Level	Maximum Waiting Time at the Lathe (Hours)	Maximum Waiting Time at the Planer (Hours)	Maximum Waiting Time at the Polisher (Hours)	Maximum Waiting Time at the Shaper (Hours)
1	95.6	1.9	4.4	2.0	1.5
2	98.9	1.7	3.3	1.5	1.4
3	79.4	1.9	8.6	1.4	1.2
4	96.0	1.1	5.1	1.3	1.3
5	99.0	1.2	2.6	1.7	1.0
6	96.6	1.2	5.0	1.5	1.1
7	99.2	1.7	2.9	1.5	1.9
8	97.5	1.9	3.9	1.4	1.2
9	96.3	1.3	4.1	1.8	1.4
10	94.5	1.2	4.8	1.6	1.8
11	98.7	1.7	3.6	1.3	1.1
12	76.5	1.1	8.8	1.6	1.1
13	92.0	1.3	7.0	1.6	1.9
14	99.2	1.3	3.1	2.0	1.8
15	98.4	1.5	4.1	1.6	2.0
16	99.2	1.6	3.1	1.7	1.6
17	99.4	1.7	3.0	1.4	1.2
18	93.5	2.0	5.3	1.9	1.4
19	99.2	1.4	3.8	1.4	1.6
20	93.9	1.6	5.3	1.3	1.8
Average	95.1	1.5	4.6	1.6	1.5
Std. Dev.	6.3	0.3	1.8	0.2	0.3
99% CI Lower Bound	91.1	1.3	3.5	1.4	1.3
99% CI Upper Bound	99.2	1.7	5.7	1.7	1.7

8.4.2 Review and Extend Previous Work

Management was pleased that the addition of two machines was sufficient to meet the service level requirement. It was decided that the job shop would have the following configuration of machines:

Lathes -- 2
Planers -- 3
Polishers -- 3
Shapers -- 3

In addition, the congestion, in terms of work in process, at the planer station would be monitored. Action would be taken to help avoid excessive waiting time at this station, which now appears to be the bottleneck.

8.4.3 Implement the Selected Solution and Evaluate

The job shop was implemented with the number of machines decided upon at the management review meeting. A monitoring system for work in process at the planer station was put in place.

8.5 Summary

Modeling the flow of multiple job types in a job shop has been presented. Each type of job has a unique route through the workstations.

Simulation experiments are used to set the number of machines at each workstation in order to meet a service level criteria. Little's law is applied to determine the average number of busy machines at each station. These values are used as the initial number of machines at each station in the simulation. After each experiment, the bottleneck station is identified. The next experiment involves increasing the number of machines at this station by one. The series of experiments ends when the service level criteria is met.

Simulation results show how the bottleneck station changes as the result of adding capital equipment.

Problems

1. Based on the simulation experiment results that follow for a job shop similar to the one discussed in this chapter, give verification evidence.

Arrivals:
- Type 1 -- 581
- Type 2 -- 373
- Type 3 -- 482

Number completing operations
- Lathe -- 1063
- Shaper -- 1065
- Polisher -- 1426
- Planer -- 859

Number waiting for operations at the end of the simulation
- Lathe -- 0
- Shaper -- 0
- Polisher -- 5
- Planer -- 0

State of resource at the end of the simulation
- Lathe -- 2 busy
- Shaper -- 6 busy
- Polisher -- 8 busy
- Planer -- 7 busy

Total number of jobs completed --1426

2. Suppose the situation in the job shop is changed as follows. The time between arrivals jobs is 0.25 hours. Two thirds of the jobs are of job type 1 and one-third are of job type 3. Develop an analytic estimate of the required number of each type of machine.

3. Develop a process model of the following small job shop. The shop processes two types of jobs in equal numbers. The time between job arrival is exponentially distributed with a mean of 3 hours. The first type of job visits stations 1 and 2. The second type of job visits stations 2, 1, and 3. Processing times are constant and as follows:

Job Type	First Station Time	Second Station Time	Third Station Time
1	2.0	1.2	
2	0.8	1.7	2.6

4. Model the serial line discussed in the application study of Chapter 7 using the job shop model developed in this chapter.

5. List systems with job shop organizations that you deal with in the course of your everyday life. Develop a single list for the entire class.

6. Provide validation evidence based on the outputs presented in this chapter.

7. Add an additional planer as well as a shaper and a polisher to the job shop in the case study in this chaper. Is the improvement due to the additional shaper worthwhile?

8. The lower limit of the approximate 99% confidence interval for the service level in Table 8-5 is less than the required average service level of 99%. As an alternative, estimate and interpret the approximate 90% confidence interval.

9. Re-run the simulation experiment with the following performance measure added: lead time for entities who lead time exceed the service level target cycle time. Interpret your results.

Case Problem

Management wishes to move the job shop toward a lean system in which there would be three workcells, one for each job type. Due to current budget constraints, no more machines than were found necessary in the case study above can be used, a total of 11. As a first step toward lean in the short term, the following options are to be evaluated with respect to service level and total number of machines.

1. A serial line to produce type 3 jobs and smaller job shop to produce type 1 and 2 jobs with 10 total machines.
2. A serial line to produce type 3 jobs and smaller job shop to produce type 1 and 2 jobs with 11 total machines.
3. Three serial lines, one for each type of job, with 11 total machines.

Build the simulation models and conduct the simulation experiments to evaluate the above options.

Case Problem Issues

1. Suppose each type of job is run on its own dedicated serial line (#3 above). How many machines of each kind are needed for each type of job?

2. Can this analysis be done using the model developed in this chapter? If so, tell how.

3. For item 1 above, how would the 10 total machines be allocated by machine type and location (serial line for jobs of type 3 and job shop for jobs of type 1 and 2)?

4. For item 2 above, how would the 11 total machines be allocated by machine type and location (serial line for jobs of type 3 and job shop for jobs of type 1 and 2)?

Part III
Lean and Beyond Manufacturing

The application studies in part three illustrate sophisticated strategies for operating systems, typically manufacturing systems, to effectively meet customer requirements in a timely fashion while concurrently meeting operations requirements such as keeping inventory levels low and utilization of equipment and workers high. These strategies incorporate both lean techniques as well as beyond lean modeling and analysis.

Before presenting the application studies in chapters 10, 11, and 12, inventory control and organization strategies are presented in chapter 9. These include both traditional and lean strategies.

Chapter 10 deals with flowing the product at the pull of the customer as implemented in the pull approach. How to concurrently model the flow of both products and information is discussed. Establishing inventory levels as a part of controlling pull manufacturing operations is illustrated.

Chapter 11 discusses the cellular manufacturing approach to facility layout. A typical manufacturing cell involving semi-automated machines is studied. The assignment of workers to machines is of interest along with a detailed assessment of the movement of workers within the cell.

Chapter 12 shows how flexible machines could be used together for production. Flexible machines are programmable and thus can perform multiple operations on multiple types of parts. Alternative assignments of operations and part types to machines are compared. The importance of simulating complex, deterministic systems is discussed.

The application studies in this and the remaining parts of the book are more challenging than those in the previous part. They are designed to be metaphors for actual or typical problems that can be addressed using simulation. The applications problems make use of the modeling and experimentation techniques from the corresponding application studies but vary significantly from them. Thus some reflection is required in accomplishing modeling, experimentation, and analysis. Questions associated with application problems provide guidance in accomplishing these activities.

Chapter 9

Inventory Organization and Control

9.1 Introduction

Even before a full conversion to lean manufacturing, a facility can be converted to a pull production strategy. Such a conversion is the subject of chapter 10. An understanding of the nature of inventories is pre-requisite for a conversion to pull. Thus, the organization and control of inventories is the subject of this chapter. Traditional inventory models are presented first. Next the lean idea of the control of inventories using kanbans is described. Finally, a generalization of the kanban approach called constant work in process (CONWIP) is discussed. In addition, a basic simulation model for inventories is shown.

9.2 Traditional Inventory Models

9.2.1 Trading off Number of Setups (Orders) for Inventory

Consider the following situation, commonly called the economic order quantity problem. A product is produced (or purchased) to inventory periodically. Demand for the product is satisfied from inventory and is deterministic and constant in time. How many units of the product should be produced (or purchased) at a time to minimize the annual cost, assuming that all demand must be satisfied on time? This number of units is called the batch size.

The analysis might proceed upon the following lines.

1. What costs are relevant?
    a. The production (or purchase) cost of each unit of the product is sunk, that is the same no matter how many are made at once.
    b. There is a fixed cost per production run (or purchase) no matter how many are made.
    c. There is a cost of holding a unit of product in inventory until it is sold, expressed in $/year. Holding a unit in inventory is analogous to borrowing money. An expense is incurred to produce the product. This expense cannot be repaid until the product is sold. There is an "interest charge" on the expense until it is repaid. This is the same as the holding cost. Thus, the annual holding cost per unit is often calculated as the company minimum attractive rate of return times the cost of one unit of the product.

2. What assumptions are made?
    a. Production is instantaneous. This may or may not be a bad assumption. If product is removed from inventory once per day and the inventory can be replenished by a scheduled production run of length one day every week or two, this assumption is fine. If production runs cannot be precisely scheduled in time due to capacity constraints or competition for production resources with other products or production runs take multiple days, this assumption may make the results obtained from the model questionable.
    b. Upon completion of production, the product can be placed in inventory for immediate delivery to customers.
    c. Each production run incurs the same fixed setup cost, regardless of size or competing activities in the production facility.
    d. There is no competition among products for production resources. If the production facility has sufficient capacity this may be a reasonable assumption. If not, production may not occur exactly at the time needed.

The definitions of all symbols used in the economic order quantity (EOQ) model are given in Table 9-1.

Table 9-1: Definition of Symbols for the Economic Order Quantity Model

Term	Definition
Annual demand rate (D)	Units demanded per year
Unit production cost (c)	Production cost per unit
Fixed cost per batch (A)	Cost of setting up to produce or purchase one batch
Inventory cost per unit per year (h)	h = i * c where i is the corporate interest rate
Batch size (Q)	Optimal value computed using the inventory model
Orders per year (F)	D/Q
Time between orders	1/F = Q/D
Cost per year	Run (order) setup cost + inventory cost = A * F + h * Q/2

The cost components of the model are the annual inventory cost and the annual cost of setting up production runs. The annual inventory cost is the average number of units in inventory times the inventory cost per unit per year. Since demand is constant, inventory declines at a constant rate from its maximum level, the batch size Q, to 0. Thus, the average inventory level is simply Q/2. This idea is shown in Figure 9-1.

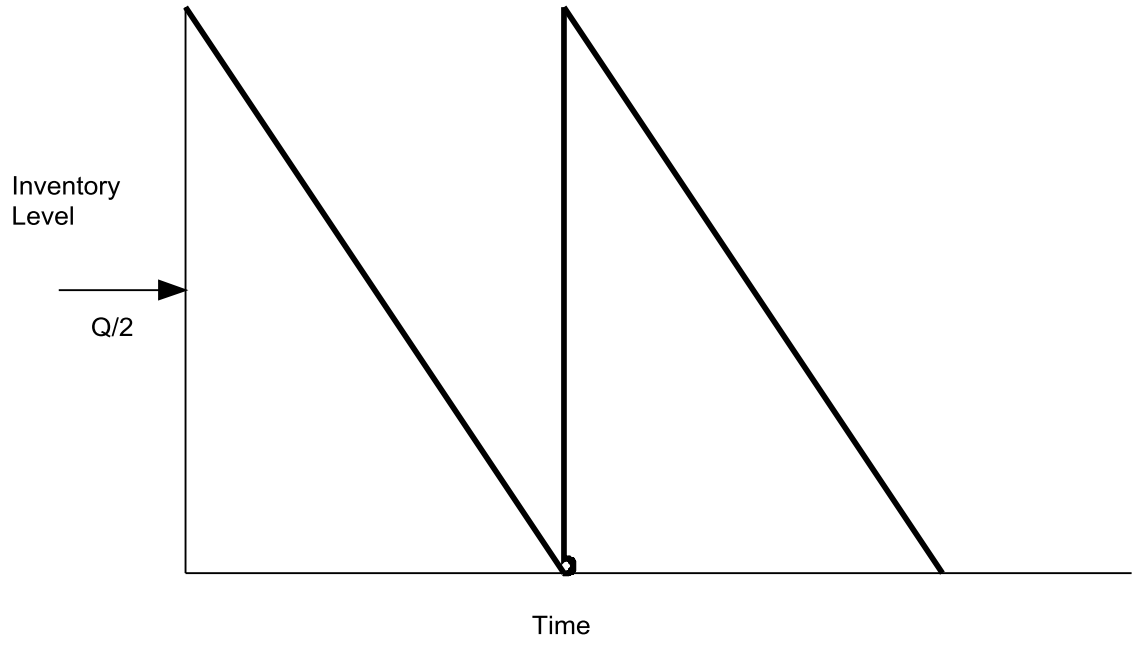


Figure 9-1: Inventory Level for EOQ Model

The number of production runs (orders) per year is the demand divided by the batch size. Thus the total cost per year is given by equation 9-1.

$$Y(Q) = h * \frac{Q}{2} + A * \frac{D}{Q} \qquad (9\text{-}1)$$

Finding the optimal value of Q is accomplished by taking the derivative with respect to Q, setting it equal to 0 and solving for Q. This yields equation 9-2.

$$Q^* = \sqrt{\frac{2 * A * D}{h}} = \sqrt{\frac{A}{h}} * \sqrt{2 * D} \qquad (9\text{-}2)$$

Notice that the optimal batch size Q depends on the square root of the ratio of the fixed cost per batch, A, to the inventory holding cost, h. Thus, the cost of a batch trades off with the inventory holding cost in determining the batch size.

Other quantities of interest are the number of orders per year (F) and the time between orders (T).

$$F^* = D / Q^* \qquad (9\text{-}3)$$

$$T^* = 1 / F^* = Q^* / D \qquad (9\text{-}4)$$

It is important to note that:

Mathematical models help reveal tradeoffs between competing system components or parameters and help resolve them.

Even if values are not available for all model parameters, mathematical models are valuable because they give insight into the nature of tradeoffs. For example in equation 9- 2, as the holding cost increases the batch size decreases and more orders are made per year. This makes sense, since an increase in inventory cost per unit should lead to a smaller average inventory.

As the fixed cost per batch increases, batch size increases and fewer orders are made per year. This makes sense since an increase in the cost fixed cost per batch results in fewer batches.

Suppose cost information is unknown and cannot be determined. What can be done in this application? One approach is to construct a graph of the average inventory level versus the number of production runs (orders) per year. An example graph is shown in Figure 9-2. The optimal tradeoff point is in the “elbow” of the curve. To the right of the elbow, increasing the number of production runs (orders) does little to lower the average inventory. To the left of the elbow, increasing the average inventory does little to reduce the number of production runs (orders).

In Figure 9-2, an average inventory of about 20 to 40 units leads to about 40 to 75 production runs a year. This suggests that optimal batch size can be changed within a reasonably wide range without changing the optimal cost very much. This can be very important as batch sizes may be for practical purposes restricted to a certain set of values, such as multiples of 12, as order placement could be restricted to weekly or monthly.

Example. Perform an inventory versus batch size analysis on the following situation. Demand for medical racks is 4000 racks per year. The production cost of a single rack is $250 with a production run setup cost of $500. The rate of return used by the company is 20%. Production runs can be made once per week, once every two weeks, or once every four weeks.

The optimal batch size (number of units per production run) is given by equation 9-2:

$$Q^* = \sqrt{\frac{2 * A * D}{h}} = \sqrt{\frac{2 * 500 * 4000}{250 * 20\ \%}} = 283$$

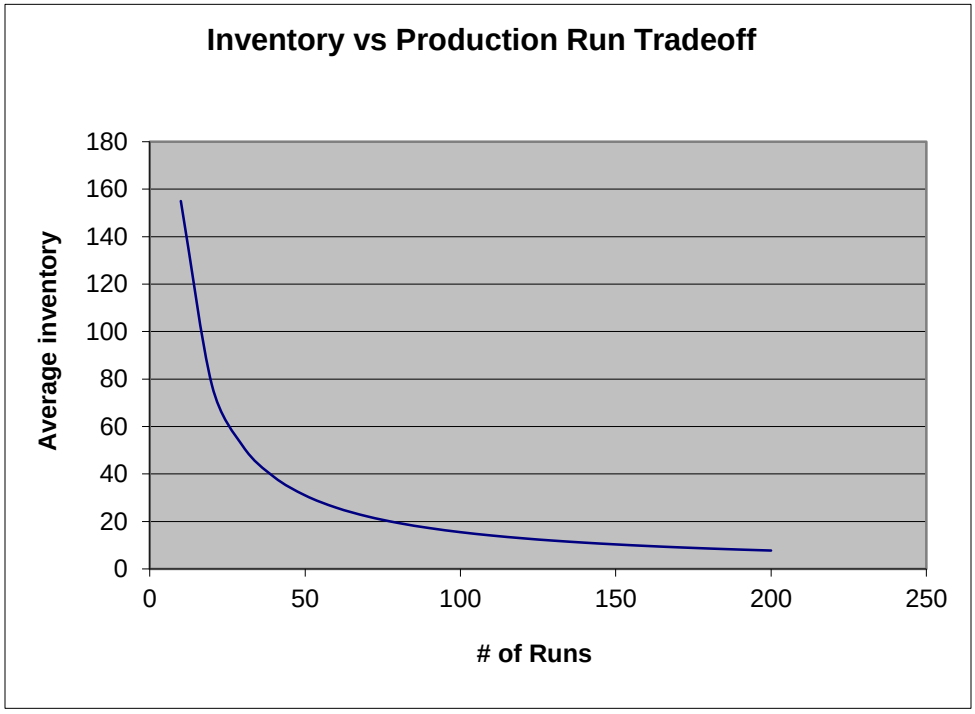


Figure 9- 2: Inventory versus Production Run Tradeoff Graph

The number of production runs per year and the time between production runs is given by equations 9-3 and 4:

$$F^* = D / Q^* = 14.1$$

$$T^* = 1 / F^* = Q^* / D = 3.7 \, weeks$$

The optimal cost is given by equation 9-1:

$$Y\left(Q^*\right) = h * \frac{Q^*}{2} + A * \frac{D}{Q^*} = 250 * 20\,\% * \frac{283}{2} + 500 * 14.1 = 7075 + 7071 = 14146$$

Applying the constraint on the time between production runs yields the following.

$$T' = 4 \, weeks$$

$$F' = 52 \, weeks / 4 \, weeks = 13$$

$$Q' = 4000 / F' = 308$$

$$Y(Q') = h * \frac{Q'}{2} + A * \frac{D}{Q'} = 250 * 20\,\% * \frac{308}{2} + 500 * 13 = 7700 + 6500 = 14200$$

Note that when the optimal value of Q is used the inventory cost and the setup cost of production runs are approximately equal. When the constrained value is used, the inventory cost increases since batch sizes are larger but the setup cost decreases since fewer production runs are made. The total cost is about the same.

9.2.2 Trading Off Customer Service Level for Inventory

Ideally, no inventory would be necessary. Goods would be produced to customer order and delivered to the customer in a timely fashion. However, this is not always possible. Wendy's can cook your hamburger to order but a Christmas tree cannot be grown to the exact size required while the customer waits on the lot. In addition, how many items customers demand and when these demands will occur is not known in advance and is subject variation

Keeping inventory helps satisfy customer demand on-time in light of the conditions described in the preceding paragraph. The **service level** is defined as the percent of the customer demand that is met on time.

Consider the problem of deciding how many Christmas trees to purchase for a Christmas tree lot. Only one order can be placed. The trees may be delivered before the lot opens for business. How many Christmas trees should be ordered if demand is a normally distributed random variable with known mean and standard deviation?

There is a trade-off between:

1. Having unsold trees that are not even good for firewood.
2. Having no trees to sell to a customer who would have bought a tree at a profit for the lot.

Relevant quantities are defined in Table 9-2.

Table 9-2; Definition of Symbols for Service Level – Inventory Trade-off Models

Term	Definition
c_s	Cost of a stock out, for example not having a Christmas tree when a customer wants one.
c_o	Cost of an overage, for example having left over Christmas trees
SL	Service level
Q	Batch size or number of units to order
μ	Mean demand
σ	Standard deviation of demand
z_p	Percent point of the standard normal distribution: $P(Z \leq z_p) = p$. In Excel this is given by NORMSINV(p)

Then it can be shown that the following equation holds:

$$SL = \frac{c_s}{c_s + c_o} = \frac{1}{1 + c_o / c_s} \qquad (9\text{-}5)$$

This equation states that the cost-optimal service level depends on the ratio of the cost of a stock out and the cost of an overage.

In the Christmas tree example, the cost of an overage is the cost of a Christmas tree. The cost of a stock out is the profit made on selling a tree. Suppose the cost of Christmas tree to the lot is \$15 and the tree is sold for \$50 (there's the Christmas spirit for you). This implies that the cost of a stock out is \$50 - \$15 = \$35. The cost-optimal service level is given by equation 9-5.

$$SL = \frac{c_s}{c_s + c_o} = \frac{35}{35 + 15} = 70\ \%$$

If demand is normally distributed, the optimal number of units to order is given by the general equation:

$$Q^* = \mu + \sigma * z_{SL} \quad (9\text{-}6)$$

Thus, the optimal number of Christmas trees to purchase if demand is normally distributed with mean 100 and standard deviation 20 is

$$Q^* = 100 + 20 * z_{0.70} = 100 + 20 * 0.524 = 111$$

There are numerous similar situations to which the same logic can be applied. For example, consider a store that sells a particular popular electronics product. The product is re-supplied via a delivery truck periodically.

In this application, the overage cost is equal to the inventory holding cost that can be computed from the cost of the product and the company interest rate as was done in the EOQ model. The shortage cost could be computed as the unit profit on the sale of the product.

However, the manager of the store feels that if the product is out of stock, the customer may go elsewhere for all their shopping needs and never come back. Thus, a pre-specified service level, usually in the range 90% to 99% is required. What is the implied shortage cost? This is given in general terms by equation 9- 7.

$$c_s = c_o * \frac{SL}{1 - SL} \quad (9\text{-}7)$$

Notice that this is equation is highly non-linear with respect to the service level.

Suppose deliveries are made weekly, the overage cost (inventory holding cost) is \$1/per week, and that a manager specifies the service level to be 90%. What is the implied cost of a stock out? From equation 9-7, this cost is computed as follows:

$$c_s = c_o * \frac{SL}{1 - SL} = \$1 * \frac{90\ \%}{1 - 90\ \%} = \$9$$

Note that if the service level is 99%, the cost of a stock out is \$99.

9.3 Inventory Models for Lean Manufacturing

In a lean manufacturing setting, the service level is most often an operating parameter specified by management. Inventory is kept to co-ordinate production and shipping, to guard against variation in demand, and to guard against variation in production. The latter could be due to variation in supplier shipping times, variation in production times, production downtimes and any other cause that makes the completion of production on time uncertain.

A very important idea is that the target inventory level needed to achieve a specified service level is a function of the variance in the process that adds items to the inventory, production, as well as the process the removes items from the inventory, customer demand. If there is no variation in these processes, then there is no need for inventory. Furthermore, the less the variation, the less inventory is needed. Variation could be random, such as the number of units demanded per day by customers, or structural: product A is produced on Monday and Wednesday and product B is produced on Tuesday and Thursday but there is customer demand for each product each day.

We will confine our discussion to the following situation of interest. Product is shipped to the customer early in the morning from inventory and is replaced by a production run during the day. Note that if the production run completes before the next shipment time, production can be considered to be instantaneous. In other words, as long as the production run is completed before the next shipment, how long before is not relevant.

Suppose demand is constant and production is completely reliable. If demand is 100 units per day, then 100 units reside in the inventory until a shipment is made. Then the inventory is zero. The production run is for 100 units, which are placed in the inventory upon completion. This cycle is completed every day.

The following discussion considers how to establish the target inventory level to meet a pre-established service level when demand is random, when production is unreliable, and when both are true.

9.3.1 Random Demand – Normally Distributed

In lean manufacturing, a buffer inventory is established to protect against random variation in customer demand. Suppose daily demand is normally distributed with a mean of μ units and a standard deviation of σ units. Production capacity is such that the inventory can be reliably replaced each day. Management specifies a service level of SL.

Consider equation 9-8,

$$P(X \leq x) \leq SL \qquad (9\text{-}8)$$

This equation says that the probability that the random variable, X, daily demand, is less than the target inventory, the constant x, must be SL. Solving for the target inventory, x, yields equation 9-9.

$$x = \mu + \sigma * z_{SL} \qquad (9\text{-}9)$$

Exercise.

Customer demand is normally distributed with a mean of 100 units per day and a standard deviation of 10 units. Production is completely reliable and replaces inventory every day. Determine the target inventory for service levels of 90%, 95%, 99% and 99.9%.

Suppose production is reliable but can occur only every other day. The two-day demand follows a normal distribution with a mean of $2 * \mu$ units and a standard deviation of $\sqrt{2} * \sigma$ units. The target inventory level is still SL.

Consider the probability of sufficient inventory on the first of the two days. Since the amount of inventory is sufficient for two days, we will assume that the probability of having enough units in inventory on the first day to meet customer demand is very close to 1.

Thus, the probability of sufficient inventory on the second day need only be enough such that the average of this quantity for the first day and the second day is SL. Thus, the probability of sufficient inventory on the second day is $SL2 = 1 - [(1 - SL) * 2]$.

This means that the target inventory for replenishment every two days is given by equation 9-10.

$$x2 = 2 * \mu + \sqrt{2}\sigma * z_{SL2} \qquad (9\text{-}10)$$

This approach can be generalized to n days between production, so long as n is small, a week or less. This condition will be met in lean production situations.

Exercise.

Customer demand is normally distributed with a mean of 100 units per day and a standard deviation of 10 units. Production is completely reliable and replaces inventory every two days. Determine the target inventory for service levels of 90%, 95%, 99% and 99.9%.

9.3.2 Random Demand – Discrete Distributed

In many lean manufacturing situations, customer demand per day is distributed among a relative small numbers of batches of units. For example, a batch of units might be a pallet or a tote.

This situation can be modeled using a discrete distribution. The general form of a discrete distribution for this situation is:

$$\sum p_i = 1 \qquad (9\text{-}11)$$

where i is the number of batches demanded and p_i is the probability of the customer demand being exactly i batches. The value of i ranges from 1 to n, the maximum customer demand. If n is small enough, then a target inventory of n batches is not unreasonable and the service level would be 1.

Suppose a target inventory of n batches is too large. Then the target inventory, x, is the smallest value of x for which equation 9-12 is true.

$$\sum_{i=1}^{x} p_i \geq SL \qquad (9\text{-}12)$$

Exercise

Daily customer demand is expressed in batches as follows:

(4, 20%), (5, 40%), (6, 30%), (7, 10%).

Production is completely reliable and replaces inventory every day. Determine the target inventory for service levels of 90%, 95%, 99% and 99.9%.

Suppose production is reliable but can occur only every other day. The two-day demand distribution is determined by convolving the one-day demand distribution with itself. Convolving has to do with considering all possible combinations of the demand on day one and the demand on day two. Demand amounts are added and probabilities are multiplied. This is shown in Table 9-3 for the example in the preceding box.

Table 9-4 adds together the probabilities for the same values of the two-day demand (day one + day two demand). For example, the probability that the two day demand is exactly 9 batches is 16%, (8% + 8%)

Table 9-3: Possible Combinations of the Demand on Day One and Day Two

Day One Demand		Day Two Demand		Day One + Day Two Demand	
Demand	Probability	Demand	Probability	Demand	Probability
4	20%	4	20%	8	4%
5	40%	4	20%	9	8%
6	30%	4	20%	10	6%
7	10%	4	20%	11	2%
4	20%	5	40%	9	8%
5	40%	5	40%	10	16%
6	30%	5	40%	11	12%
7	10%	5	40%	12	4%
4	20%	6	30%	10	6%
5	40%	6	30%	11	12%
6	30%	6	30%	12	9%
7	10%	6	30%	13	3%
4	20%	7	10%	11	2%
5	40%	7	10%	12	4%
6	30%	7	10%	13	3%
7	10%	7	10%	14	1%

.

Table 9-4: Two-Day Demand Distribution

Demand	Probability
8	4%
9	16%
10	28%
11	28%
12	17%
13	6%
14	1%

Exercise

Daily customer demand is expressed in batches as follows:

(4, 20%), (5, 40%), (6, 30%), (7, 10%).

Production is completely reliable and replaces inventory every two days. Determine the target inventory for service levels of 90%, 95%, 99% and 99.9%.

9.3.3 Unreliable Production – Discrete Distributed

Suppose production is not reliable. That is the number of days to replace inventory is a discrete random variable. Further suppose that demand is a constant value.

Let q_j be the probability of taking exact j days to replace inventory. Then the number of days, d, of inventory that should be kept is the smallest value of d that makes equation 9-13 true.

$$\sum_{j=1}^{d} q_j \geq SL \qquad (9\text{-}13)$$

Exercise

Daily customer demand is a constant 10 batches.

The number of days to replenish the inventory is distributed as follows:
(1, 75%), (2, 15%), (3, 7%), (4, 3%).

Determine the target inventory for service levels of 90%, 95%, 99% and 99.9%.

9.3.4 Unreliable Production and Random Demand – Both Discrete Distributed

Now consider the application where production is unreliable and demand is random. Both the number of days in which the inventory is re-supplied and the customer demand are discrete random variables. Note that the question of interest is: What is the distribution of the demand in the time taken to replenish the inventory?

Consider the simplest application: Production will take either one or two days to replenish the inventory. Thus, it is appropriate to use the one day demand for setting the inventory level with probability q_1 and it is appropriate to use the two day demand for setting the inventory level with probability q_2. This means that the combined distribution of the demand and the number of days to replenish the inventory must be computed.

This will be illustrated with a numeric example. Suppose customer demand expressed in batches is: (1, 40%), (2, 30%), (3, 20%), (4, 10%). Inventory can be replaced in either one day with probability 60% or two days with probability 40%.

1. Compute the two day demand distribution.

Units	Probability
2	16.00%
3	24.00%
4	25.00%
5	20.00%
6	10.00%
7	4.00%
8	1.00%
	100.00%

2. Compute the one and two day conditional distributions. The condition is that the inventory is replaced in that number of days. The demand distribution is multiplied by the probability that the inventory is replaced in that number of days.

One day Demand			
Units	Probability	Condition	Conditional Probability
1	40%	60%	24.0%
2	30%	60%	18.0%
3	20%	60%	12.0%
4	10%	60%	6.0%
	100%		60.0%

Two Day Demand			
Units	Probability	Condition	Conditional Probability
2	16%	40%	6.4%
3	24%	40%	9.6%
4	25%	40%	10.0%
5	20%	40%	8.0%
6	10%	40%	4.0%
7	4%	40%	1.6%
8	1%	40%	0.4%
	100%		40.0%

3. Combine the two conditional distributions into a single distribution. Add the conditional probabilities for all entries with the same number of units.

Combined Distribution	
Units	Probability
1	24.0%
2	24.4%
3	21.6%
4	16.0%
5	8.0%
6	4.0%
7	1.6%
8	0.4%
	100.0%

Note that for a customer service level of 98%, six units would be kept in inventory.

Exercise

Daily customer demand is expressed in batches as follows:

(4, 20%), (5, 40%), (6, 30%), (7, 10%).

Production is completely not reliable is distributed as follows: (1, 80%), (2, 20%).

Determine the target inventory for service levels of 90%, 95%, 99% and 99.9%.

9.3.5 Production Quantities

Replacing inventory means that the production volume each day is the same random variable as customer demand. Thus, the quantity to produce varies from day to day (or every other day to every other day). This can cause capacity and scheduling issues.

9.3.6 Demand in Fixed Time Period

Suppose the number of units (batches) demanded in fixed period of time, T, is of interest. Suppose the time between demands is exponentially distributed. It follows mathematically that the number of demands in a period of time T is Poisson distributed:

$$p(x) = \frac{e^{-mean} * mean^{x}}{x!};\ x \text{ is a non - negative integer} \qquad (9\text{-}14)$$

where x is the number of units demanded and mean is the average number units demanded in time T. Often the mean must be computed by multiplying two quantities:

1. The average number of units demanded per hour.
2. The number of hours in T.

The Excel function Poisson can be used to compute probabilities using equation 9-14.

Poisson(x, mean, FALSE).

A product has a mean demand of 1.5 units per hour. Suppose production is constant with a takt time of 40 minutes (= 60 minutes / 1.5 units). What is the distribution of the demand in the takt time?

Demand per hour	1.5	
Hours in T	0.666667	
Mean demand in T	1	
X	Probability	Cumulative
0	0.368	0.368
1	0.368	0.736
2	0.184	0.920
3	0.061	0.981
4	0.015	0.996
5	0.003	0.999
6	0.001	1.000

How many units are needed in inventory for a 95% service level the takt time that is such that the probability of running out of inventory before a unit is replaced is 5%?

9.3.7 Simulation Model of an Inventory Situation

Consider a simulation model and experiment to validate the 95% service level in the previous example. Production produces an item to inventory at a constant rate of 1.5 units per hour, one unit every 40 minutes. Since the demand is Poisson distributed it follows that the time between demands is exponentially distributed with a mean equal to the takt time of 40 minutes.

The model is as follows. There is one process for demands that take items from the inventory and one process for adding items back to the inventory.

The initial conditions for any simulation experiment involving inventory must include the initial inventory level which is set to the target inventory value. Determining the target inventory value was discussed in the previous sections in this chapter. Each simulation language has its own requirements for setting the initial value of state variables such inventory levels.

Inventory demand and replenishment model

```
Define Arrivals:
        Demand Process
                Time of first arrival:      0
                Time between arrivals:      Exponentially distributed with a mean of 40 minutes
                                            Number of arrivals:      Infinite

        Replenishment process
                Time of first arrival:      0
                Time between arrivals:      Constant 40 minutes
                                            Number of arrivals:      Infinite

Define Attributes:
        ArrivalTime                         // Time of demand

Define State Variables:
        CurrentInventory=3                  // Number of items inventory with an initial value of three

Demand Process
Begin
        ArrivalTime = Clock
        Wait until CurrentInventory > 0     // Wait for an item in inventory
        CurrentInventory--                  // Remove one item from inventory
        // Record Service Level
        if ArrivalTime = Clock then         tabulate 100 in ServiceLevel
        else                                tabulate    0 in ServiceLevel
End

Replenishment Process
Begin
        CurrentInventory++                  // Add item to inventory
End
```

9.4 Introduction to Pull Inventory Management

The inventor of just-in-time manufacturing, Taiichi Ohno, defined the term *pull* as follows:

> Manufacturers and workplaces can no longer base production on desktop planning alone and then distribute, or push, them onto the market. It has become a matter of course for customers or users, each with a different value system, to stand in the frontline of the marketplace and, so to speak, pull the goods they need, in the amount and at the time they need them.

A supermarket (grocery store) has long been a realization of a pull system. Consider a shelf filled with cans of green beans. As customers purchase cans of green beans, less cans remain on the shelf. The staff of the grocery store restocks the shelf whenever too few cans remain. New cans are taken from boxes of cans in the store room. Whenever the number of boxes of cans in the store room becomes too few, additional boxes are ordered from the supplier of green beans.

Note than in this pull system, shelves are restocked and consequently new cases of green beans are ordered depending on the number of cans on the shelves. The number of cans on the shelves depends on current customer demand for green beans.

The alternative to a pull system, which is no longer commonly used, is a push system. In a push system supermarket, the manager would forecast customer demand for green beans for the next time period, say a month. The forecasted number of green beans would be ordered from the supplier. The allocated shelf space would be stocked with cans of green beans. If actual customer demand was less than the forecasted demand, the manager would need to have a sale to try to sell the excess cans of green beans. If the actual demand was greater than the forecasted demand, the manager would somehow need to acquire more green beans.

This illustration points out one fundamental breakthrough of lean manufacturing: inventory levels, both work-in-process (WIP) and finished goods, are controlled characteristics of how a production system operates instead of a result of how it operates as in a push system.

9.4.1 Kanban Systems: One Implementation of the Pull Philosophy

The most common implementation of the pull philosophy is kanban systems. The Japanese word *kanban* is usually translated into English as *card*. A kanban or card is attached to each part or batch of parts (tote, WIP rack, shelf, etc.). To understand the significance of such cards, consider a single workstation followed by a finished goods inventory and proceeded by a raw materials inventory as shown in Figure 9-3. The following items shown in Figure 9-3 are specific to kanban systems.

1. A move kanban shown as a half-moon shaped card attached to the items in the raw material inventory.
2. A production kanban shown as diamond shaped card attached to the items in the finished goods inventory.
3. Stockpoints: locations where kanbans are stored after removal from an item.

The dynamics of this kanban system are as follows.

1. A customer demand causes an item to be removed from the finished goods inventory. The item is given to the customer and the diamond shaped kanban attached to the item is placed in the stockpoint near the finished goods inventory.
2. Periodically, the diamond shaped kanbans are collected from the stockpoint and moved to the workstation. The workstation must produce exactly one item for each diamond shaped kanban it receives. Thus, the finished goods inventory is replenished. Note only the inventory removed by customers is replaced.
3. In order to produce a finished goods item, the workstation must use a raw material item. The workstation receives a raw material item by taking a half-moon shaped kanban to the raw material inventory.

Note the following characteristics of a kanban system.

1. The amount of inventory in a kanban system is proportional to the number of kanbans in the system.
2. Kanban cards and parts flow in oppose directions. Kanbans flow from right to left and parts flow from left to right.
3. The amount of finished goods inventory required depends on the time the workstation takes to produce a part and customer demand. A lower bound on the finished goods inventory can be set given a customer service level, the expected time for the workstation to produce a part, and the probability distribution used to model customer demand.

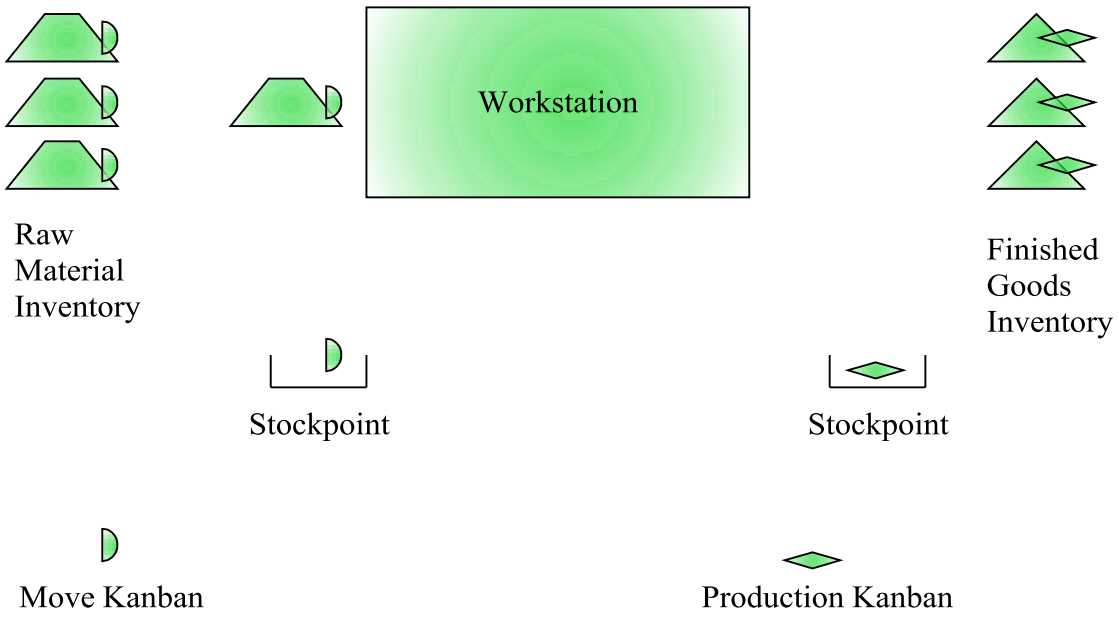


Figure 9-3: Single Workstation Kanban System

Kanban systems can be implemented in a variety of ways. As a second illustration, consider a modified version of the single workstation kanban system. Suppose only one kanban type is used and information is passed electronically. Such a system is shown in Figure 9-4 and operates as follows:

1. A customer demands an item from the finished goods inventory. The kanban is removed from the item and sent to the workstation immediately.
2. The workstation takes the kanban to the raw material inventory to retrieve an item. The kanban is attached to the item.
3. The workstation processes the raw material into the finished good.
4. The item with the kanban attached is taken to the finished goods inventory.

The number of kanbans can be set using standard methods for establishing inventory levels that have been previously discussed. Try the following problems.

1. Demand for finished goods is Poisson distributed at the rate of 10 per hour. Once an item has been removed from finished goods inventory, the system takes on the average 30 minutes to replace it. How much finished goods inventory should be maintained for a 99% service level?

2. Suppose for problem 1, the time in minutes to replace the inventory is distributed as follows: (30, 60%; 40, 30%; 50, 10%). How much inventory should be kept in this application?

3. Suppose for problem 1, all inventory is kept in containers of size 4 parts. There is one kanban per container. How many kanbans are needed for this situation?

Simulation of a kanban system is discussed in the next chapter.

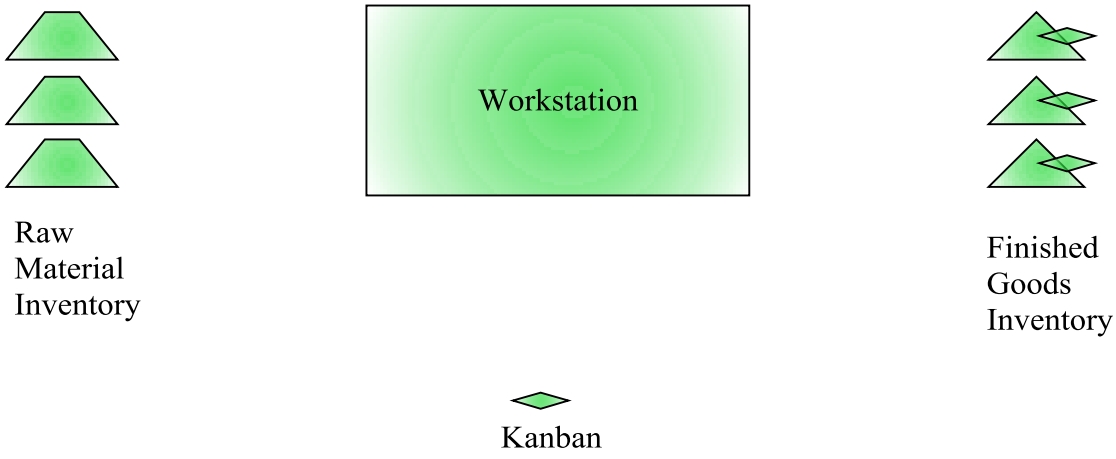


Figure 9-4: Single Workstation Kanban System -- One Kanban Type

9.4.2 CONWIP Systems: A Second Implementation of the Pull Philosophy

One simple way to control the maximum allowable WIP in a production area is to specify its maximum value. This can be accomplished by using a near constant work-in-process system, or CONWIP system. A production area could be a single station, a set of stations, an entire serial line, an entire job shop, or an entire work cell.

Figure 9.5 shows a small CONWIP system with maximum number of jobs in the production area equal to 2. The rectangle encloses the production system that is under the CONWIP control. Two jobs are in processing, one at each workstation. Thus, the third job cannot enter the production system due to the CONWIP control limit of 2 jobs on the production line even though there is space for the job in the buffer of the first workstation. This job should be waiting in an electronic queue of orders as opposed to occupying physical space outside of the CONWIP area.

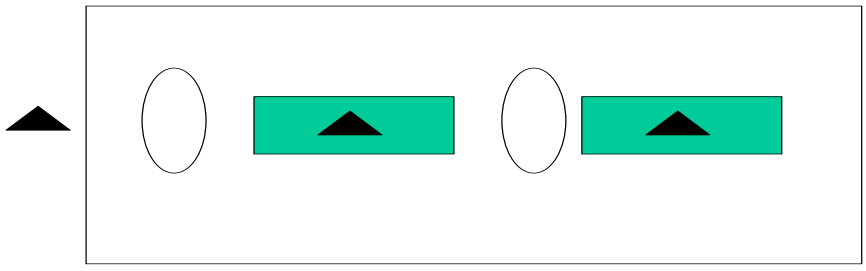


Figure 9-5: CONWIP System Illustration

The following are some important characteristics or traits of a CONWIP system.

1. The CONWIP limit is the only parameter of a CONWIP system.
    a. This parameter must be greater than or equal to the number of workstations in the production area. If not, at least one of the workstations will always be starved.
    b. The ideal CONWIP limit is the smallest value that does not constrain throughput.
    c. In a *multiple* product production area, each job, regardless of type, counts toward the capacity imposed by the *single* CONWIP limit.

2. A CONWIP system controls the maximum WIP in a production area.
    a. The maximum amount of waiting space before any work station is equal to the CONWIP limit or less. It is possible, but unlikely, that all jobs are at the same station at the same time. Thus, buffer sizes before workstations are usually not a constraint on system operation.
    b. If defective parts are detected at the last station on a production line, the CONWIP limit is the upper bound on the number of defective parts produced.
    c. A smaller footprint is needed for WIP storage.

3. Jobs waiting to enter a production area are organized on an electronic or paper list. No parts are waiting.
    a. The list can be re-ordered as needed so that the highest priority jobs are always at the head of the list. For example, if an important customer asks for a rush job it can always be put at the head of the list. The most number of jobs preceding the highest priority job is given by the CONWIP limit.
    b. If the mix of jobs changes, the CONWIP system dynamically adapts to the mix since the system has only one parameter.
    c. Recall Little's Law: WIP = LT * TH. In CONWIP system, WIP is almost constant. Thus, the lead time to produce is easy to predict given a throughput (demand) rate. With the WIP level controlled, the variability in the cycle time is reduced.

4. For a given value of throughput, the average and maximum WIP level in a CONWIP system is less than in a non-CONWIP (push) system.

5. In a CONWIP system, machines with excess capacity will be idle a noticeable amount of the time, which makes some managers very nervous and makes balancing the work between stations more important.

6. Some CONWIP systems arise naturally as result of the material handling devices employed. For example, the amount of WIP may be limited by the number racks or totes available in the production area.

A simulation model of a CONWIP control would include two processes: one for entering the CONWIP area and one for departing the CONWIP area as shown in the following.

CONWIP Processes

```
Define State Variables:
        CONWIPLimit                 // Number of items allowed in CONWIP area
        CONWIPCurrent               // Number of items currently in CONWIP area

EnterCONWIPArea Process
Begin
        Wait until CONWIPCurrent < CONWIPLimit      // Wait for a space in the CONWIP area
        CONWIPCurrent++                             // Add 1 to number in CONWIP area
End

Leave CONWIPArea Process
Begin
        CONWIPCurrent--             // Give back space in CONWIP Area
End
```

Consider the average lead time of jobs in a production area with M workstations. Each workstation has process time t_j. Then the average total processing time is given by summing the average processing times for all work stations yielding the raw processing time, equation 9-15.

$$t_p = \sum_{j=1}^{M} t_j \tag{9-15}$$

Suppose the following:

1. The CONWIP limit is set at $N \geq M$.
2. The production area is balanced, that is the processing time at each station is about the same.
3. Processing times are near constant.

Then the following are true:

1. On the average at each workstation, a job will wait for $\frac{N - M}{M}$ other jobs. M jobs are in processing, one at each station. Thus N-M jobs must be waiting for processing. It is equally likely that a job will be at any station. Thus, the average number of jobs waiting at any station is given by the above quantity.
2. The average waiting time at any particular station is: $\left(\frac{N - M}{M}\right) t_j$. (9-16)
3. The total lead time at each station is:

$$\left(\frac{N - M}{M}\right) t_j + t_j = \left(1 + \frac{N - M}{M}\right) t_j = \left(\frac{N}{M}\right) t_j \tag{9-17}$$

Suppose instead that processing times are random and exponentially distributed. This is for practical purposes the practical worst case processing time since $c_T = 1$.

Then the following are true:

1. On the average at each workstation, a job will wait for $\frac{N-1}{M}$ other jobs. The other N -1 jobs are each equally likely to be at any workstation.

2. The average waiting time at each station is: $\left(\frac{N-1}{M}\right)t_j$. (9-18)

3. The total lead time at each station is: $\left(\frac{N-1}{M}\right)t_j + t_j = \left(1 + \frac{N-1}{M}\right)t_j$ (9-19)

9.4.3 POLCA: An Extension to CONWIP

Suri (2010) proposes the Paired-cell Overlapping Loops of Cards with Authorization (POLCA) approach to control the maximum allowable WIP for jobs processed by any pair of Quick Response Manufacturing (QRM) cells. POLCA can be viewed as an extension of CONWIP and is illustrated in Figure 9.6.

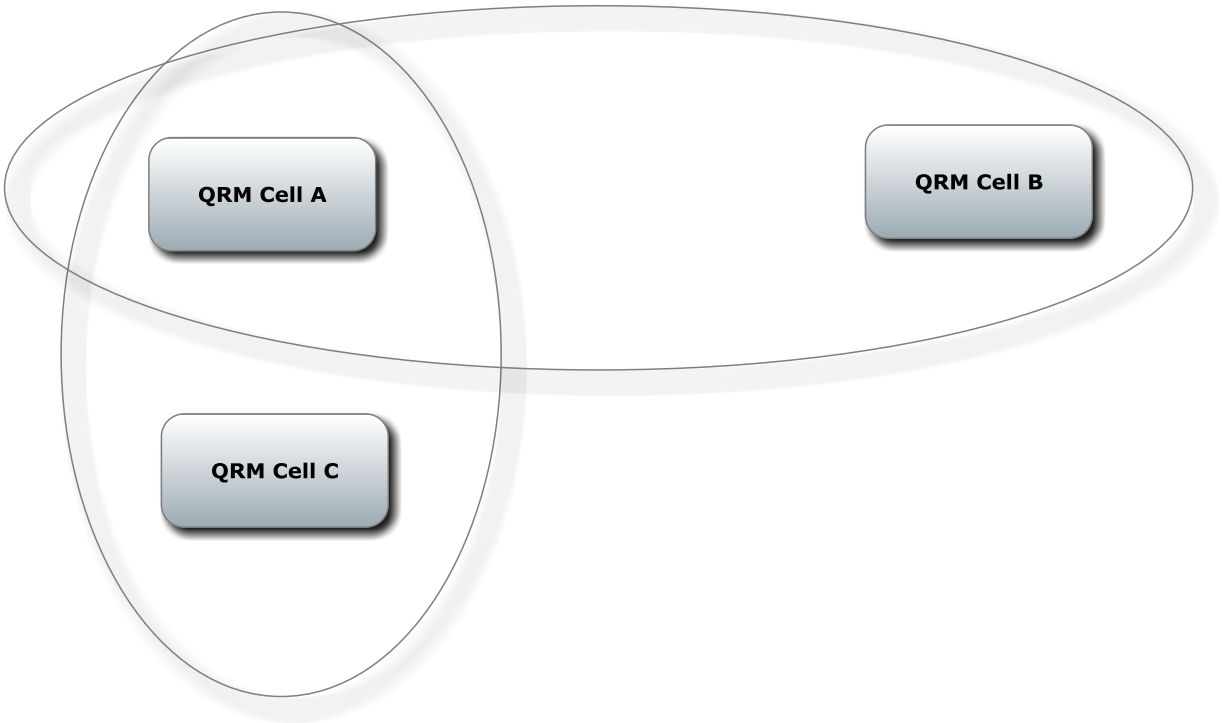


Figure 9-6: POLCA Illustration

In Figure 9-6, there are two types of jobs: 1) those that are processed by QRM Cell A and QRM Cell B (A-B jobs) as well as 2) those that are processed by QRM Cell A and QRM Cell C (A-C jobs). The WIP for each type of job is controlled separately. There is one maximum WIP value for A-B jobs and a second maximum WIP value for A-C jobs. Thus, there are A-B cards in the system and A-C cards in the system.

To start processing a job, two criteria must be met.

1. There is a card available for that job type, i.e. an A-B card for an A-B job, similar to CONWIP.
2. The current date is at or after the projected start date for the job. The start date is computed as the delivery date minus the allowed time to complete the job.

The card is released for reuse when the job is completed in the second of the pair of cells. That is an A-B card must be acquired before the job starts processing in QRM cell A and is released upon completion of processing in QRM cell B.

The time allowed to complete the job could be determined by expert opinion, experience, the VUT equation or simulation.

Suri suggests estimating the number of POLCA cards needed using Little's Law.

WIP = LT * TH

WIP = # of POLCA cards
LT = Lead time in the first QRM cell + Lead time in the second QRM Cell
TH = Demand rate for jobs for example the number of jobs required per week.
For example, if the average lead time in QRM cell A is 30 minutes, the average lead time in QRM cell B is 25 minutes, the demand per day is 30 units, and the working day is 16 hours then the number of A-B POLCA cards needed is as follows:

LT = (30 + 25)/60 = 0.92 hours
TH = 30/16 = 1.875 units per hour
Number of A-B POLCA cards = LT * TH = 2

The following are some important characteristics or traits of a POLCA system.

7. The POLCA limits are the only parameters of a POLCA system.
    a. If each of the QRM cells in a pair has only one POLCA card type, then POLCA is just like CONWIP.
    b. The ideal POLCA limits are the smallest values that do not constrain throughput, which may be greater than the limit estimated using Little's Law.
    c. In a *multiple* product QRM cell pair, each job, regardless of type, counts toward the capacity imposed by the *single* POLCA limit for that pair of cells. For example, there is one limit on the number of A-B POLCA cards regardless of the number of job types flowing from QRM cell A to QRM cell B.

8. A POLCA system controls the maximum WIP in a production area.

9. Jobs waiting to enter a production area are organized on an electronic or paper list. No parts are waiting.
    a. The list can be re-ordered as needed so that the highest priority jobs are always at the head of the list. For example, if an important customer asks for a rush job it can always be put at the head of the list. The most number of jobs preceding the highest priority job is given by the sum of the POLCA limits.
    b. If the mix of jobs changes for any cell pair, the POLCA system dynamically adapts to the mix since there is only one parameter for the cell pair.

10. In a POLCA system, machines with excess capacity will be idle a noticeable amount of the time, which makes some managers very nervous and makes balancing the work between stations more important.

A simulation model of a POLCA control would include two processes: one for entering the first POLCA cell and one for departing the second POLCA cell. Note this is similar to the simulation model for a CONWIP system except there must be one variable for the POLCA limit for each cell pair. In the following example there are two cell pairs: A-B and A-C.

```
POLCA Processes

Define Attributes
        JobType                         // Type of job: either A-B or A-C

Define State Variables:
        POLCALimitAB                    // Number of items allowed in  QRM Cells A-B Processing
        POLCACurrentAB                  // Number of items currently in QRM Cells A-B Processing
        POLCALimitAC                    // Number of items allowed in  QRM Cells A-C Processing
        POLCACurrentAC                  // Number of items currently in QRM Cells A-C Processing

EnterPOLCAPair Process
Begin
        If JobType = AB
        Begin
                Wait until POLCACurrentAB < POLCALimitAB    // Wait for a space in the QRM Cell Pair
                POLCACurrentAB++                            // Add 1 to number in QRM Cell Pair
        End
        If JobType = AC
        Begin
                Wait until POLCACurrentAC < POLCALimitAC    // Wait for a space in the QRM Cell Pair
                POLCACurrentAC++                            // Add 1 to number in QRM Cell Pair
        End
End

Leave CONWIPArea Process
Begin
        If JobType = AB POLCACurrentAB--                    // Give back space in QRM Cell Pair
        If JobType = AC POLCACurrentAC--                    // Give back space in QRM Cell Pair
End
```

Problems

1. If you were assigned problem 5 in chapter 7 then do the following.

    a. Add two inventories to the model one for each part type. Arrivals represent demands for one part from a finished goods inventory. One completion of production a part is added to the inventory.
    b. Add a CONWIP control to the model. The control is around the three workstations.

2. Suppose that demand for a product is forecast to be 1,000 units for the year. Units may be obtained from another plant only on Fridays. Create a graph of the average inventory level (Q/2) versus the number of orders per year to determine the optimal value of Q.

3. Suppose the programs for a Lions home game cost $2.00 to print and sell for $5.00. Program demand is normally distributed with a mean of 30,000 and a standard deviation of 2000.

    a. Based on the shortage cost and the overage cost, how many programs should be printed?
    b. Suppose the service level for program sales is 95%.
        i. How many programs should be printed?
        ii. What is the implied shortage cost?
    c. Construct a graph showing the number of programs printed and the implied shortage cost for service levels from 90% to 99% in increments of 1%.

4. Suppose the Tigers print programs for a series at a time. A three game weekend series with the Yankees is expected to draw 50,000 fans per game. For **each** game, the demand for the programs is normally distributed with a mean of 30,000 and a standard deviation of 3,000. How many programs should be printed for the **weekend** series for a service level of 99%? Note: You must determine the three day demand distribution first.

5. Daily demand in pallets for a particular product made for a particular customer is distributed as follows:

    (5, 75%), (6, 18%), (7, 7%)

    a. How many pallets should be kept in inventory for a 90% service level? For a 95% service level?

    b. Compute the 2-day distribution of demand.

    c. Suppose the inventory can only be re-supplied every 2-days. How many pallets should be kept in inventory for each of the following service levels: 90%, 95%, 99%, and 99.5%?

    d. Suppose the inventory replenishment is unreliable. The replenishment occurs in one day 75% of the time and in 2 days 25% of the time. How many pallets should be kept in inventory for each of the following service levels: 90% and 99%?

6. The inventory for a part is replaced every 4 hours. Demand for the part is at the rate of 0.5 parts per hour. How much inventory should be kept for a 99% service level? Assume that demand is Poisson distributed.

7. Consider a CONWIP system with 3 workstations. The line is nearly balanced with constant processing times as follows (2.9, 3.2, 3.0) minutes.
    a. Derive an equation for the throughput rate given the equation for average part time in the system and Little's Law.
    b. Construct a graph showing the cycle time as a function of the CONWIP limit N.

c. Construct a graph showing the throughput rate as a function of the CONWIP limit.
d. Based on the graphs, select a CONWIP limit.

8. Consider a CONWIP system with 3 workstations. The line is nearly balanced with exponentially distributed processing times with means as follows (2.9, 3.2, 3.0) minutes.
   a. Derive an equation for the throughput rate given the equation for average part time in the system and Little's Law.
   b. Construct a graph showing the cycle time as a function of the CONWIP limit N.
   c. Construct a graph showing the throughput rate as a function of the CONWIP limit.
   d. Based on the graphs, select a CONWIP limit.

9. Consider a Kanban system with a finished goods inventory. Inventory is stored in containers of size 6 items. Customer demand is Poisson distributed with a rate of 10 per hour. Replacement time is uniformly distributed between 2 and 4 hours. Construct a curve showing the number of kanbans required for a 95% service level. (Hint: Consider replacement times of 2 hours, 2.25 hours, 2.50 hours, …, 4 hours).

10. Estimate the number of POLCA cards needed using Little's Law for the following pair of workstations.

    Demand: 100 pieces per 8 hour day, which is constant.

    QRM Cell A with one workstation: Processing time is 4 minutes, exponentially distributed.

    QRM Cell B with one workstation: Processing time is constant, 4 minutes.

Chapter 10

Inventory Control using Kanbans

10.1 Introduction

Inventory contol using kanbans was introduced in Chapter 9. Here, the discussion is extended to include simulation modeling and experimentation of inventory control using kanbans in order to establish target inventory levels. The inventory level is determined by the number of finished goods inventory tags and the number of work-in-process tags allowed. The minimum number of tags needed is proportional to the maximum number of items in the inventories (Mittal and Wang, 1992). Having too few tags causes an interruption in production flow which can lead to unmet demand. Having too many tags causes excess inventory and increases associated costs.

10.2 Points Made in the Case Study

Previously, a process model has represented the physical movement of items through a system. In this application study, the model must represent both the flow of control information that specifies where and when to route items as well as the movement of those items through a system.

Often, models evolve from previously existing models. The model of the job shop with a push system orientation presented in chapter 8 is evolved into a model of the job shop with a pull orientation, including inventory management.

There is a trade-off between maximizing the service level to customers and minimizing the inventory on hand. The service level is the percent of demands that are met on time. A series of simulation experiments can be run varying the number of kanbans, and thus inventory capacity, of each type of product to help quantify this trade-off.

In push systems, inventory level is a consequence of how the system operates and can be a simulation experiment performance measure. In a pull system, the inventory level is a model parameter whose value is to be set through simulation experiments.

A high service level to customers may be achieved even though work-in-process inventory is not always available when a workstation is directed to perform its operation. In other words, a workstation may be starved. Thus, a high service level to each workstation may not be necessary.

Previous information about how a system operates, or should operate, can be combined with simulation results to draw conclusions. This technique is known as the use of **prior information**. In this case, the expected mix of item types demanded is known and is used in conjunction with performance measure estimates to set the number of kanbans. In addition, the number of kanbans for an item type could be the same in all inventories to minimize the number of inventory parameters. This number will be based on the number needed to provide the required customer service level in the finished goods inventory.

10.3 The Case Study

The job shop described in chapter 8 is being converted to a pull inventory control strategy as an intermediate step toward a full lean transformation. The shop consists of four workstations: lathe, planer, shaper, and polisher. The number of machines at each station was determined in chapter 8: 3 planers, 3 shapers, 2 lathes and 3 polishers. The time between demands for each item type is exponentially distributed. The mean time between demands for item type 1 is 2.0

hours, for type 2 items 2.0 hours, and for type 3 items 0.95 hours. The items have the following routes through the shop:

Type 1: lathe, shaper, polisher
Type 2: planer, polisher
Type 3: planer, shaper, lathe, polisher

There is a supermarket following each workstation to hold the items produced by that station. Figure 10-1 shows the job shop configuration plus supermarkets, with job types in each supermarket identified. No routing information is shown. Improvements will be made at each station such that the processing times will be virtually constant and the same regardless of item type.

Planer:	1.533 hours
Shaper:	1.250 hours
Lathe:	0.8167 hours
Polisher:	0.9167 hours

Management anticipates changes in demand each month. The simulation model will be used as a tool to determine the number of kanbans to use in each month.

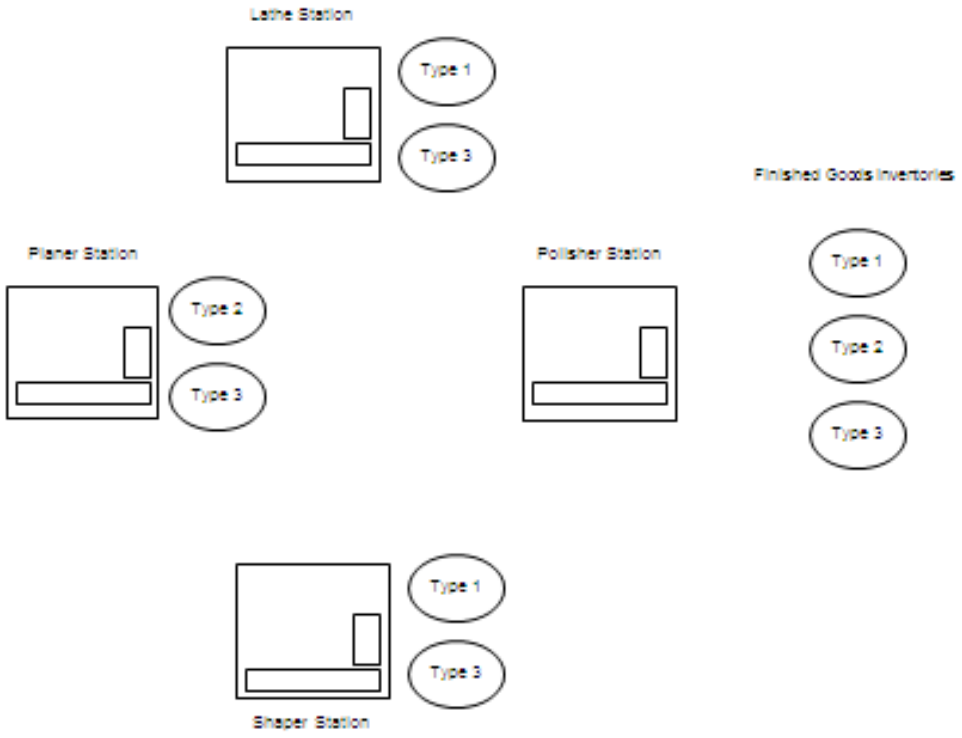


Figure 10-1: Job Shop Configuration

10.3.1 Define the Issues and Solution Objective

Management wishes to achieve a 99% service level provided to customers. The service level is defined as the percent of customer demands that can be statisfied from the finished goods inventories at the time the demand is made. At the same time, management wishes to minimize the amount of finished goods and in process inventory to control costs.

Thus initially, management wishes to determine the minimum number of kanbans for each item type associated with each inventory. The minimum number of kanbans establishes the maximum number of items of each type in each inventory.

10.3.2 Build Models

The process of the flow of control information through the job shop will be the perspective for model building. Note that the flow of control information from workstation to workstation follows the route for processing an item type in reverse order. For example, the route for type 1 items is lathe, shaper, polisher but the flow of control information is polisher, shaper, lathe.

The control information must include the name of the supermarket into which an item is placed upon completion at a workstation with the number of items of each type in the supermarket tracked. In addition, the control information should include the name of the inventory from which an item is taken for processing at a workstation. Supermarket / inventory names are constructed as follows. For the finished goods inventories, the name is INV_FINISHED_ItemType = INV_FINISHED_1 if ItemType = 1 and so forth. For the work in process inventories the name is INV_Station_ItemType, for example INV_SHAPER_1 for type one 1 items that have been completed by the shaper. Thus, the polisher places type 1 items in the inventory INV_FINISHED_1 and removes items from INV_SHAPER_1 since the shaper was the preceding station to the polisher on the production route of a type 1 item. Table 10-1 summarizes the supermarkets / inventories associated with each item type at each workstation. In the model, a distinct state variable models each of the inventories.

Table 10-1: Inventory Names by Item Type and Station

Supermarket / Inventory	Item Type	Output from Station	Input to Station or Customer
INV_FINISHED_1	1	Polisher	Customer
INV_FINISHED_2	2	Polisher	Customer
INV_FINISHED_3	3	Polisher	Customer
INV_SHAPER_1	1	Shaper	Polisher
INV_LATHE_1	1	Lathe	Shaper
INV_PLANER_2	2	Planer	Polisher
INV_LATHE_3	3	Lathe	Polisher
INV_SHAPER_3	3	Shaper	Lathe
INV_PLANER_3	3	Planer	Shaper

The transmission of control information via a kanban is initiated when a finished item is removed from an inventory. The station preceding the FGI, in this case the polisher for all item types, is instructed to complete an item to replace the one removed from the FGI. The polisher station removes a partially completed item from an inventory, for example INV_SHAPER_1 for item type 1, for processing. The item completed by the polisher is placed in the appropriate FGI. The removal of a partially completed item from INV_SHAPER_1 is followed immediately by processing at the shaper to complete a replacement item. This processing at the polisher and shaper can occur concurrently. Information flow and processing continues in this fashion until all inventories for the particular type of item have been replenished.

Each entity has the following attributes:

- ArriveTime: time of arrival of a demand for a job in inventory
- JobType: type of job
- Location: location of the production control information (kanban) relative to the start of the route of a job: 1..4
- $Route_i$: station at the ith location on the route of a job

P_Inv_i: the name of the inventory from which a partially completed item is removed for processing at the ith location on the route of a job

F_Inv_i: the name of the inventory into which the item completed at the workstation is placed at the ith location on the route of a job

The arrival process for type one jobs is shown below. Arrivals represent a demand for a finished item that subsequent triggers the production process to replace the item removed from inventory to satisfy the demand.

The entity attributes are assigned values. Notice that the value of location is set initially to one greater than the ending position on the route. Thus, production is triggered at the last station on the route, which triggers production on the second last station on the route, and so forth.

The arrival process model includes removing an item from a FGI. Thus arriving entites wait for a finished item to be in inventory, remove an item when one is available, and update the number of finished items in the inventory.

Define Arrivals:

Type1

Time of first arrival: 0
Time between arrivals: Exponentially distributed with a mean of 2 hours
Number of arrivals: Infinite

Type2

Time of first arrival: 0
Time between arrivals: Exponentially distributed with a mean of 2 hours
Number of arrivals: Infinite

Type3

Time of first arrival: 0
Time between arrivals: Exponentially distributed with a mean of 0.95 hours
Number of arrivals: Infinite

Define Resources:

Lathe/2 with states (Busy, Idle)
Planer/3 with states (Busy, Idle)
Polisher/3 with states (Busy, Idle)
Shaper/3 with states (Busy, Idle)

Define Entity Attributes:

ArrivalTime // part tagged with its arrival time; each part has its own tag
JobType // type of job
Location // location of a job relative to the start of its route: 1..4
Route(5) // station at the ith location on the route of a job
ArriveStation // time of arrival to a station, used in computing the waiting time
F_Inv(4) // the name of the inventory into which a completed item is placed // at the ith location on the route
P_Inv(4) // the name of the inventory from which an item to be completed // is taken at the ith location on the route

```
Process ArriveType1
Begin
        Set ArrivalTime = Clock          // record time job arrives on tag
        Set JobType     = 1                      // type of job
        Set Location    = 4                      // job at start of route

        // Set route
        Set Route(1) to P_Lathe
        Set Route(2) to P_Shaper
        Set Route(3) to P_Polisher

        // Set following inventories
        Set F_Inv(1) to I1Lathe
        Set F_Inv(2) to I1Shaper
        Set F_Inv(3) to I1Final

        // Set preceding inventories
        Set P_Inv(1) to NULL                     // NULL is a constant indicating no inventory
        Set P_Inv(2) to I1Lathe
        Set P_Inv(3) to I1Shaper

        // Get and update inventory
        Wait until I1Final > 0
        Set I1Final --

        // Record service level
        If (Clock > ArrivalTime) then
        Begin
        // Arrival waited for inventory
                tabulate 0 in ServiceLevel1
                tabulate 0 in ServiceLevelAll
        End
        Else
        Begin
        // Arrival immediately acquired inventory
                tabulate 100 in ServiceLevel1
                tabulate 100 in ServiceLevelAll
        End

        Send to P_Router
End
```

The process at a station includes requesting and receiving items in inventory from preceding stations, processing a item, and placing completed items in inventory at the station. All stations follow this pattern but differ somewhat from each other.

As shown in Figure 10-1, no operations precede the planer station for any item type. Thus information triggering additional production at other stations is unnecessary. Upon completion of the planer operation, a job is added to the inventory whose name is the value of entity attribute F_INV[Location], for example INV_PLANER_1.

The process model of the shaper station is like the process model of the planer station with the retrieval of partially completed items from preceding workstations added. As soon as a partially completed item is retrieved, the routing process is invoked to begin generating the replacement for the item removed from inventory.

The lathe station model is similar to the shaper station model, except that it is the first station on the route for items of type 1. The polisher station model is similar to the shaper station model. Developing the process models of the lathe and polisher stations is left as an exercise for the reader. The shaper and planer station models are shown below.

```
Process Shaper
// Shaper Station
Begin
        // Acquire Preceeding Inventory
        Wait until P_Inv(Location) > 0 in Q_Shaper
        Set P_Inv(Location) --
        Clone to P_Router

        // Process item on Shaper
        Wait until Shaper is Idle in Q_Shaper
        Make Shaper Busy
        Wait for 1.25 hours
        Make Shaper Idle
        Set F_Inv(Location)++
End

Process Planer
// Planer Station
Begin
        // Acquire Preceeding Inventory
        If P_Inv(Location) != NULL) then
        Begin
                Wait until P_Inv(Location) > 0 in Q_Planer
                P_Inv(Location) --
                Clone to P_Router
        End

        // Process item on Planer
        Wait until Planer is Idle in Q_Planer
        Make Planer Busy
        Wait for 0.9167 hours
        Make Planer Idle
        Set F_Inv(Location)++
End
```

The routing process is shown below. The attribute Location is updated by subtracting one from the current value. If the control information has been processed by the first workstation on a route, Location is equal to zero and nothing else needs to be done. Otherwise, the control information is sent to the preceding workstation on the route.

```
Process Router
Begin
        Location - -
        If Location > 0 then send to Route(Location)
End
```

Note the evolution of the model presented in chapter 8 into the model presented in this chapter. The routing process has been modified to send control information through a series of workstations in the reverse order of item movement for processing. This is how the model of the push system orientation was modified to represent a pull system orientation.

Arrivals are interpreted as demands for an item from a finished goods inventory instead of a new item to process. Processing of a new item is triggered via the routing process when a demand is satisfied from the inventory.

Inventory management is added to the model for FGI's for each type of item as well as inventories of partially completed items.

10.3.3 Identify Root Causes and Assess Initial Alternatives

The design of the simulation experiment is summarized in Table 10-2. Management has indicated that demand is expected to change monthly. Thus, a terminating experiment with a time interval of one month is employed. There are three random number streams, one for the arrival process of each item type. Twenty replicates are made. Since stations are busy only in response to a demand, all stations idle is a reasonable state for the system and thus appropriate for initial conditions.

Table 10-2: Simulation Experiment Design for the Just-in-Time Job Shop

Element of the Experiment	Values for This Experiment
Type of Experiment	Terminating
Model Parameters and Their Values	Initial number of items of each type in each inventory 1. Infinite 2. Number needed to provide a 99% service level for the average time to replace an item in the finished goods inventory
Performance Measures	1. Number of items of each type in each buffer 2. Customer service level
Random Number Streams	Three, one for the arrival process of each item type.
Initial Conditions	Idle stations
Number of Replicates	20
Simulation End Time	184 hours (one month)

The experimental strategy is to determine a lower and an upper bound on the inventory level and thus the number of kanbans. First, the minimum inventory level needed for a 100% service level will be determined. This is the maximum inventory level that would ever be used that is the upper bound. The lower bound is the number of items in finished goods inventory needed to achieve a 99% service level for the average time to replace an item taken from the finished goods inventory. The service level in this case will likely be less than 99% since the time to replace some units will be greater than this average.

Prior information is information known before the simulation results are generated that is used along with these results to reach a conclusion. In this case, the following prior information is available.

1. For each item type, the number of kanbans associated with each inventory (finished goods and work in process at each station) should be the same by management policy.
2. All inventories for an item type should be the same as the finished goods inventory level.
3. The finished goods inventory level for a product relative to the other products should be proportional to the arrival rate of the demand for that product relative to the arrival rate of the demand for all products together, at least approximately.

The first piece of prior information makes the kanban control system simpler to operate since the number of kanbans depends only on the product, not the workstation as well. The second point recognizes that the service level depends on the availability of items in a finished goods inventory

when a customer demand occurs. The service level does not depend on the availability of partially completed items in other inventories when requested by a workstation for further processing.

The third point recognizes that the number of kanbans should be balanced between products with respect to customer demand. Table 10-3 show the computations necessary to determine the percent of the customer demand that is for each product. The demand per hour is the reciprocal of the time between demands. The sum of the demand per hour for each item is the total demand per hour. Thus, the percent of the demand for each item is determined as the demand per hour for that item divided by the total.

Table 10-3: Percent of Demand from Each Product

Item	Time Between Demands	Demand per hour	% of Demand
1	2.00	0.50	24%
2	2.00	0.50	24%
3	0.95	1.05	52%
Total	0.49	2.05	100%

The upper bound on the number of kanbans associated with each inventory, equal to the number of items in each inventory in this case, is estimated as follows. The initial number of items in an inventory is set to infinite. In other words, the state variable modeling the inventory is initially set to a very large number. Thus, there will be no waiting for a needed item because it is not in inventory. The inventory level will be observed over time. The minimum inventory level observed in the simulation represents the number of units that were never used and thus are not needed.

Setting the inventory level as discussed in the previous paragraph implies that the service level would be 100% since by design there is always inventory to meet a customer demand.

The simulation results for the infinite inventory case are shown in Table 10-4. These results can be interpreted using the prior information discussed above. In this way, the number of kanbans in each inventory for each item is the same as the finished goods inventory for that item. Thus, the upper bound on the number of items need in each inventory is 4 for item type 1, 4 for item type 2 and 6 for item type 3. Thus, a total of 44 items are needed in inventory.

Note that the percent of the total finished goods inventory for each item is near the percent demand shown in Table 10-3: Type 1, 29% from the simulation versus 24% of the demand; Type 2, 29% versus 24% and Type 3, 44% versus 52%. Further, the percent of the total for type 1 and type 2 are equal to each other as in Table 10-3. Thus, validation evidence is obtained.

Table 10-4: Maximum Inventory Values from the Pull Job Shop Simulation

Replicate	FGI Type 1	Lathe Type 1	Shaper Type 1	FGI Type 2	Planer Type 2	FGI Type 3	Lathe Type 3	Planer Type 3	Shaper Type 3
1	4	3	4	4	4	7	6	5	7
2	4	5	4	4	3	5	5	5	6
3	4	4	5	5	5	9	9	9	10
4	3	4	4	4	4	5	6	5	7
5	4	4	4	4	4	4	4	4	5
6	3	3	4	6	6	5	5	5	6
7	4	4	5	3	3	6	6	5	6
8	5	4	5	4	4	6	6	5	7
9	4	4	4	4	4	5	6	5	6
10	4	4	4	4	4	6	6	6	7
11	3	3	3	4	4	5	5	5	7
12	3	3	4	5	5	12	13	11	14
13	3	3	4	4	4	6	5	5	6
14	3	3	4	4	4	4	5	4	5
15	4	3	4	4	4	7	7	5	7
16	3	3	4	4	4	5	5	5	6
17	3	3	4	4	4	5	5	5	6
18	5	5	6	4	4	5	4	5	6
19	4	4	4	4	4	6	6	6	7
20	3	4	4	4	4	7	6	6	7
Average	3.7	3.7	4.2	4.2	4.1	6.0	6.0	5.6	6.9
Std. Dev.	0.671	0.671	0.616	0.587	0.641	1.835	1.974	1.638	1.971
99% CI Lower Bound	3.2	3.2	3.8	3.8	3.7	4.8	4.7	4.5	5.6
99% CI Upper Bound	4.1	4.1	4.6	4.5	4.5	7.2	7.3	6.6	8.2
Proposed Initial Capacity	4	4	4	4	4	6	6	6	6
% of Total FGI	29%			29%		44%			

Alternatively, the number of items in inventory, and hence the number of kanbans, will be estimated as the number required to provide a 99% service level for the average time needed to replace a product taken from inventory. In other words, for the average time interval from when a product is taken by a customer from finished goods inventory till it is replaced in the finished goods inventory by the production system, the customer service level should be at least 99%. See Askin and Goldberg (2002) for a discussion of this strategy.

The average time interval to replace a product is the sum of two terms: the amount of time waiting for the polisher and the polisher processing time. The former can be determined using the VUT equation and adjusted here for multiple machines (m=3) as shown in equation 10-1.

$$CT_q \approx VUT = \left(\frac{c_a^2 + c_{CT}^2}{2} \right) \left(\frac{\mu^{\sqrt{2(m+1)}-1}}{(1-\mu) * m} \right) CT \qquad (10\text{-}1)$$

The V term will be 0.5. The processing time at the polisher is a constant yielding zero for the coefficient of variation. The time between demands is exponentially distributed and thus has a coefficient of variation of 1. The utilization of the polisher is the processing time (0.9167 hours) divided by the number of machines at the polisher station (3) times the total number of units demanded per hour as shown in Table 10-3. Thus, the utilization of the polisher is 91.3%.

Using these values in equation 10-1 yields 0.1747 hours for the average waiting time before processing at the polisher. The average processing time at the polisher is 0.9167 hours. Thus, the average time for the polisher to complete one item is 1.0913 hours.

The finished goods inventory needed for each product must make the following probability statement true:

P(demand in 1.0913 hours $\leq$ finished good inventory level) $\geq$ 99%.

Since the time between demands for units is exponentially distributed, the number of units demanded in any fixed period of time is poisson distributed with mean equal to the average number of units demand in 1.0913 hours. The mean is computed as 1.0913 hours times the average number of units demand per hour shown in Table 10-3.

Table 10-5 shows the number of items in finished goods inventory needed to achieve at least a 99% service level for 1.0913 hours.

Table 10-5: Finished Goods Inventory Levels for the Average Time to Replace a Unit

Item	Expected Demand in 1.0913 Hours	Inventory Level	Service Level	Percent of Total Inventory
1	0.55	3	99.8%	30%
2	0.55	3	99.8%	30%
3	1.15	4	99.4%	40%
Total		10		100%

Note that the percent of total inventory for each product corresponds reasonably well to that given in Table 10-2, given the small number of units in inventory. Thus, validation evidence is obtained.

The lower bound on the total number of units in inventory is 31 which is 13 units less than the upper bound of 44.

Simulation results shown in Table 10-6 show the service level for each product and overall obtained when the inventory levels shown in Table 10-5 are used.

Table 10-6: Service Level Simulation Results for Finished Goods Inventory Levels (3, 3, 4)

Replicate	Type 1	Type 2	Type 3	Overall
1	98.1	98.8	97.2	97.8
2	97.8	97.6	98.9	98.4
3	96.7	92.6	89.9	92.2
4	100.0	100.0	95.9	97.9
5	98.6	98.8	100.0	99.3
6	100.0	94.1	98.9	97.8
7	96.8	100.0	96.7	97.4
8	97.6	94.4	95.0	95.5
9	98.9	99.0	96.7	97.8
10	97.6	98.9	94.4	96.2
11	100.0	96.9	97.5	97.9
12	100.0	93.2	86.1	91.1
13	100.0	98.8	96.4	97.8
14	100.0	98.8	100.0	99.7
15	98.9	93.9	96.1	96.3
16	100.0	98.9	98.5	99.0
17	100.0	98.7	98.4	98.8
18	96.8	98.9	98.9	98.4
19	98.0	98.9	96.3	97.4
20	100.0	95.7	95.8	96.7
Average	98.8	97.3	96.4	97.2
Std. Dev.	1.26	2.41	3.32	2.17
99% CI Lower Bound	98.0	95.8	94.3	95.8
99% CI Upper Bound	99.6	98.9	98.5	98.6

The service levels are all less than the required 99% through the approximate 99% confidence interval for the service level of type 1 items contains 99%. Note in addition that the range of service levels across the replicates is small.

10.3.4 Review and Extend Previous Work

Management was pleased with the above results. It was thought, however, that the service level obtained when using the upper bound inventory values should be determined by simulation and compared to the service level obtained when using the lower bound values. This was done and the results are shown in Table 10-7.

Table 10-7: Comparison of Service Level for Two Inventory Capacities

	Type 1			Type 2			Type 3			Overall		
Replicate	(3,3,4)	(4,4,6)	difference	(3,3,4)	(4,4,6)	difference	(3,3,4)	(4,4,6)	difference	(3,3,4)	(4,4,6)	difference
1	98.1	100.0	1.9	98.8	100.0	1.2	97.2	99.5	2.4	97.8	99.8	2.0
2	97.8	100.0	2.2	97.6	100.0	2.4	98.9	100.0	1.1	98.4	100.0	1.6
3	96.7	100.0	3.3	92.6	96.7	4.1	89.9	93.8	3.8	92.2	96.0	3.8
4	100.0	100.0	0.0	100.0	100.0	0.0	95.9	100.0	4.1	97.9	100.0	2.1
5	98.6	100.0	1.4	98.8	100.0	1.2	100.0	100.0	0.0	99.3	100.0	0.7
6	100.0	100.0	0.0	94.1	98.0	3.9	98.9	100.0	1.1	97.8	99.5	1.6
7	96.8	100.0	3.2	100.0	100.0	0.0	96.7	100.0	3.3	97.4	100.0	2.6
8	97.6	98.8	1.2	94.4	100.0	5.6	95.0	100.0	5.0	95.5	99.7	4.3
9	98.9	100.0	1.1	99.0	100.0	1.0	96.7	100.0	3.3	97.8	100.0	2.2
10	97.6	100.0	2.4	98.9	100.0	1.1	94.4	100.0	5.6	96.2	100.0	3.8
11	100.0	100.0	0.0	96.9	100.0	3.1	97.5	100.0	2.5	97.9	100.0	2.1
12	100.0	100.0	0.0	93.2	99.0	5.8	86.1	90.6	4.5	91.1	94.9	3.8
13	100.0	100.0	0.0	98.8	100.0	1.3	96.4	100.0	3.6	97.8	100.0	2.2
14	100.0	100.0	0.0	98.8	100.0	1.2	100.0	100.0	0.0	99.7	100.0	0.3
15	98.9	100.0	1.1	93.9	100.0	6.1	96.1	99.5	3.4	96.3	99.7	3.4
16	100.0	100.0	0.0	98.9	100.0	1.1	98.5	100.0	1.5	99.0	100.0	1.0
17	100.0	100.0	0.0	98.7	100.0	1.3	98.4	100.0	1.6	98.8	100.0	1.2
18	96.8	97.9	1.1	98.9	100.0	1.1	98.9	100.0	1.1	98.4	99.5	1.1
19	98.0	100.0	2.0	98.9	100.0	1.1	96.3	100.0	3.7	97.4	100.0	2.6
20	100.0	100.0	0.0	95.7	100.0	4.3	95.8	99.5	3.7	96.7	99.7	3.0
Average	98.8	99.8	1.0	97.3	99.7	2.3	96.4	99.1	2.8	97.2	99.4	2.3
Std. Dev.	1.26	0.53	1.14	2.41	0.85	1.95	3.32	2.45	1.61	2.17	1.39	1.14
99% CI Lower Bound	98.0	99.5	0.3	95.8	99.1	1.1	94.3	97.6	1.7	95.8	98.5	1.5
99% CI Upper Bound	99.6	100.2	1.8	98.9	100.2	3.6	98.5	100.7	3.8	98.6	100.3	3.0

The following can be noted from Table 10-7.

1 For the upper bound inventory values, (4, 4, 6), the approximate 99% service level confidence intervals include 99%.
2. The approximate 99% confidence intervals of the difference in service level do not contain zero. Thus, it can be concluded with 99% confidence that the service level provided by the lower bound on inventory values is less than that provided by the upper bound inventory values.
3. The approximate 99% confidence intervals of the difference in service level are relatively narrow.

Based on these results, management decided that an acceptable service level would be achieved by using a target inventory of 4 units for jobs of type 1 and 2 as well as 6 units for jobs of type 3.

10.3.5 Implement the Selected Solution and Evaluate

The selected inventory levels were implemented and the results monitored.

10.4 Summary

This chapter emphasizes how simulation is used to evaluate the operating strategies for systems. In addition, simulation is helpful in setting the parameters of such operating strategies. The use of simulation in modeling a pull production strategy is shown. The evolution of previously existing models is illustrated.

Problems

1. Develop the process model for the lathe station.

2. Develop the process model for the polisher station.

3. Develop a process model of a single workstation producing one item type that uses a pull production strategy.

4. Find verification evidence for the model discussed in this chapter.

5. Provide additional validation evidence for the model discussed in this chapter.

6. Compare the routing process used in the model in this chapter to that used in chapter 8.

7. Compare the process at each workstation used in the model in this chapter to that in the model in chapter 8.

8. Provide a justification for using different inventory levels at different stations and the FGI for the same product.

9. Find an inventory level between the lower and upper inventory sizes that provides a 99% service level. How much inventory is required?

10. Conduct additional simulation experiments using the model developed in this chapter to determine the product inventory levels that yield a 95% service level.

11. For one customer demand, augment the model to produce a trace of the movement of the entities through the model.

Case Problem -- CONWIP

Convert the assembly line presented in the chapter 7 application problem to a CONWIP production strategy. The assembly line was described as follows.

A new serial system consists of three workstations in the following sequence: mill, deburr, and wash. There are buffers between the mill and the deburr stations and between the deburr and the wash stations. It is assumed that sufficient storage exists preceding the mill station. In addition, the wash station jams frequently and must be fixed. The line will serve two part types. The production requirements change from week to week. The data below reflect a typical week with all times in minutes.

Time between arrivals -	Part type 1:	Exponentially distributed with mean 2.0
	Part type 2:	Exponentially distributed with mean 3.0
Time at the mill station -	Part type 1:	0.9
	Part type 2:	1.4
Time at the deburr station -	Uniform (0.9, 1.3) for each part type	
Time at wash station -	1.0 for each part type	
Time between wash station jams -	Exponentially distributed with mean 30.0	
Time to fix a wash station jam -	Exponentially distributed with mean 3.0	

Arrivals represent demands for completed products. Demands are satisfied from finished goods inventory. Each demand creates a new order for the production of a product of the same type after it is satisfied. The completed product is place in the finished goods inventory.

Three quantities must be determined through simulation experimentation:

1. The CONWIP level, that is the maximum number of parts allowed on the line concurrently.
2. The target FGI level for part type 1.
3. The target FGI level for part type 2.

Two approaches to setting these values could be taken. Choose either one you wish.

1. Approach one.
    a. Set the FGI inventory level for each product as described in this chapter. Set the CONWIP level to infinite (a very high number). Use an infinite (again a very high number) FGI inventory level to determine the minimum number of units needed for a 100% service level.
    b. Determine the inventory level needed for a 99% service level during the average replacement time analytically. The average replacement time is the same for each part type. Determine the average lead time using the VUT equation for each station. Sum the results. Remember that c_a at a following station is equal to c_d at the preceding station. Hints: 1) The VUT equation assumes that there is only one part type processed at a station. Thus, the processing time to use a the mill station is the weighted average processing time for the two part types. The weight is the percent of the total parts processed that each part type is of the total: 60% part type 1 and 40% part type 2. The formulas for the average and the variance for this situation are given in the discussion of discrete distributions in chapter 3. 2) The formula for the variance of a uniform distribution is given in chapter 3. 3) Ignore the downtime at the was station for this analysis.

c. Assess the service level for the inventory level midway between the lower and upper bound.
d. Pick the lowest level of inventory of three that you have tested that yields close to a 99% service level. Note average and maximum WIP on the serial line for this value.
e. Set the CONWIP level to the lowest value that doesn't negatively impact the service level. The minimum feasible CONWIP level is 3. Try values of 3, 4, 5, … until one is found that does not impact the service level. Confirm your choice with a paired-t analysis.
f. Compare the maximum WIP before the CONWIP level was establish to the CONWIP level you selected.

2. Approach two:
   a. Find the minimum CONWIP level that maximizes throughput. Set the two FGI levels to infinite (a very high number) so that the service level is 100%. The minimum feasible CONWIP level is 3. Try values of 3, 4, 5, … until one is found such that the throughput is no longer increasing. Confirm your choice with a paired-t analysis.
   b. Compare the maximum WIP in the serial line without the CONWIP control to the CONWIP level you select. The former could be determined by setting the CONWIP level to a large number.
   c. Estimate the finished goods inventory level need to satisfy customer demands using the approach described in this chapter and after the CONWIP level has been established. Use an infinite (again a very high number) FGI inventory level to determine the minimum number of units needed for a 100% service level.
   d. Determine the inventory level needed for a 99% service level during the average replacement time analytically. The average replacement time is the same for each part type: the average lead time at station j is given by the following equation discussed in Chapter 9 where M = 3 stations and N is the CONWIP level you selected: $\left(\frac{N-1}{M}\right)CT_j + CT_j$
   e. Assess the service level for the inventory level midway between the lower and upper bound and pick the lowest level that yields close to a 99% service level.

Terminating Experiment: Use a simulation time interval of 184 hours.

Application Problem Issues

1. How should the CONWIP control be modeled?

2. What should the ratio of the two FGI levels be if prior information is used?

3. Should the mean or maximum WIP level on the serial line with no CONWIP control be compared to the CONWIP level?

4. Given the CONWIP control, is it necessary to model the finite buffer space between the stations on the serial line? Why or why not?

5. How will verification and validation evidence be obtained?

Case Problem -- POLCA

Convert the assembly line presented in the chapter 7 application problem to a set of QRM Cell pairs as follows.

A QRM analysis determined that there will be three QRM cells processing two part types.

A. Mill and deburr serving both part type 1 and part type 2
B. Wash station 1 serving part type 1
C. Wash station 2 serving part type 2

The wash stations jam frequently and must be fixed.

Time between wash station jams -	Exponentially distributed with mean 30.0
Time to fix a wash station jam -	Exponentially distributed with mean 3.0

Production requirements change from week to week. The data below reflect a typical week with all times in minutes.

Time between arrivals -	Part type 1:	Exponentially distributed with mean 2.0
	Part type 2:	Exponentially distributed with mean 3.0
Time at the mill station -	Part type 1:	0.9
	Part type 2:	1.4
Time at the deburr station -	Uniform (0.9, 1.3) for each part type	
Time at wash station 1 for part type 1-	1.7	
Time at wash station 2 for part type 2 -	2.5	

Arrivals represent demands for completed products. Demands are satisfied from finished goods inventory. Each demand creates a new order for the production of a product of the same type after it is satisfied. The completed product is place in the finished goods inventory.

Three quantities must be determined through simulation experimentation:

4. The number of POLCA cards of each type (A-B and B-C).
5. The target FGI level for part type 1.
6. The target FGI level for part type 2.

Approach

3. Determine an upper bound on the needed inventory for each part type as follows. Set the POLCA cards levels to infinite (a very high number). Use an infinite (again a very high number) FGI inventory level to determine the minimum number of units needed for a 100% service level.
4. Determine a lower bound on the needed inventory for each part type as follows. Determine the inventory level needed for a 99% service level during the average replacement time analytically. Determine the average replacement time separately for each part type. Remember however that the average time in QRM Cell A will be the same for each part type when determine using the VUT equation as described in items a, b, and, c. Remember that c_a at a following station is equal to c_d at the preceding station.
    a. The VUT equation assumes that there is only one part type processed at a station. Thus, the processing time to use at the mill station is the weighted average processing time for the two part types. The weight is the percent of the total parts processed that each part type is of the total: 60% part type 1

and 40% part type 2. The formulas for the average and the variance for this situation are given in the discussion of discrete distributions in chapter 3.

b. The formula for the variance of a uniform distribution is given in chapter 3.
c. Ignore the downtime at the wash stations for this analysis.

5. Assess the service level for the inventory level midway between the lower and upper bounds.
6. Pick the lowest level of inventory of three that you have tested that yields close to a 99% service level.
7. Set the POLCA levels to the lowest values that don't negatively impact the service level. Confirm your choice with a paired-t analysis.

Terminating Experiment: Use a simulation time interval of 184 hours.

Application Problem Issues

1. How should the POLCA control be modeled?

2. What should the ratio of the two FGI levels be if prior information is used?

3. What should the ratio of the number of A-B POLCA cards to the number of A-C POLCA cards be if prior information is used?

4. Given the POLCA control, is it necessary to model the finite buffer space between the stations on the serial line? Why or why not?

5. How will verification and validation evidence be obtained?

Chapter 11

Cellular Manufacturing

11.1 Introduction

Cellular manufacturing systems provide an alternative organization to serial lines and job shops for producing products. Manufacturing is reorganized into cells that provide the operational benefits of a serial line. Each cell performs all of the processing needed on one, or at most a few, similar part types. Each part is processed in the same sequence through a series of machines or manual operations. Cellular manufacturing is effective whenever production volumes are sufficiently high to justify dedicated equipment and only a short series of production steps are required.

For example, the job shop discussed in chapters 8 and 10 could be reorganized into 3 cells, one for each of the three part types. Each cell would behave like the serial line discussed in chapter 7. Each cell would have a sufficient number of each kind of machine to process the type of part for which it was responsible.

Notice that the cellular manufacturing approach eliminates, or at least reduces, the need for setups, since only one part or a small number of similar parts are processed in a particular cell.

Cellular manufacturing seeks to minimize the movement distance of parts within a cell. In addition, each worker in a cell may support multiple operations at several workstations. Thus, each workstation and its machines must be placed in close proximity with all other workstations. A straight line layout results in the first workstation being too distant from the last workstation. Alternative, a U-shaped layout is often used to meet the workstations close together requirement. This is illustrated in Figure 11-1.

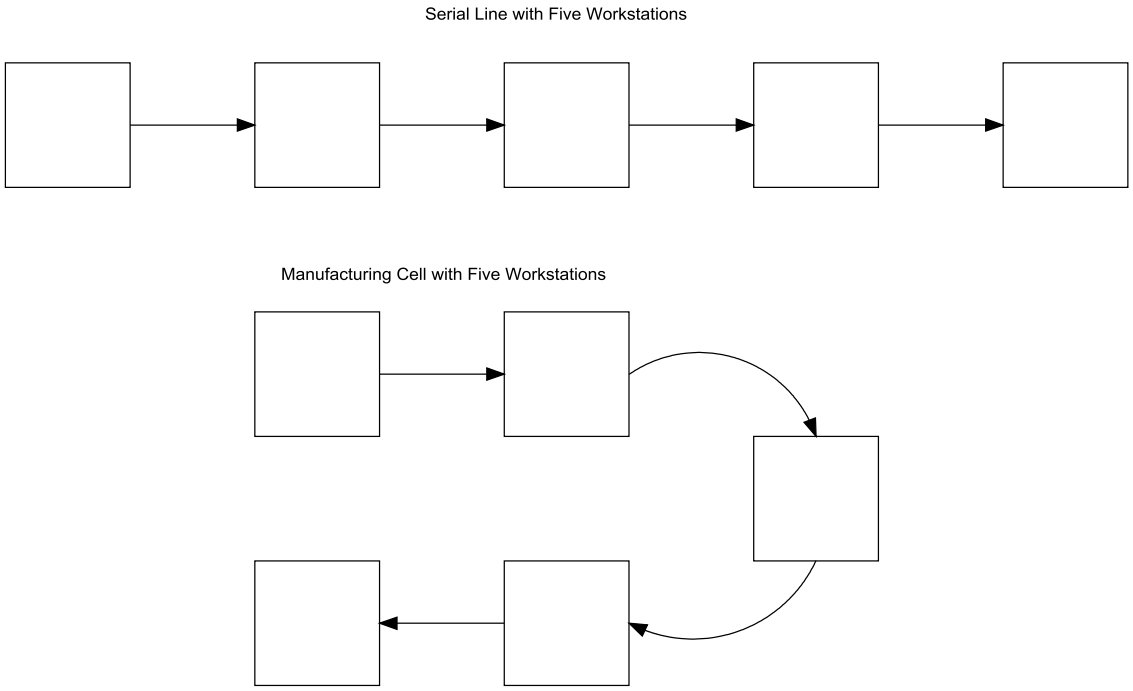


Figure 11-1: Comparison between Serial Line and Manufacturing Cell Layouts

Some companies, such as Innotech, organize all production into cells. Each cell is operated by its workers as an independent business. The businesses share some common resources such as shipping docks and material handling equipment. Rubich and Watson (1998) give some advantages of this approach.

- Improved communication and teamwork – operators are close enough to talk and help each other if necessary.
- An understanding of the entire manufacturing process from raw material to finished product
- An opportunity to meet and discuss issues with customers if any customer concerns develop
- An environment where cell operators have a greater sense of control in how their business (cell) is run
- Responsibility and ownership for producing high quality products on time
- Higher job satisfaction through increased job responsibility and variety

Another goal of cellular manufacturing is to minimize the work-in-process inventory. This is accomplished using the principle of one piece flow that seeks to move individual parts through a work cell as quickly as possible. A worker seeks to keep one piece or part moving through the entire cell. This is the opposite approach to processing multiple parts (a batch) at one workstation and then moving the entire batch to the next workstation for processing. In other words, one piece flow uses a batch size of one.

One piece flow can be used to minimize WIP which results in shorter lead time according to Little's Law. Required manufacturing space is reduced through better layouts and WIP reduction, which also simplifies material handling.

One piece flow works as shown for two stations in Figure 11-2. Initially, the diamond part has completed processing at WS1 and the circle part is waiting in the buffer. The worker arrives to WS1. First the worker removes the diamond part from the machine. Next the worker initiates the circle part on the machine. Finally, the worker moves the diamond part to WS2 and then initiates that part on the machine.

Cellular manufacturing employs a pull strategy. The number of parts the cell must produce per day (or week or month) is established based on the known or forecasted demand for parts. The number of available work hours is set. Then the takt time is computed using equation 11-1.

$$\text{takt time} = \frac{\text{available work hours per day}}{\text{demand per day}} \tag{11-1}$$

The takt time is the maximum time between parts completed by the work cell if demand is to be met. It is also in a sense the mimimum time. If the cell completes parts faster than the takt time, too many parts are made.

The time between completion of parts at the slowest workstation in the cell should not exceed the takt time. This means that the operation time at that workstation should be less than the takt time. Furthermore, one piece flow is assisted greatly by making the operation time at each workstation as close to the same as possible.

This application study presents the use of simulation in determining the staffing for a manufacturing cell. Alternative assignments of workers to machines in the cell are considered. The effect of variation on cell operations is assessed. The use of simulation to evaluate and enhance the original cell staffing plan developed using standard cellular manufacturing computations is shown.

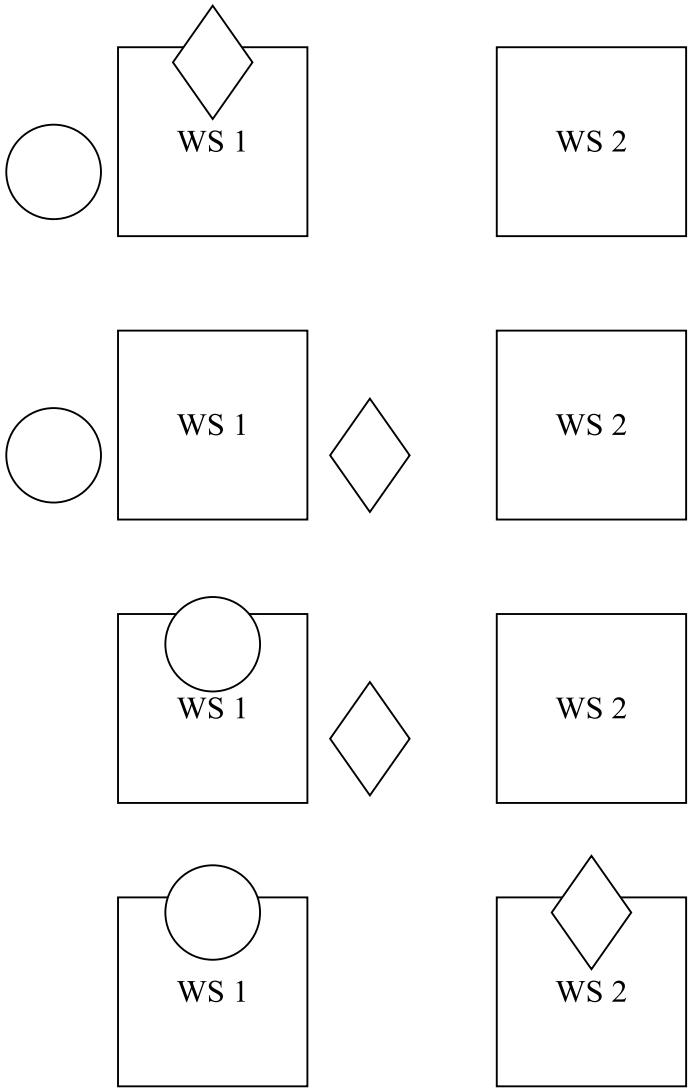


Figure 11-2: One Piece Flow Sequence
for Two Parts and Two Stations

11.2 Points Made in the Case Study

Some benefits of using simulation to enhance lean manufacturing techniques are illustrated. The effect of random variation in raw material (part) arrival times as well as worker walking times is taken into account. Performance of the cell, in terms of the maximum work-in-process and throughput, is predicted. The simulation model and experiment are used to validate the cell design generated by traditional cellular manufacturing computations. Operating rules to co-ordinate part arrivals and the movement of the cell operators are tested.

The movement of both a part through the operations of the cell and a worker from workstation to workstation within a cell must be included in the model. The movement of workers is used as the perspective for model building. The number of parts in each area of each cell is counted. Part movement results in changes in the counts. The average part cycle time in the cell can be estimated from the average number of parts in the cell using Little's Law. This approach is based on a technique developed by Hyden, Roeder and Schruben (2001).

Random variation in worker walking time may be significant since waiting while a worker walks between machines may cause a delay in the start of an operation. Such delays could effectively reduce the capacity of the cell. Thus, the cell could fail to meet its throughput target.

The work done at a workstation must be modeled as multiple tasks: initiating an operation, the operation itself, and removing the part from the machine after the operation is completed. Different resource combinations, machines and workers, are required for each task. Thus, the joint allocation of machine and worker resources must be accomplished.

More than one worker is required to staff the cell. The effectiveness of alternate assigments of workers to workstations can be determined using simulation experiments.

A trace of worker activities can be generated to aid in model validation. The trace is used to provide evidence that the worker moves through the cell as was intended.

11.3 The Case Study[1]

This application study has to do with validating the design of a new manufacturing cell particularly with respect to staffing requirements as well as work in process inventory levels and throughput. An initial value for the number of workers required as well as an initial assignment of workers to workstations can be determined by standard, straightforward cellular manufacturing computations.

A simulation study is required to validate that the number of workers and their assignment to workstations determined by the cellular manufacturing analysis will allow the cell to meet its throughput requirements. The effect on throughput as well as WIP due to other assignments and numbers of workers can be evaluated.

Factors not included in the initial calculations can be taken into account in the simulation model. Task and walking times as well as the time between arrivals of parts may be random variables. Cell operating rules for co-ordination between the activities of multiple workers as well as part arrivals to the cell is necessary.

11.3.1 Define the Issues and Solution Objective

A new manufacturing cell is being implemented. The design of the cell is shown in Figure 11-3. The cell consists of seven workstations each with one machine as well as a raw material inventory of parts to process. A completed finished goods inventory is included.

The work area at each work station is shown by a heavy dot. The worker walking path in the cell is shown by a line. Note that a worker may walk directly between workstation M2 and workstation M6. The worker who is responsible for a machine also walks the part from the immediately preceding workstation or inventory. The worker responsible for workstation M7 also walks a completed part to the finished goods inventory.

Table 11-1 provides basic data concerning the operation at each workstation. All times are in seconds. Manual times represent the constant standard times.

---

[1] Professor Jon Marvel defined this application problem as well as providing other invaluable assistance. Mr. Joel Oostdyk implemented a prototype model. Ms. Michelle Vette provided some excellent insight for improving the application problem.

Table 11-1: Workstation Processing Information (Times in Seconds)

Workstation / Task Name	Workstation / Task ID	Initiation Time (Manual)	Operation Time (Automated)	Removal Time (Manual)	Total Time
Pick Up Raw Material	RM	4			4
Turn Outer Diameter	M1	4	23	3	30
Bore Inner Diameter	M2	5	41	4	50
Face Ends	M3	4	32	4	40
Grind Outer Diameter	M4	3	29	3	35
Grind Outer Diameter	M5	3	29	3	35
Inspect	M6	14			14
Drill	M7	3	24	3	30
Place in Finished Goods Inv.	FG	5			5
Total		45	168	20	233

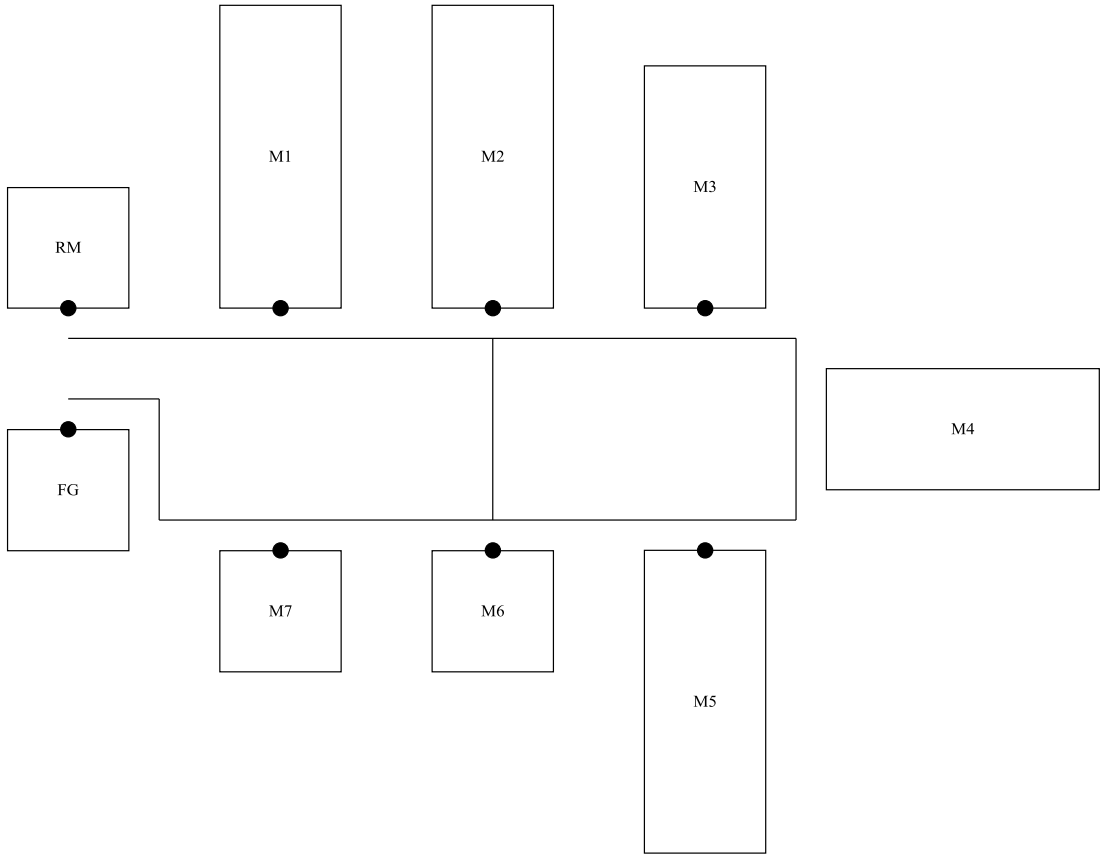


Figure 11-3: Manufacturing Cell

The cell is responsible for producing 1000 units of one part each day. The cell will operate for two shifts of 460 minutes each. Thus the takt time is computed using equation 11-1 to be:

$$\text{takt time} = \frac{\text{available work time per day}}{\text{demand per day}} = \frac{460 \ X \ 2}{1000} = 0.920 \text{ minutes} = 55.2 \text{ seconds}$$

The number of workers needed in the cell can be determined as follows. Notice that the total manual operation time, the time a worker is required, is shown in Table 11-1 to be 65 (= 45 + 20) seconds. This total time divided by the takt time (65 / 55.2) is between 1 and 2. Thus, a minimum of two workers is required.

In addition, worker walking time must be taken into account. It is highly desirable to have workers walk a circular route. Walking time plus manual task time for the route must be less than the takt time. Any assignment should seek to balance the manual operation plus walking time among the workers. Workers walk on the average of 2 feet per second.

Table 11-2 shows the walking distance between adjacent workstations.

Table 11-2: Walking Distances Between Workstations

Workstation / Task ID	Workstation / Task ID	Walking Distance (feet)
RM	M1	7
M1	M2	7
M2	M3	7
M2	M6	8
M3	M4	8
M4	M5	8
M5	M6	7
M6	M7	7
M7	FG	10

One possible assignment using two workers is the following (Assignment A):

Worker 1: RM, M1, M2, M7, FG
(Task time, 31 seconds; walking time, 21.5 seconds; total time, 52.5 seconds)

Worker 2: M3, M4, M5, M6
(Task time, 34 seconds; walking time, 19 seconds; total time, 53 seconds)

The standard work cell design procedures did not take into account the following factors that may proof to be significant in the operation of the cell:

1. Walking times are modeled as triangularly distributed random variables with the minimum equal to 75% of the mean and the maximum equal to 125% of the mean. Based on the VUT equation, this could add to the cycle time and WIP in the cell. Thus, the effect of random walking times needs to assessed.
2. There is concern as to whether a constant time between arrivals of parts from another area of the plant can be achieved. The practical worst case assumptions (Hopp and Spearman, 2007) lead to modeling the time between arrivals as exponentially distributed with mean equal to the takt time. Again by the VUT equation, considering the time between arrivals to be a random variable could add to the cycle time and WIP in the cell. Thus, the performance of the cell for the case of a constant interarrival time for parts must be compared to the case of an exponentially distributed interarrival time.
3. The following operational rule will be employed. Worker 1 will wait at the raw material station and worker two will wait at station M2 until a part is available to walk to the next station.

The simulation study must show that the above assignment is feasible, given the three operational factors. Furthermore, the utilization of workers in the proposed assignment scheme is very high, 95% for worker 1 and 96% for worker 2. It is possible that it is not feasible to effectively co-

ordinate the tasks of both workers and the operations of the machines. Thus as alternative assignment was proposed (Assignment B).

Worker 1: RM, M1, M2 (Total time, 41 seconds)

Worker 2: M3, M4, M5 (Total time, 43 seconds)

Worker 3: M6, M7, FG (Total time, 42 seconds)

Each worker has several tasks. Each task must be performed in sequence as the worker walks around the cell. For example, worker 1 in assignment A has the following sequence of tasks:

1. Wait for part at RM
2. Process part at RM
3. Move part from RM to M1
4. Unload previous part from M1
5. Initiate part on M1
6. Move unloaded part from M1 to M2
7. Remove part from M2
8. Initiate part unloaded from M1 on M2
9. Walk without a part to M6
10. Wait for an inspected part
11. Walk with an inspected part from M6 to M7
12. Remove part from M7
13. Initiate inspected part on M7
14. Walk with part removed from M7 to FG
15. Process part at FG
16. Walk with no part to RM

The following priorities are of fundamental importance in achieving one piece flow and not loosing machine capacity. These are reflected in the worker task sequence.

1. After removing a part from a machine, a worker will start another part on the same machine if one is available before performing any other task.
2. After walking a part from a preceding workstation or inventory and upon arriving at the next workstation, the worker will initiate the operation on a part, if the machine is available.

From the point of view of a part, the work cell will operate in the following way. Parts arrive to the raw material inventory from another area of the plant. The average time between arrivals is equal to the takt time.

Parts move through the same processing steps at each workstation except M6: initiation on the machine by a worker, automated processing by the machine, and removal from the machine by a worker. Processing at M6 is consists of one manual inspection step.

Workers move parts between machines as well as from the raw materials inventory to the first workstation and from the last workstation to the finished good inventory. Part processing and movement is constrained by the availability of workers and machines.

11.3.2 Build Models

The model will be built from the perspective of worker movement. A worker walks between stations in a prescribed route and performs one or two tasks at each station. Parts reside in inventories. A typical station has the following inventories: waiting for initiation on a machine, waiting for unloading from a machine, and waiting to be walked to the next station. A worker action changes the number of parts in one of the inventories.

The model consists of four processes:

1. Arrival of parts to the raw material inventory.
2. Worker 1
3. Worker 2
4. Automated processing on a machine which does not require worker assistance.

The following inventories exist in the model.

1. Workstations M1 – M5 and M7: waiting for initiation on a machine (WaitInitialize), waiting for unloading from a machine (WaitUnload), and waiting to be walked to the next station (WaitWalk).
2. Workstation M6: waiting to be walked to the next station (WaitWalk).
3. Raw materials: (RMInv)
4. Finished goods: (FGInv)

Entities in the arrival of parts process and the automated processing process represent parts. For the latter process, entity attributes are:

1. ID: ID number of the workstation station where automated processing occurs: 1, 2, 3, 4, 5, or 7.
2. OpTime: Processing time at the workstation.

The worker is the only entity in the worker process. This entity has one attribute:

1. WithPart: 1, if the worker has a part when walking between workstations and zero otherwise.

The following variables are used in the model.

1. WIPCell: The total number of parts in the work cell
2. WalkTime (9, 9): Average walking time between each pair of stations, FG (8), and RM(9).

The part arrival process follows. A part arrives and the RM inventory is increased by 1 as well as the total WIP in the cell.

```
Define Arrivals:
    Parts
        Time of first arrival: 0
        Time between arrivals: Exponentially distributed with a mean of 55.2 seconds
                Number of arrivals: Infinite

Define Resources:
    M(7)/1        with states (Busy, Idle)     // Workstation resources

Define Entity Attributes:
    WorkstationID      // ID number (1, …, 7) of workstation to process a part
    OperationTime      // Time to process a part at a workstation
    WithPart           // The number of parts carried by a worker from station to station (0, 1)

Define State Variables
    WIPCell            // The amount of work-in-process in the cell
    RMInv              // Raw material inventory
    FGInv              // Finished goods inventory
    WaitInitialze(7)   // Number of items waiting for initialization at a workstation
    WaitUnload(7)      // Number of items waiting unloading at a workstation
    WaitWalk(7)        // Number of items wating to be walked to the next workstation
    WalkTime (9, 9)    // Inter-station walk times

Process PartArrival
Begin
    WIPCell ++
    RMInv ++
End
```

The automated processing process is exactly the same as the single worker station process discussed previously. Upon the completion of processing, the number of parts waiting to unload is increased by one.

```
Process AutomatedMachine
Begin
    WaitUntil M(WorkstationID)/1 is Idle in Queue QM(WorkstationID)
    Make M(WorkstationID)/1 Busy
    Wait for OperationTime
    Make M(WorkstationID)/1 Idle
     WaitUnload(WorkstationID) ++
End
```

The process for Worker 1 starting at RM through arrival at workstation M2 follows. Note that the worker waits at RM for a part to carry to workstation M1. Otherwise, the worker will carry a part between workstations, unload a part or initialization a part only if a part is available. Each inventory is updated as the worker acts.

```
Process Worker1
Begin
    // From raw material inventory to M1
    Wait until RMInv > 0    // Wait for the next part
    Wait for 4 seconds      // Processing at raw material inventory
    RMInv –                 // Update raw material inventory
    WithPart = 1            // Worker carrying one part
    Wait for triangular WalkTime(9,1)*75%, WalkTime(9,1), WalkTime(9,1)*125%  // To M1
    WaitInitialize (1) ++   // Add to initialize inventory at M1

    // Processing at M1
    If WaitUnload(1) > 0 then
    Begin
        // Unload Part at M1
        Wait until M(1)/1 is Idle in Queue QM(1)
        Make M(1)/1 Busy
        Wait for 3 seconds
        Make M(1)/1 Idle
        WaitUnload(1)--
        WaitWalk(1)++
    End

    If WaitInitialize(1) > 0 then
    Begin
        // Initialize Part at M1
        Wait until M(1)/1 is Idle in Queue QM(1)
        Make M(1)/1 Busy
        Wait for 4 seconds
        Make M(1)/1 Idle
        WaitInitialize(1)—

        // Process part in parallel with worker walking
        ID = 1
        OperationTime = 23
        Clone to AutomatedMachine
    End

If WaitWalk(1) > 0 then
Begin
//  Walk with part
    WaitWalk(1) –
    WithPart = 1
End
Else WaitPart = 0       // No part

Wait for triangular WalkTime(1,2)*75%, WalkTime(1,2), WalkTime(1,2)*125%  // To M2
WaitInitialize(2) ++    // Arrive at M2 and Update Inventory
End
```

11.3.3 Identify Root Causes and Assess Initial Alternatives

Experimentation with the model is used to address the issues previously raised with respect to performance of the cell.

1. The effect of random walking times.
2. The effect of random times between arrivals.
3. The number of workers used in the cell: 2 or 3.
4. The effect of operational rules for workers.

The amount of work in process in the cell should be very low. Thus, the total WIP in the cell will be used as a performance measure. The WIP at RM is also of interest. In addition, a trace showing the time sequence of worker movements and activities is desired for both model and cell design validation.

The design of the simulation experiment is shown in Table 11-3. Since the cell is assigned a certain volume of work each day, a terminating experiment of duration one work day (920 minutes) is used. Twenty replicates are used. Random number streams are needed for worker walking time as well as the time between arrivals.

Table 11-3: Simulation Experiment Design for the Manufacturing Cell

Element of the Experiment	**Values for This Experiment**
Type of Experiment	Terminating
Model Parameters and Their Values	1. Time between arrivals (random or constant) 2. Number of workers (2 or 3)
Performance Measures	1. WIP in the cell 2. WIP at RM
Random Number Streams	1. Worker walking time 2. Time between arrivals
Initial Conditions	One part at each station
Number of Replicates	20
Simulation End Time	920 minutes (one day)

Initial conditions of that reflect the principle of one piece flow are appropriate. Thus, there is one part at each station initially. At all stations except M6, the part is placed in the WaitUnload inventory. At station M6, the part is placed in the WaitWalk inventory.

Simulation results for the cases when 2 workers are used are shown in Table 11-4.

The cell performs very well when the time between arrivals is constant. The maximum number of parts in the cell is 9, one more than the number of stations plus the raw material inventory. At most 1 part is in the raw material inventory. However, when the time between arrivals is exponentially distributed, large maximum WIP sizes are seen both in the cell in general and in the raw material inventory. Note, however, that the difference between the maximum WIP in the cell and the maximum WIP in the raw material inventory for each replicate is either 7 or 8. Thus, WIP is properly restricted to the raw material inventory.

Table 11-4: Work in Process in the Cell and in RM – Two Workers Cases

	Maximum WIP in Cell			Maximum WIP in RM		
Replicate	Constant Time between Arrivals	Random Time between Arrivals	Difference	Constant Time between Arrivals	Random Time between Arrivals	Difference
1	9	52	43	1	44	43
2	9	65	56	1	58	57
3	9	89	80	1	81	80
4	9	51	42	1	43	42
5	9	30	21	1	22	21
6	9	37	28	1	29	28
7	9	75	66	1	67	66
8	9	39	30	1	31	30
9	9	31	22	1	23	22
10	9	47	38	1	39	38
11	9	51	42	1	44	43
12	9	62	53	1	54	53
13	9	72	63	1	64	63
14	9	48	39	1	40	39
15	9	22	13	1	14	13
16	9	37	28	1	29	28
17	9	31	22	1	23	22
18	9	25	16	1	17	16
19	9	79	70	1	71	70
20	9	31	22	1	23	22
Average	9	48.7	39.7	1	40.8	39.8
Std. Dev.	0	19.4	19.4	0	19.5	19.5
99% CI Lower Bound	9	36.3	27.3	1	28.3	27.3
99% CI Upper Bound	9	61.1	52.1	1	53.3	52.3

Table 11-5 contains a portion of the trace for worker 1 for one replicate of the constant time between arrivals case. The trace shows the actions the worker takes from processing a part at RM to processing the next part at RM. The time between starting the processing of a part at RM and return was 52.06 seconds, only slightly less than the expected time of 55.2 seconds. Thus, there is some evidence that the worker can perform all assigned tasks in less than the takt time. The trace shows that worker performs all assigned tasks in the required sequence. Thus, model and system design validation evidence is obtained.

Table 11-5: Worker 1 Action Trace

Simulation Time	Worker	Workstation	Action
55.20	Worker1	RM	Start
59.20	Worker1	RM	End
62.64	Worker1	M1	Arrive
62.64	Worker1	M1	Unload Start
65.64	Worker1	M1	Unload End
65.64	Worker1	M1	Initialize Start
69.64	Worker1	M1	Initialize End
73.46	Worker1	M2	Arrive
73.46	Worker1	M2	Unload Start
77.46	Worker1	M2	Unload End
77.46	Worker1	M2	Initialize Start
82.46	Worker1	M2	Initialize End
85.98	Worker1	M6	Arrive
89.86	Worker1	M7	Arrive
89.86	Worker1	M7	Unload Start
92.86	Worker1	M7	Unload End
92.86	Worker1	M7	Initialize Start
97.86	Worker1	M7	Initialize End
103.13	Worker1	FG	Arrive
108.13	Worker1	FG	End
110.26	Worker1	RM	Arrive
110.40	Worker1	RM	Start

The same results for the case where three workers are used are shown in Table 11-6 along with a comparison to the two workers case. For the random time between arrivals case, the average maximum WIP in the cell is 32.0 when three workers are used. This is notably less than the average when two workers are used: 48.7. Similarly, the average maximum WIP at RM is less when three workers are used: 22.0 versus 40.8. The reductions in WIP in the cell and at station RM due to using three workers instead of two are statistically significant at an approximate 99% confidence level. The approximate 99% confidence intervals of the difference do not contain zero.

Table 11-6: Work in Process in the Cell and in RM – Three Workers Case with Comparison to the Two Workers Case for Random Times Between Arrivals

	Maximum WIP in Cell			Maximum WIP at RM		
Replicate	**Two Workers**	**Three Workers**	**Difference**	**Two Workers**	**Three Workers**	**Difference**
1	52	37	15	44	27	17
2	65	32	33	58	22	36
3	89	49	40	81	39	42
4	51	31	20	43	21	22
5	30	24	6	22	14	8
6	37	35	2	29	25	4
7	75	48	27	67	38	29
8	39	28	11	31	18	13
9	31	24	7	23	14	9
10	47	25	22	39	15	24
11	51	28	23	44	18	26
12	62	37	25	54	27	27
13	72	38	34	64	28	36
14	48	32	16	40	22	18
15	22	22	0	14	12	2
16	37	31	6	29	21	8
17	31	26	5	23	16	7
18	25	25	0	17	15	2
19	79	39	40	71	29	42
20	31	28	3	23	18	5
Average	48.7	32.0	16.8	40.8	22.0	18.9
Std. Dev.	19.4	7.6	13.3	19.5	7.6	13.4
99% CI Lower Bound	36.3	27.1	8.2	28.3	17.1	10.3
99% CI Upper Bound	61.1	36.8	25.3	53.3	26.8	27.4

11.3.4 Review and Extend Previous Work

Management was pleased with the results of the simulation experiments. The cell appears to work as designed using standard cellular manufacturing calculations when the time between arrivals is constant.

Exponentially distributed times between arrivals result in large maximum WIP levels. Controls placed on cell operations, in particular requiring a worker to wait at RM for a part, resulted in the all of the excess WIP residing in the RM. Thus, the cell appears to be capable of operating effectively even in the presence of random variation in part arrival.

11.3.5 Implement the Selected Solution and Evaluate

It was decided to implement the cell with two workers. If the high utilization of the two workers constrained the actual operation of the cell, a third worker could be added.

11.4 Summary

Manufacturing work cells can be designed using standard calculations. Simulation can be used to validate that the designed cell will operate as intended, in part using a trace of worker actions. The effect of random behavior, such as random times between arrivals and random walking times, can be assessed. WIP levels can be estimated. The use of alternative numbers of workers can be evaluated.

Problems

1. Compare the cellular manufacturing organization presented in this chapter with the serial line discussed in chapter 7.

2. Write down the task sequence for Worker 2 for worker assignment A.

3. Model using pseudo – code part of the process for Worker 2 from picking up a part at workstation M2 through arriving at workstation M4, for worker assignment A.

4. For the case of two workers and random time between arrivals, estimate the average cycle time for a part to traverse the cell using Little's Law. The cycle time is the time between entering the raw material inventory and entering the FGI. Suppose the average WIP in the cell is 30.8 with a 95% confidence interval for the mean: (22.6, 38.9).

5. Does the work cell behave like a CONWIP system? Why or why not?

6. An extremely long time between arrivals, say triple the mean, is possible when using the exponential distribution to model this quantity. What is the potential effect of such long times between arrivals on the capacity of the cell?

7. Consider the probability distribution of the time worker two takes to complete all tasks once. Assume walking times are normally distributed with same mean and variance as the triangularly distributed times. The mean and variance of a triangular distribution are computed as follows:

Mean: $$\frac{\min + \text{mode} + \max}{3}$$

Variance: $$\frac{\min^2 + \text{mode}^2 + \max^2 - \min * \text{mode} - \min * \max - \text{mode} * \max}{18}$$

   a. The time for worker two to complete all tasks once is normally distributed. Compute the mean and standard deviation of this distribution. Assume that the minimum is 75% of the mean and the maximum is 125% of the mean as stated on page 11-5. Thus, the distribution is symmetric implying mean = mode.

   b. What is the probability that the time to complete all tasks once is greater than the takt time?

8. Perform a gross capacity analysis for each station in the cell. This means computing the maximum number of parts each station can produce in one work day.

9. Print out a trace of the events effecting worker 2 in task assignment A. Does this provide validation evidence?

10. Add an additional performance measure to the model: The percent of times a worker traverses an assigned route in more than the takt time. Rerun the model to estimate this performance measure.

11. Model the time between arrivals as gamma distributed with mean 55.2 seconds and standard deviation 27.6 seconds. Compare the maximum WIP in the cell to the values in Table 11-4.

Case Problem[2]

An injector is produced in two steps: assembly and calibration. This study will focus on the calibration area only. The assembly area can produce a batch of 24 parts in 82 minutes. Each batch is placed on a WIP cart. A batch is only produced if a WIP cart is available. Injectors must be cured for 24 hours after assembly before they can enter the calibration area.

To control work in process, the number of WIP carts is limited to the fewest number need to avoid constraining throughput. Only one WIP cart can be in the calibration area at a time.

The calibration area consists of four workstations that can be labeled W1, W2, W3, and W4. Each workstation processes one injector at a time. A worker is not needed for automated operations and thus is free to due other tasks.

At workstation W1, the worker initiates the injector in 25 seconds. The workstation performs an automated test in 10 seconds. Finally, the worker removes the part in 5 seconds. A manual operation is performed at workstation W2. The operation time is triangularly distributed with minimum 4.0 minutes, mode 5.0 minutes, and maximum 7.8 minutes. At workstation W3, the worker initiates the part in 5 seconds. An automated operation is performed in 4.1 minutes. The worker removes the part in 2 seconds. Workstation W4 is a packing operation performed by the worker in 5 seconds.

The calibration area is served by one worker. Worker walking times between stations are as follows:

Station	W1	W2	W3	W4
W1	0	3	7	10
W2		0	4	7
W3			0	3
W4				0

Determine the number of WIP carts required. Generate a trace of worker tasks to validate the model.

Case Problem Issues

1. How should the WIP carts be modeled?

2. How should the constraint on the number of WIP carts in the calibration area be modeled?

3. How should the injector curing requirement be modeled?

[2] This application problem is derived from the capstone masters degree project performed by Carrie Grimard.

4. Write down the sequence of tasks for the calibration area worker.

5. Discuss how the number of WIP carts can effect cycle time.

6. Beside throughput, are any other performance measures important? If so, what are they?

7. An entity moving through the assembly area represents a WIP cart while an entity moving through the calibration area represents an individual injector. How is the conversion from WIP cart to injector accomplished?

8. Specify the experimental strategy for determining the number of WIP carts.

9. Discuss how to obtain verification and validation evidence.

10. Determine how to model arrivals to the assembly process.

11. Determine how to model batch processing times in the assembly area. Note that the batch processing times are the sum of 24 individual processing time. Should this sum be explicitly computed? Can the central limit theorem be applied?

12. Compute the expected number of WIP carts needed to maximize throughput.

13. What are the initial conditions for the experiment?

Terminating Experiment: The simulation time interval is 5 days (120 hours of work time).

Chapter 12

Flexible Manufacturing Systems

12.1 Introduction

Consider a manufacturing facility that is required to produce multiple part types. Demand is insufficient to warrant a dedicated work cell for any part type. However, demand for all part types together is sufficient to potentially justify automating the production process.

What is needed in this case is a flexible manufacturing system (FMS). Such a system operates efficiently and cost effectively regardless of the mix of part types produced. It is comprised of flexible machines that perform a range of operations on a variety of parts with only minor setup required when switching between part types. Such machines must be programmable or computer numerically controlled (CNC). They must be capable of storing, automatically setting up (loading), and using a variety of tools. A new part type could be introduced without significant additional capital investment, at least if it was sufficiently similar to existing part types.

An FMS requires automated material handling capabilities to move parts between machines as well as into and out of the system. An FMS must be highly automated and thus requires co-ordinated, computer based control.

The initial capital cost of an FMS is high relative to a work cell dedicated to a single part. This investment is worthwhile if the FMS can effectively produce a mix of part types more economically than can a set of dedicated work cells, one per type of part. A flexible manufacturing system could have as many as 20 machines. A system consisting of one or two flexible machines is called a cell.

An FMS operates in a similar manner to a work cell. A part arrives to a single load-unload station where it is attached to a fixture that is mounted on a pallet. More than one part could be attached to the fixture. Parts need not be batched by type upon arrival since machines are able to adapt to processing different part types with relatively little setup time. Measuring the WIP is important since the WIP level is proportional to the number of pallets and fixtures needed.

Since machines are flexible, more than one machine can perform each operation the part requires. Thus, breakdowns do not hamper the operation of an FMS to the same degree as for a dedicated work cell.

One aspect of the design of an FMS is the scheduling of parts on specific machines. In this case study, the assignment of parts to machines using an optimization algorithm is compared with the use of a heuristic dynamic scheduling rule. The performance measure of interest is the total time to produce a given number of parts.

12.2 Points Made in the Case Study

This FMS case study shows how optimization models and simulation models can be used together to address system design and operation issues. An optimization model is used to assign parts, and consequently the tools need to operate on the parts, to machines. A simulation model is used to assess how well the system operates using this assignment.

In modeling an FMS system, each machine and each tool is modeled as a resource. In addition, the model must keep track of which tool is on which machine. Thus, complex logic for assigning resources representing machines and tools to work on parts represented by entities is required. How to choose between alternative resources must be specified. Here, a choice between machines to perform the same operation must be made.

The simulation model must include the concurrent use of two resources, one a machine and another a tool. Furthermore, a selection among available machines must be made. Such logic may be coded in a high level programming language, such as C, and referenced from the simulation model.

Complex system control logic may be included in a simulation model. In this FMS model, a part waits to begin an operation until the required tool and any of the machines that can operate on it are available. A list of such waiting parts must be maintained. When tool and machines resources enter the free state, the model must search the entire list of waiting parts until one that can be processed by the newly freed resources is found. It is possible that no such part will be found. If the simulation engine does not examine each entity waiting for a resource, then logic to search the entire list of waiting transactions must be specified by the modeler.

Previously, arrivals in a model represented a single entity entering a system. In this case study, parts arrive to the FMS in a batch. Thus, an entity arrival in the model represents a batch of parts. An arriving entity is cloned to create one entity for each part.

A system may be sufficiently complex that simulation is necessary for analysis, design, and understanding even if no quantities are modeled using probability distributions. This is due to the structural variability in the system. Pritsker (1989) estimated that about one-third of all simulation projects employ deterministic models. In this case study, the assignment of tools and part operations to machines over time is sufficiently complex that an intuitive understand of system dynamics is not possible. Simulation is necessary to assess the affect of proposed assignments on system operations.

The length of the simulation run need not be specified as a constant time but could be specified as a condition to be met. In this case, the ending simulation time is the makespan of a production run, a performance measure of interest.

12.3 The Case Study

A flexible manufacturing cell consists of three types of flexible machines. Each type of machine performs a different set of operations. The cell must process three part types. Each operation required by a part type uses one particular tool and can be performed on any of multiple machine types. Not all machine types can perform all operations on all part types. Operation times vary by machine type. A tool is loaded on one of the machines at a time and may be moved between machines as needed.

Figure 12-1 shows the system completely idle before a production run is made. Tool bins hold the tools currently assigned to each machine. The state of each machine, in terms of the type of part being processed, is shown. The system contains two type A machines, two type B machines, and one type C machine.

Production of 240 parts in one day is of interest. Parts arrive in batches of 24 every 75 minutes starting at the beginning of the day. The mix of part types in each batch is the same: 4 of type 1; 10 of type 2; and 10 of type.

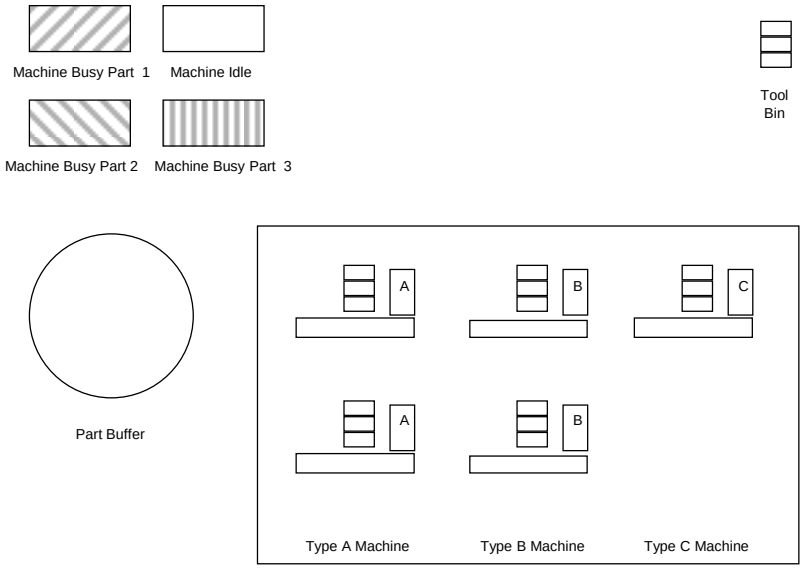


Figure 12-1: Flexible Manufacturing System

Management wishes to minimize the makespan for the 240 parts as well as in-process inventory. Recall that the in-process inventory level is proportional to the number of fixtures and pallets the FMS requires. The lead time for parts in the FMS is of interest. The utilization of each machine is important.

Part and tool movement between machines requires 30 seconds. Machine setup time is minimal and can be ignored.

Table 12-1 presents the operation time data.

Table 12-1: Operation Times for the FMS System

			Operation Time (Min)			
Part Type	**Parts to Produce / Day**	**Operation ID**	**Machine Type A**	**Machine Type B**	**Machine Type C**	**Tool**
1	40	1	12	11	10	A
		2	13	15		B
		3	14	14		C
2	100	1	2	4		A
		2	2	6	6	C
3	100	1	4			D
		2	5		8	E
		3			4	F

12.3.1 Define the Issues and Solution Objective

The method for assigning a part operation, as well as the tool required by that operation to a machine, must be determined. Two schemes are proposed.

1. Assign the part operation and tool to the IDLE machine with the shortest processing time for that operation at the time the part is ready to begin the operation. (Dynamic Scheduling).
2. Assign each part operation to one and only one machine type using the machine loading heuristic in Askin and Standridge (1993), pp. 144-153.

Regardless of the scheme used, movement of parts and tools between machines will be minimized. The subsequent operation on a part will be performed on the same machine as the current operation if subsequent operation is allowed on that machine and the required tool is already loaded on the machine. Whether the same machine can perform the subsequent operation will be determined when the current operation is completed.

Note that the first scheme uses any machine that can perform the operation on a part. It selects between IDLE machines based on operation time, smallest time first. It seeks to avoid part waiting and to minimize the waiting time for each operation.

The second scheme seeks to balance the work load among the machines. It will make a part wait for its assigned machine type even if a machine of another type is IDLE and could process the part.

The priority order of machines for each operation on each part type using the dynamic scheduling approach with the machine having the shortest processing time given priority is shown in Table 12-2, which directly results from the data in Table 12-1.

Table 12-2: Machine Priority -- Shortest Processing Time First Scheme

Part Type	Operation ID	First Priority	Second Priority	Third Priority
1	1	C	B	A
	2	A	B	
	3	B	A	
2	1	A	B	
	2	A	C	B
3	1	A		
	2	A	C	
	3	C		

The machine loading heuristic of scheme 2 seeks to equalize the workload between machine types. Results of applying the optimization algorithm to assign operations to machine types are given in Table 12-3.

Table 12-3: Machine Priority -- Equalize Workload Among Machine Types Scheme

Part Type	Operation ID	First Priority	Second Priority	Third Priority
1	1	C		
	2	B		
	3	B		
2	1	A		
	2	A		
3	1	A		
	2	A		
	3	C		

Note the difference between the two schemes. The first priority machine type for each is the same, except for operation 2 on part type 1. In the first scheme, a part proceeds to a second or third priority machine if the first priority machine is busy. In the second scheme, a part simply waits if the first priority machine is busy.

12.3.2 Build Models

The model includes arrivals of batches of parts as well as decomposing the batches into individual parts. Thus, arriving entities represent batches of parts and subsequent entities represent parts. The part entities have the following attributes:

ArrivalTime = Time of arrival to the FMS
PartType = Part type
Machine = Machine used in current operation
Tool = Tool used in current operation
OpTime = Operation time for current operation = f(machine used)
CurrentOp = ID number of the current operation (1, 2, 3)

The model of an operation on a part requires two resources, one representing a machine and the other a tool. Movement between machines must be included in the model both for parts and tools. An procedure to select the machine to perform an operation on a part is included as well as a second procedure to determine whether the machine on which a part currently resides can perform the next operation.

Each tool resource has an attribute: ToolLocation = The machine on which it currently resides.

The pseudo code for the arrival process follows. A batch of 24 parts arrives every 75 minutes. Ten batches arrive in total. Each batch is separated into component parts: 4 of type 1, 10 of type 2, and 10 of type 3. Each part is sent to a process that models the first operation that is performed on it.

```
Define Arrivals:
    Batches
        Time of first arrival:      0
        Time between arrivals:      75 minutes
        Number of arrivals:         10

Define Resources:
    // Flexible Machine Resources
    MachA_1/1       with states (Busy, Idle)
    MachA_2/1       with states (Busy, Idle)
    MachB_1/1       with states (Busy, Idle)
    MachB_2/1       with states (Busy, Idle)
    MachC_1/1       with states (Busy, Idle)

    // Tool Resources
    ToolA/1         with states (Busy, Idle)
    ToolB/1         with states (Busy, Idle)
    ToolC/1         with states (Busy, Idle)
    ToolD/1         with states (Busy, Idle)
    ToolE/1         with states (Busy, Idle)
    ToolF/1         with states (Busy, Idle)
```

```
Define StateVariables:
    WIPCount        // Number of parts in FMS
    ToolLocation(5) // Tool location

Define Lists:
    OpOneList
    OpTwoList
    OpThreeList

Define Entity Attributes:
    ArrivalTime     // Time of arrival to the FMS
    PartType        //  Part type
    Machine         //  Machine used in current operation
    Tool            //  Tool used in current operation
    OpTime          //  Operation time for current operation = f(machine used)
    CurrentOp       //  ID number of the current operation (1, 2, 3)

Process BatchArrival
Begin
    ArrivalTime = Clock     // arrival of a batch of 24 parts
    CurrentOp  =  1
    PartType = 1            // type 1 parts
    Clone 4 OpFirst
    PartType = 2            // type 2 parts
    Clone 10 OpFirst
    PartType = 3            // type 3 parts
    Clone 10 OpFirst
End
```

Each part requires either two or three operations. The processes for each of the operations are similar but not identical. Each includes two essential steps: the transportation of the part and tool to the machine, if required, followed by the actual operation on the part. Two resources, a machine and a tool, are required. Which machine is determined by the machine assignment scheme employed.

At the beginning of the model of the first operation, the machine selection procedure is used to determine if the required tool is available. If so, the procedure determines if any machine is available to process the part and if so which one should be employed. The machine selection procedure is as follows:

```
MACHINE SELECTION PROCEDURE (OPERATION_NUMBER, PART_TYPE)
{
        IF THE REQUIRED TOOL RESOURCE FOR THE PART_TYPE FOR
        OPERATION_NUMBER IS IN THE IDLE STATE
        {
           FOR EACH MACHINE TYPE IN PRIORITY ORDER
           {
              IF ANY MACHINE OF THAT TYPE IS IN THE IDLE STATE
              {
                 Machine = RESOURCE ID OF SELECTED MACHINE
                 Tool    = RESOURCE ID NUMBER OF REQUIRED TOOL
                 OpTime  = OPERATION TIME FOR PART FOR OPERATION_NUMBER
                 RETURN
              }
           }
        }
}
```

If the required tool is not free or no machine is available to perform the operation, the part entity must wait in a buffer. There is one buffer in the model for each of the three operations. When a tool or machine resource becomes free, the model will search each buffer to find a part to process.

If the required tool and a machine are available to process the part, the tool and machine resources enter the busy state. Part entity attributes are assigned the name of the tool and machine as well as the process time for the operation on the selected machine. A time delay to move the tool and or part to the selected machine is incurred. The tool attribute recording its location (ToolLocation) is assigned the name of the selected machine. The operation is performed on the part. When the operation is completed, the tool resource enters the IDLE state. The lists of part entities waiting for operations 1, 2 or 3 are searched and the processing of another part is begun if possible.

The pseudo code modeling the first operation follows.

```
Process OpFirst
Begin
    WIPCount ++              // Add one to number of parts in FMS
    MachineSelection (CurrentOp, PartType)
    If Machine is Null then Add entity to list Op1List
    Else
        Begin
            // Perform first operation
            Get Tool
            Get Machine
            ToolLocation (Tool) = Machine
            Wait for 30 seconds  // Tool and part movement
            Wait for OpTime      //  Perform operation
            Free Tool
            SearchforNextPart   // Procedure to search all lists for next part to use tool
        End
    Send to OpSecond
End
```

The model of the second operation begins by determining if the part can remain on the same machine using the subsequent machine procedure. This will be the case if the machine is able to perform the operation and the tool required for that operation is available.

If the part cannot remain on the machine used for the first operation, the resource modeling this machine enters the idle state. The lists of part entities waiting for operations 1, 2 or 3 are searched and the processing of another part is begun if possible. The machine selection procedure is used to attempt to find a machine to process the part entity completing the first operation in the same way as was done for the first operation.

If the second operation can be performed on the same machine as the first, the tool resource for this operation enters the busy state, the operation is performed, and the tool resource enters the idle state. The lists of part entities waiting for operations 1, 2 or 3 are searched and the processing of another part is begun if possible. Type 2 parts do not require a third operation. Thus, at the end of the second operation, the machine resource processing a type 2 part enters the idle state and the search of the lists of waiting parts is conducted. The pseudo code for the second operation follows. The model of the third operation is similar to the model of the second operation.

```
SUBSEQUENT MACHINE PROCEDURE (OPERATION_NUMBER, PART_TYPE,
CURRENT_MACHINE)
{
     IF THE REQUIRED TOOL RESOURCE FOR OPERATION_NUMBER ON THIS PART IS
     IN THE IDLE STATE
     {
        IF THE REQUIRED TOOL RESOURCE FOR OPERATION_NUMBER ON PART_TYPE
        IS LOADED ON CURRENT_MACHINE AND CURRENT_MACHINE CAN PERFORM
        OPERATION_NUMBER ON THIS PART_TYPE
        {
           THE REQUIRED TOOL RESOURCE ENTERS THE BUSY STATE
           Tool = RESOURCE ID NUMBER OF REQUIRED TOOL
           OpTime = OPERATION TIME FOR PART_TYPE FOR OPERATION_NUMBER
        }
     }
}
```

```
Process OpSecond
Begin
   CurrentOp = 2
   SubsequentMachine (CurrentOp, PartType, Machine)
   If Tool is Null then
   Begin
      // Move to another machine
      Free Machine
      SearchforNextPart   // Procedure to search all lists for next machine to use tool
      MachineSelection (CurrentOp, PartType)          // Find another machine
      If Machine is Null then Add entity to list Op2List
      Else
         Begin
            // Perform second operation on another machine
            Get Tool
            Get Machine
            ToolLocation (Tool) = Machine
            Wait for 30 seconds  // Tool and part movement
            Wait for OpTime      //  Perform operation
           Free Tool
            SearchforNextPart   // Procedure to search all lists for next part to use tool
         End
   End
   Else
      Begin
         // Perform second operation on current machine
         Get Tool
         Wait for OpTime      //  Perform operation
        Free Tool
         SearchforNextPart   // Procedure to search all lists for next part to use tool
      End
   Send to OpThird
End
```

12.3.3 Identify Root Causes and Assess Initial Alternatives

The simulation experiment will determine both the makespan for 240 parts as well as the number of pallets and fixtures required. The latter can be accomplished by measuring the number of parts in the FMS as was previously discussed.

The design of the simulation experiment is summarized in Table 12-4. We are interested in the time to produce 240 parts. Thus, a terminating experiment with initial conditions of no parts in the system is appropriate. Since no quanities are modeled using probability distributions, no random number streams are needed and one replicate is sufficient. The two schemes for assigning parts to machines identified above are to be simulated. Performance measures have to do with the time to complete production on 240 parts, the lead time for parts, the number of parts in the FMS (WIP), and the utilization of the machines.

Results are shown in Table 12-5.

Table 12-4: Simulation Experiment Design for FMS Machine Loading

Element of the Experiment	Values for This Experiment
Type of Experiment	Terminating
Model Parameters and Their Values	Machine load scheme used: 1. Available machine with the shortest processing time 2. Machine loading heuristic from Askin and Standridge
Performance Measures	1. Time to produce 240 parts 2. Number of parts in the FMS (WIP) 3. Part Lead Time 4. Machine Utilization
Random Number Streams	None
Initial Conditions	Empty buffers and idle stations
Number of Replicates	1
Simulation End Time	Time to produce 240 parts

Table 12-5: Simulation Results for FMS Machine Loading

	Loading Scheme	
Performance Measure	**Shortest Processing Time First**	**Balance Machine Type Workloads**
Makespan (Minutes)	1099	900
WIP –		
Average	30.8	46.2
Maximum	63	86
Lead Time (Minutes) –		
Average	141	173
Standard deviation	114	74
Machine Utilization –		
Type A	69.5%	82.8%
Type B	72.5%	67.5%
Type C	70.0%	96.0%

The makespan for the balance machine type workloads approach is 199 minutes less than for the shortest processing time first approach. The former approach results in higher utilizations for machine types A and C as well as a lower utilization for machine type B. Recall that operations were assigned to machine types A and C instead machine type B since the operation times for machine type B most often were higher than for the other two types.

Recall from the VUT equation that increasing the utilzation results in a longer lead time at a station. Thus, it could be expected that the balance machine type workloads approach would have a longer lead time than the shortest processing time first approach. In addition, Little's Law

indicates that the WIP is proportional to the lead time and thus could also be larger. However, the balance machine type workloads scheme reduces the standard deviation of the lead time.

The maximum number of parts in the FMS is higher under the balance machine type workloads approach. This means that more fixtures and pallets are required using this approach.

12.3.4 Review and Extend Previous Work

Management accepted the simulation results presented in the previous section and decided to use the balance machine type workloads scheme. This decision was primarily based on the need to mimimize makespan. The cost of additional fixtures and pallets will be bourn to support this approach.

12.3.5 Implement the Selected Solution and Evaluate

During system operation, the time to produce a required batch of parts and the number of parts in the FMS will be monitored.

12.4 Summary

This case study shows how ad hoc operating rules, such as use the idle machine with the shortest processing time, are often inferior to operation rules developed using formal models. Simulation is used to test alternative rules and quantify the difference in their effects. Because systems are complex, simulation is needed even when such system operating models are deterministic. Complexity arises from the concurrent use of multiple resources such as tools and machines as well as the ability of resources such as machines to serve multiple tasks. The need for the model to organize and manage entities waiting for such resources instead of relying on the simulation engine to do so transparently to the model has been demonstrated.

Problems

1. Provide validation evidence for the FMS machine loading simulation based on the tool and machine utilization. Compute the expected utilization of each tool and machine type. Compare these results to the simulation results for the balance machine type workloads case. The machine utilization is shown in Table 12-5.

	Tool Utilization
A	74.2%
B	68.9%
C	93.8%
D	50.5%
E	59.8%
F	49.5%

2. Tell why the standard deviation of the time parts spend in the FMS (lead time) is greater than 0 since there are no random variables in the model.

3. Is it proper to compute a t-confidence interval for the mean part lead time? Why or why not?

4. Defend the use of the shortest processing time first loading scheme based on the simulation results shown in Table 12-5.

5. List service systems that you have encountered that have the flexibility characteristics of the manufacturing system discussed in this chapter.

6. Compare the model of the FMS to the model of the serial system discussed in chapter 7.

7. Write the model of third operation on a part in pseudo-code.

8. The model in this case study assumes that tools and parts can be moved concurrently to a machine. Modify the process model so that first the part is moved then the tool. Movements occur only if the part or tool is not resident on the selected machine.

9. Resimulate the model with all parts available for processing at time 0 and compare the results to those in Table 12-5.

10. Simulate the following improved version of the shortest processing time first rule and compare the results with those in Table 12-5. Don't allow operation 1 on machine type A.

11. Assess the effect of operation clustering on machine loading. Operations are assigned to specific machines, not just machine types. All part type 2 operations are assigned to one type A machine along with the first operation for part type 3. The second operation for part type 3 is assigned to the other A machine. The third operation for part type 3 is assigned to the type C machine. The first operation for part type 1 is assigned to the type C machine, the second to one type B machine, and the third to the other type C machine.

12. Develop and test heuristics that embellish the operation clustering based assignment given in the previous problem. For example, allow the least utilized machine to be a second priority for the most utilized machine if feasible.

13. Management will purchase one more tool of any of the six tool types if that will help shorten makespan. Which tool should be purchased? Evaluate your choice using the simulation model developed in this chapter.

14. Test the idea that a part should stay on the same machine as long as it is feasible for that machine to perform the next operation on the part. In this case, the required tool is moved to that machine as soon as it is idle.

Case Problem

A flexible manufacturing facility must produce 1680 parts of one type per 80-hour lead (Wortman and Wilson, 1984; Kleijnen and Standridge, 1988). The part flow through the facility is follows.
1. Parts arrive to the facility from a lathe at a constant rate of 21 per hour.
2. Parts require three operations in the following sequence: Op10, Op20, and Op30.
3. A part is washed before and after each operation.

Each operation is performed either by a fixed machine or a flexible machine. A fixed machine can perform one and only one of three operations but a flexible machine can perform any of the three operations. Parts are moved between machines and the wash station by a single automated guided vehicle (AGV). The AGV system transports parts with little or no human assistance. The vehicle picks up loads at designated pick-up points and transports them to designated drop-off points. Each pick-up and drop-off point is associated with a machine or work station. A central computer assigns material movement tasks to the AGV and monitors the vehicle position. The AGV moves in one direction on a fixed track around the center of the system.

Operation processing times are 14.0, 5.0, and 8.0 minutes respectively for Op10, Op20, and Op30. Washing time is 18 seconds.

AGV travel time is 20 seconds around the entire loop. The following table shows AGV travel time between each pair of workstations.

	Wash Station	OP 10	OP 20	OP 30	Flexibile
Wash Station	0	5	9	15	11
OP10	5	0	4	10	6
OP20	9	4	0	15	11
OP30	15	10	15	0	16
Flexible	11	6	11	16	0

Figure 12-2 gives an overview of the system at the beginning of the production period with all machines idle and no parts in the system. Part movement by the AGV is indicated. The particular operation performed by each machine is displayed.

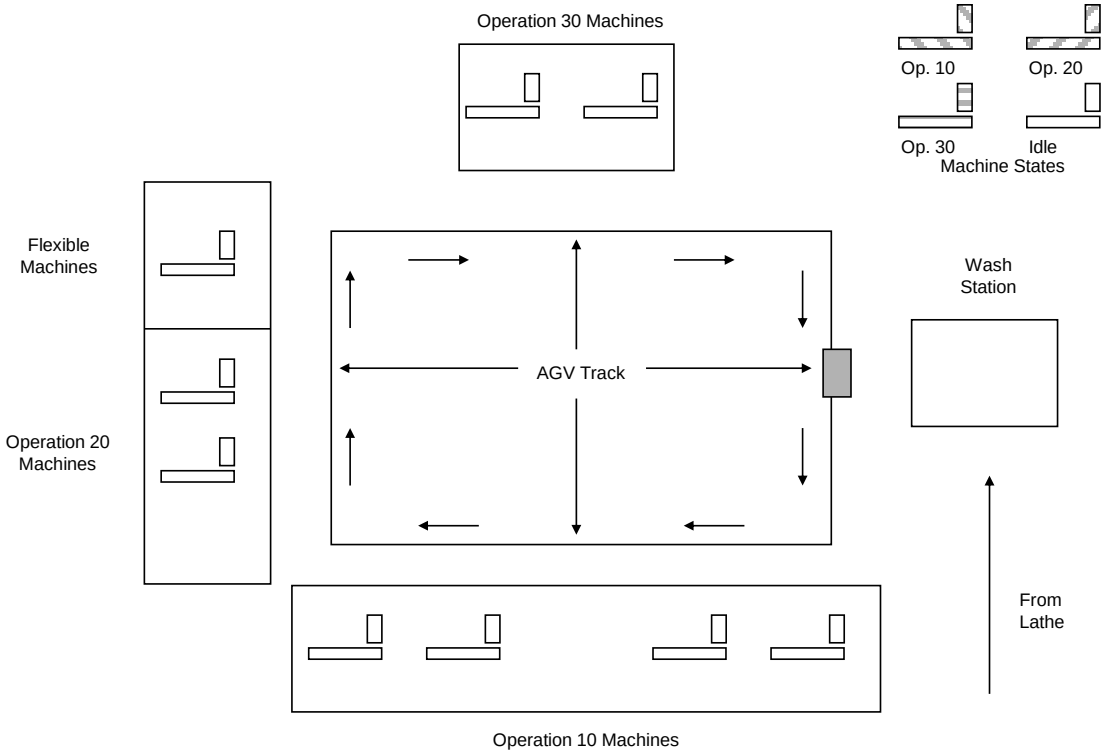


Figure 12-2: Overview of the Flexible Manufacturing System: (4,2,2,1) Configuration.

The objective is to find the minimum cost combination of fixed machines (Op10, Op20, or Op30 only) and flexible machines (all three operations) that meets the throughput requirement. Flexible machines cost more than fixed machines. Thus, all possible work should be done by fixed machines. Flexible machines are employed to avoid buying an excessive number of fixed machines. In addition, management is also interested in minimizing lead time for a part. Thus, management will consider a configuration of machines that includes flexible machines and increases cost in order to reduce lead time as long as the total number of machines does not exceed the total number of fixed machines required to do the work by more than one.

The following operating rule is employed for each operation to select between fixed and flexible machines. A part will use a fixed machine if it is available. If not, it will use a flexible machine if one is available. If neither a fixed machine nor a flexible machine is available, the part will use the first machine of either type, fixed or flexible, that becomes available.

One possibility that should be considered is using the minimum number of fixed machines needed to process all parts in a timely fashion with no flexible machines utilized.

Assess the structural variability seen in this system. Generate a trace of all system activities. Use the trace to identify the structural variability.

Case Problem Issues

1. What performance measure are important in this problem?

2. How will the AGV system be modeled?

3. Describe how to model the choice between a fixed and a flexible machine for an operation in the simulation language you are using.

4. When does a part entity acquire and free a machine resource relative to acquiring and freeing the AGV resource?

5. How can the effect on system operations of part waiting for the AGV be determined.

6. How many fixed machines of each type are needed if no flexible machines are used?

7. Construct an alternative to the machine configuration in number 6 as follows. Replace one fixed machine of each type with a sufficient number of flexible machines. Determine the number of flexible machines in this case.

8. Why tell why all parts do not have the same time in the system.

Part IV
Supply Chain Logistics

Previous parts of this book discussed fundamental organizations of systems and strategies for operating those systems. This part deals with supply chain management which is defined by Hopp and Spearman (2007) to be: The overall system wide co-ordination of inventory stocks and flows to ensure the purpose of inventories is met with minimal dollar investment.

A publication from Jones-Lang-LaSalle (2008) defines the lean supply chain as "a set of organizations directly linked by upstream and downstream flows of products, services, finances and information that collaboratively work to reduce cost and waste by efficiently and effectively pulling what is required to meet the needs of the individual customer." The focus of part IV is on lean supply chain logistics that is the flow of product between organizations to minimize inventory and the cost of movement, particularly transportation equipment such as trucks and rail cars. A key element of lean supply chain logistics is demand management: Providing products and services when requested (pulled) by the customer. Thus, movement of product is a function of customer demand.

The use of modeling and analysis in achieving lean supply chain logistics with proper demand management is discussed. Ideally, there would be zero inventory. Product would be instantaneously delivered to customers who would immediately consume it upon arrival. There would be no in process inventory except for items currently being operated upon.

However, inventory must be kept to deal with variation, both random and structural (generated by design), in demand, production, and delivery. All inventory raises costs without adding value to a product. Thus, the management of inventory involves trading off lower costs with having enough product, raw material, and partially finished goods to meet customer demands and keep operations working.

Chapter 13 discusses the management of a retail store inventory by a supplier. Detailed operations are not included. Only daily demand and production volumes are modeled. Inventory levels as well as production schedules are determined. Analytic computations aid in setting inventory levels.

Chapter 14 discusses the logistics of moving goods and maintaining a fleet of trucks to do so. The use of simulation in determining the number of trucks required to meet customer service level expections is shown. This is known as fleet sizing.

Chapter 15 discusses maintaining inventory at multiple locations in a complex supply chain. Rail movements between locations are described. Time varying expected demand for product is included. Building inventory in advance for periods when the expected demand exceeds production capacity is described. The use of customer services levels as the primary driver of supply chain logistics is discussed.

Chapter 13

Automated Inventory Management

13.1 Introduction

An inventory is a collection of parts or finished products that are waiting for use or shipment. Inventories increase costs by requiring storage space that could otherwise be productively used or simply not constructed. The cost of producing or purchasing what is stored cannot be recovered until the final product is delivered. Thus, minimizing inventories is important. On the other hand, not having a part or finished product when needed may lead to a stoppage of production or a dissatisfied customer who takes business elsewhere, thus decreasing revenue. Thus, having enough inventory is essential.

Lean demand management requires that manufacturers work with suppliers so that raw material or purchased parts are delivered precisely when needed. Manufacturers co-operate with large volume retailers to manage their inventories and ship product when sales records indicate that current inventory levels are low. Information gathered from scanning product bar codes at customer checkout can be aggregated and transferred electronically to the manufacturer nightly to enable this procedure.

This case study deals with an automated inventory system for a single product. The retail seller electronically collects information at the point of sale and transmits total daily sales to the supplier. The supplier must organize production and delivery to the seller such that the seller's inventory is not excessive and sales are not lost due to a lack of inventory.

13.2 Points Made in the Case Study

The automatic inventory system in this case study illustrates a fundamental consideration of demand and inventory management: the cost of holding inventory trades off with the need to meet customer demands.

Entities sometimes do not represent physical entities. In this case, a entity represents control information flowing within the inventory system.

A system can respond to changes in state variable values. The inventory system responds to changes in the amount of inventory on hand. When critical values are reached, state events, in the form of arriving entities, occur and initiate appropriate responses. The ability to model the dynamic response of a system to state variable values changes is a unique simulation capability.

A **Monte Carlo** simulation is usually defined as taking samples of one or more random variables, manipulating the samples, and gleaning information from the results in a situation where time plays no substantive role. This simulation experiment has these Monte Carlo characteristics. However, multiple points in time, each separated by one day, are considered. Changes in state variable values from day to day, determined by the random samples, are significant components of the simulation.

In previous case studies, detailed operations effecting individual entities are modeled. In this system, the aggregate affect of production and sales on inventory management are described. Statistical distributions are used to quantify this aggregate behavior. Manipulations of these distributions based on principles of probability and statistics assist in determining system and model parameter values.

The model used in this case study illustrates a simulation capability of fundamental importance. The model consists of three processes. Each process changes the values of the same state variables. The processes independently determine what actions to take based on the current

state variable values. However, no information is explicitly transmitted between the processes. The simulation engine transparently performs all co-ordination tasks.

13.3 The Case Study

A large manufacturer of office supplies (pens, pencils, tape, etc.) sells in large volume to a discount office supply retailer. In order to retain this customer, the manufacturer must manage the customer's inventory and automatically generate shipments of products when necessary. Missing any shipment due to a lack of available product results in a large financial penalty. However, management wishes to minimize the amount of inventory on hand to keep storage space costs and investment in unsold product low. In addition, it is in the best interest of the manufacturer if the retailer does not lose any sales due to a lack of product on hand. At the same time, the manufacturer cannot expect the retailer to keep excessive inventory.

13.3.1 Define the Issues and Solution Objective

We will consider only one product. Others can be assessed in a similar manner. Sales data supplied by the customer can be analyzed and the daily sales volume characterized by a statistical distribution. The sales data concerns one region with 37 stores. Thus, this data is a sum of 37 values. As was discussed in Chapter 5, the normal distribution may provide a good fit to such data. Using software for fitting data to distributions, it was found that a normal distribution with mean 180 cartons and standard deviation 30 cartons fits the actual daily regional sales data.

The manufacturer and the retailer have agreed that one shipment every three days on the average is acceptable. The distribution of three days sales can be determined using probability theory as follows. The distribution of the sum of three normally distributed random variables is also normally distributed with the mean equal to the sum of the three means and variance equal to the sum of the three variances. (Standard deviations don't add.) Thus, three days sales is normally distributed with mean 540 cartons and standard deviation 52 cartons. The 99% percent point of a normal distribution with mean 540 and standard deviation 52 is approximately 660. Thus, the amount of inventory needed to meet three days of sales with probability 99% is 660 cartons.

The reorder point, the inventory level that triggers a shipment from the manufacturer, must be set. Since shipments take one day, it is tempting to set the reorder point to the amount of inventory to meet one day's demand with probability of 99%, approximately 250 cartons. However, consider the consequences if the inventory at the end of a day is 300 cartons. No shipment is sent. The next day suppose the demand is 120 cartons leaving 180 cartons in inventory and triggering a shipment. The probability of the following day's demand exceeding 180 cartons is 50%. Thus, sales could be lost while the shipment is being processed.

The reorder point will be set at the amount of inventory to meet two days demand with probability of 99%. Two days demand is normally distributed with mean 360 and standard deviation 42. Thus the reorder point is set to be 460 cartons.

The nominal maximum production level for the product is 240 cartons per day. Actual data shows the production level to be uniformly distributed between 220 and 235 due to units that fail to pass inspection and random equipment failures.

There is no production on any day if inventory at the manufacturer is sufficiently high to meet the next shipment. A range for this inventory level, called the production cut-off point, can be computed as follows. The target number of units in inventory at the retailer after a shipment is received is 660. An order, which takes one day to receive, is placed when there are 460 cartons in inventory. The average number of sales in one day is 180 cartons. Thus, the average shipment has (660 - 460) + 180 = 380 cartons. The maximum shipment to the retailer is 660 cartons.

There are two important parameters of the inventory system:

1. The number of cartons in the inventory of the retailer, the reorder point, that triggers a new shipment from the manufacturer, currently proposed to be 460 cartons.

2. The number of cartons in inventory at the manufacturer that allows the following day's production to be canceled, the production cut-off point, currently proposed to be in the range 380 to 660 cartons.

Figure 13-1 summarizes the inventory system. Inventory is generated by production at the manufacturer and moved to the retailer as needed. Inventory levels, product movement, and production status are shown. Note again how this system is driven by dynamic decisions based on the values of state variables.

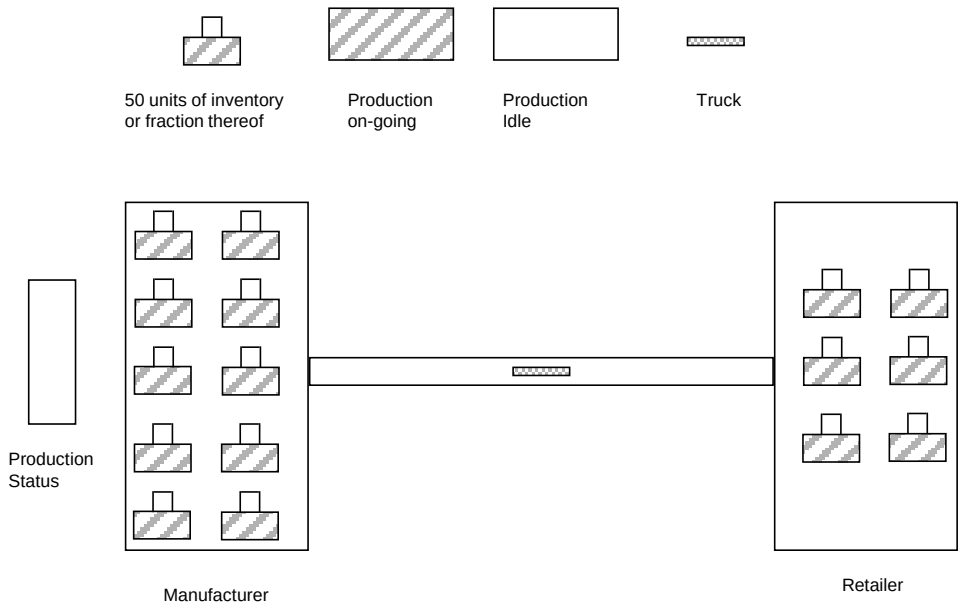


Figure 13-1: Automated Inventory Management System

13.3.2 Build Models

The model consists of three parallel processes:

1. Production at the manufacturer.
2. Sales at the retailer.
3. Shipments from the manufacturer to the retailer.

The first two processes schedule entity arrivals every day. The latter processes event triggered arrivals that occur when the retailers inventory drops below 460 cartons.
First the variables used throughout the model will be defined.

```
Define Variables
        ProductionInv           // Amount of inventory at the manufacturing plant
        RetailInv               // Amount of inventory at the retail plant
        Cutoff                  // Production cut off level
        DailyProd               // Daily production at the manufacturing facility
        Sold                    // Daily sales
        Demand                  // Daily demand
        Reorder                 // Reorder Point
        Ordered                 // Order volume from retailer
        Shipped                 // Number of units shipped from the manufacturing
```

Consider production at the manufacturer. A entity arrives once per day to control the production of new units. If the number of units in inventory is less than the production cut-off point, new units are made and added to the inventory. The model of this process follows.

```
Define Arrivals:
        Time of first arrival:      0
        Time between arrivals:  1 day
        Number of arrivals:     Infinite

Process Manufacture
Begin
        If ProductionInv > CutOff then
        Begin // No need for production today
                Tabulate 0 in Production
        End
        Else
        Begin // Produce today
                Tabulate  100 in T_Production
                Set       DailyProd = uniform 220, 235
                Increment ProductionInv by DailyProd
        end
End
```

Next consider the process for sales at the retailer. One entity representing sales information is created each day. The number of units demanded may exceed those available in inventory. This is an undesirable situation. The number of units demanded beyond those that are available in inventory represents lost sales. The model of the sales process follows.

Final consider the order process. A state event occurs when the number of units in the retail inventory becomes less than 460. The time till delivery is one day. The ordering process follows.

```
Define Arrivals:
        Time of first arrival:      0
        Time between arrivals:  1 day
        Number of arrivals:         Infinite

Process Sales
Begin
        Set Demand = normal 180, 30
        If  RetailInv  > Demand then
        Begin // Sufficient Inventory to meet demand
                Tabulate 100 in DailySales
                Set Sold = Demand
        End
        Else
        Begin  // Insufficient Inventory to meet demand
                Tabulate   0 in DailySales
                Set  Sold = RetailInv
        End

        Decrement RetailInv current by Sold
End
```

```
Define Arrivals:
        When RetailInv becomes less than Reorder
        Number of arrivals:         Infinite

Process Ship
Begin
        Set Ordered = min (660, 660 - RetailInv + 180)

        If   Ordered  > ProductionInv then
                Begin // Insufficient inventory for today's order
                Tabulate 0 in Shipments
                Set  Shipped = ProductionInv
                End

        Else
        Begin  // Sufficient Inventory
                Tabulate 100 in Shipments
                Set Shipped = Ordered
        End

         // Make shipment
        Decrement ProductionInv by Shipped
        Wait for 24 hr
        Increment RetailInv by Shipped
End
```

13.3.3 Identify Root Causes and Assess Initial Alternatives

Table 13-1 gives the experiment design for the inventory system simulation.

Table 13-1: Simulation Experiment Design for the Inventory System

Element of the Experiment	Values for This Experiment
Type of Experiment	Terminating
Model Parameters and Their Values	1. Re-order point for retailers inventory, 460 units 2. Production cancellation point based on manufacturer's inventory (380, 660) units
Performance Measures	1. Number of days with lost sales 2. Amount of inventory at the retailer 3. Number of days with no production 4. Amount of inventory at the manufacturer 5. Number of shipments 6. Number of shipments with insufficient units
Random Number Streams	1. Number of units manufactured 2. Number of units demanded
Initial Conditions	1. Inventory at the retailer – average of the reorder point and the maximum desirable inventory, 560 units 2. Inventory at the manufacturer -- Mid-point of the product cancellation range, 520 units
Number of Replicates	20
Simulation End Time	365 days (one year)

Management felt that demand data would be valid for no more than one year. Thus, a terminating experiment with a time period of one year was used. The number of units demanded each day and the number of units produced each day that production occurs are modeled as random variables. Thus, two random number streams are needed. Twenty replicates will be performed.

The initial inventory at the manufacturer and at the retailer must be set. Management believed that typical conditions are as follows. The number of units at the retailer should most often be between the re-order point and the intended maximum inventory level or 460 - 660. The average of these values, 560, will be used for the initial conditions. A typical number of units at the manufacturer should be within the range of the cut-off level for production, 380 - 660 units. The mid-point of the range, 520, is used.

As discussed previously, model parameters are the re-order point for the retailers inventory, whose value in the first experiment will be 460 units, and the production cancellation point. The values used for this latter quantity are the average shipment size, 380 units, and the maximum shipment size, 660 units.

There are several performance measures. The number of days with lost sales measures how well the inventory management system helps the retailer meet demand for the product. In addition, the inventory level at the retailer is of concern. At the manufacturer, the number of days without production and the inventory level are of interest. Finally, the total number of shipments from the manufacturer to the retailer, as well as the number of shipments with less than the requested number of units, should be estimated. The latter results from a lack of inventory at the manufacturer.

The inventory level at the retailer must be examined. Figures 13-2 and 13-3 show the inventory level at the retailer over time from the first replicate for each value of the cut-off point. In both graphs, the majority of the values are between 100 and 500 cartons.

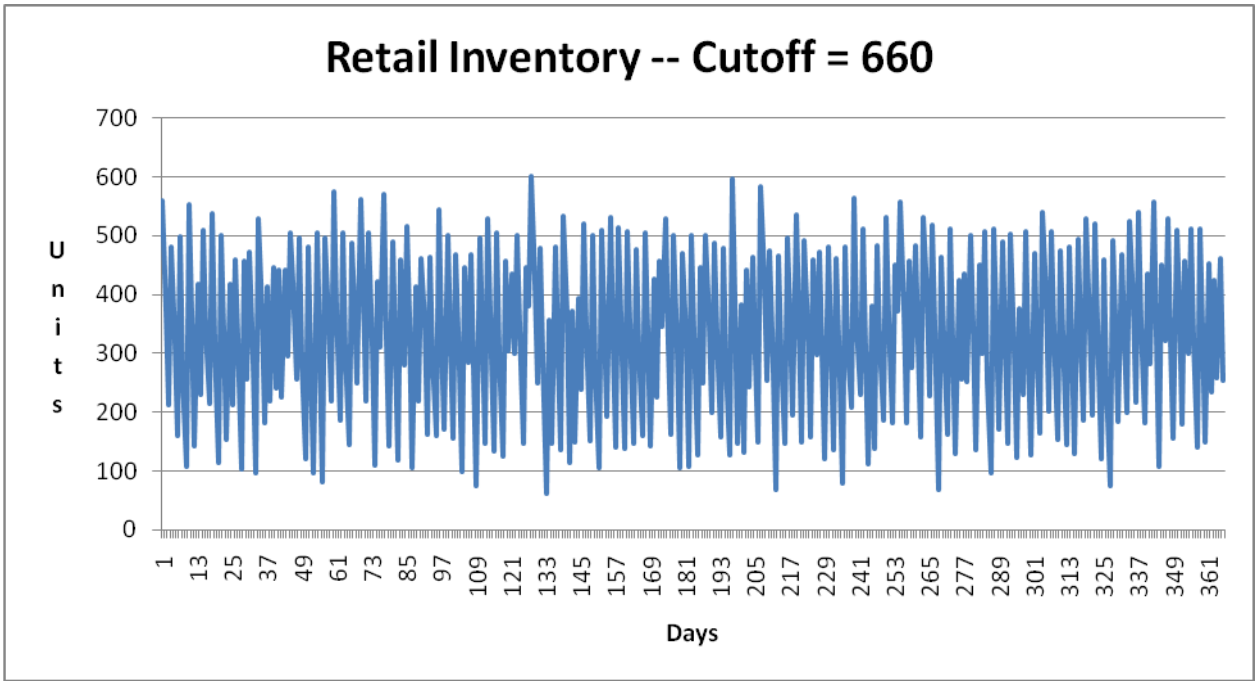


Figure 13-2: Inventory at Retailer – Production Cut-off Point of 660

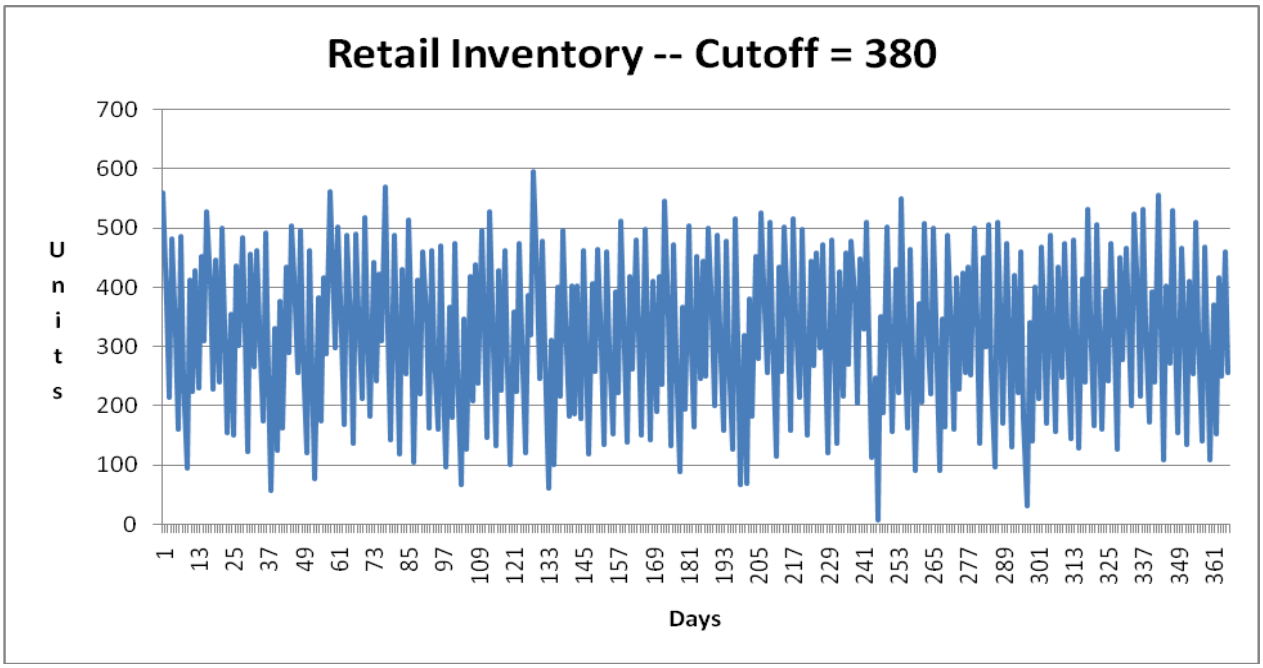


Figure 13-3: Inventory at Retailer – Production Cut-off Point of 380

Figures 13-4 and 13-5 show the inventory levels at the manufacturer for each value of the production cut-off point. The higher cut-off value, 660, results in an inventory between 400 and 800 cartons most of the time. The lower value, 380, results in an inventory between 200 and 500 cartons most of the time.

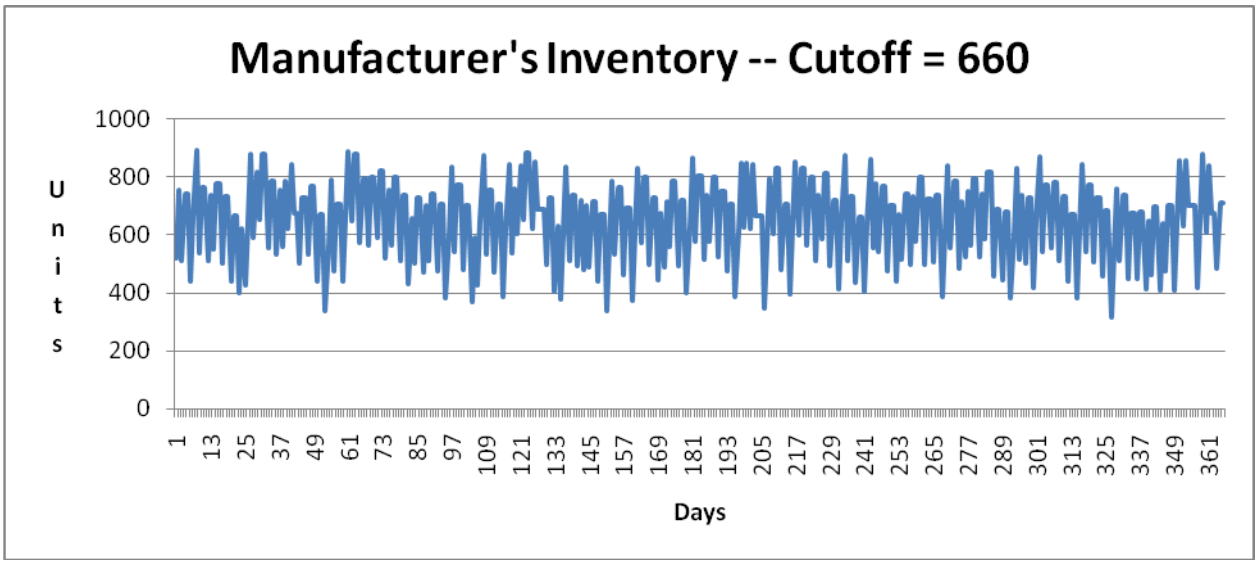


Figure 13-4: Inventory at Manufacturer – Production Cut-off Point of 660

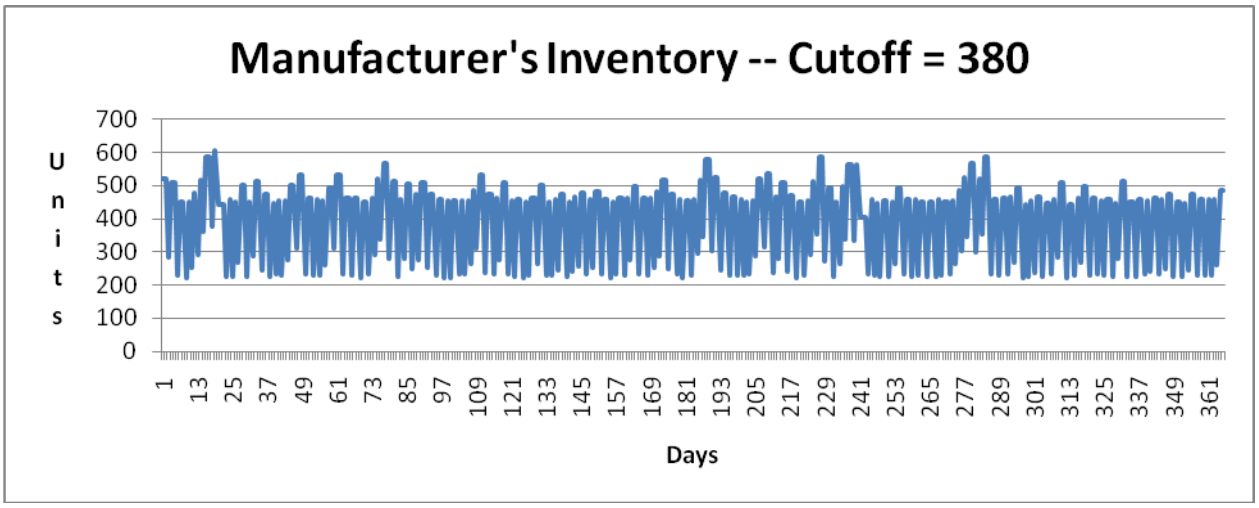


Figure 13-5: Inventory at Manufacturer – Production Cut-off Point of 380

There were only 4 days of lost sales over all 20 replicates when the cut-off point was 660 and 3 days when the cut-off point was 380. This indicates that the reorder point was set correctly, or at least not too low.

Table 13-2 summarizes the other performance measure values resulting from the simulation experiment.

The number of days with no production is approximately the same for both values of the production cut-off point. The number of days per year without production is about 78 or 1.5 days per week. Thus, 5.5 days of production per week should be sufficient.

When the higher production cut-off point is used, all shipments contain the number of units requested by the retailer. When the lower cut-off point is used, slightly less than half of the shipments have an insufficient number of units, that is fewer units than requested by the retailer. The lower cut-off value leads to an average of 10.4 additional shipments per year.

Table 13-2: Results of the Inventory System Experiment

	Days with No Production			# of Insufficient Shipments			Total # of Shipments		
Replicate	Cut-off Point 660	Cut-off Point 380	Difference	Cut-off Point 660	Cut-off Point 380	Difference	Cut-off Point 660	Cut-off Point 380	Difference
1	80	81	1	0	73	73	133	143	10
2	75	76	1	0	66	66	137	147	10
3	74	75	1	0	76	76	136	146	10
4	79	80	1	0	64	64	135	145	10
5	79	80	1	0	79	79	132	144	12
6	73	74	1	0	78	78	136	146	10
7	76	77	1	0	65	65	133	145	12
8	76	77	1	0	57	57	136	146	10
9	76	77	1	0	60	60	135	146	11
10	78	79	1	0	79	79	134	144	10
11	75	77	2	0	68	68	134	147	13
12	72	74	2	0	60	60	138	148	10
13	74	75	1	0	55	55	135	147	12
14	78	79	1	0	67	67	136	145	9
15	72	74	2	0	74	74	137	148	11
16	74	75	1	0	71	71	138	146	8
17	78	79	1	0	73	73	134	146	12
18	81	82	1	0	72	72	135	144	9
19	72	73	1	0	61	61	137	148	11
20	77	78	1	0	76	76	137	145	8
Average	75.9	77.1	1.2	0	68.7	68.7	135.4	145.8	10.4
Std. Dev.	2.7	2.6	0.4	0	7.5	7.5	1.7	1.4	1.4
99% CI Lower Bound	74.2	75.4	0.9	0	63.9	63.9	134.3	144.9	9.5
99% CI Upper Bound	77.7	78.8	1.4	0	73.5	73.5	136.5	146.7	11.3

13.3.4 Review and Extend Previous Work

Management felt that the foremost objective is to satisfy the retailer. The automated inventory management system appears to meet the objective. The retailer is able to meet all customer demand without carrying excessive inventory. Most of the time, the inventory is less than three days average demand for the product.

The higher value for the cut-off point is preferred. This results in no shipments with less units than the retailer demanded as well as fewer total shipments. Management is willing to accept a larger inventory at the manufacturer to better satisfy the customer.

Management noted that the variance between replicates is small. This small variance should make the automated inventory system easier to operate and control.

Production of the product 5.5 days a week will be scheduled.

13.3.5 Implement the Selected Solution and Evaluate

The automated inventory system will be installed and will operate with a reorder point of 460 units and a production cut-off point of 660 units. The retailer was assured by the simulation results of the ability to meet customer demand completely and consistently as well as holding a relatively low inventory. System performance will be monitored using the measures defined for the simulation experiment.

13.4 Summary

This chapter illustrates how dynamic decision making based on state variables may be incorporated into models. System details are not modeled directly. Aggregate behavior is modeled using statistical distributions. Graphs show the dynamics of inventory levels. System behavior due to alternative values of inventory system parameters is assessed.

Problems

1. Provide verification evidence for the inventory system experiment based on the following results.

Number of days processed:	365
Number of days without production:	77
Number of days with production:	288
Number of shipments:	145
Number of shipments of sufficient quantity:	78
Number of shipments of insufficient quantity:	67

2. Provide validation evidence for the inventory system experiment based on the simulation results presented in this case study.

3. One possibility that could arise during the simulation experiment is the following. Suppose a shipment of insufficient units failed to bring the inventory at the retailer above the reorder point. What would be the consequences for the simulation experiment? What conclusions could be drawn about operating the system with the particular reorder and cut-off point values?

4. Discuss management's decision to use the higher cut-off point value. Defend using the lower value since the customer can still meet all demand.

5. Include detection and response of the condition described in problem 3 in the model and resimulate the using the lower value of the cut-off point.

6. Find a cut-off point value between 380 and 660 that improves system operation. Use a reorder point of 460 units.

7. Find a reorder point lower than 460 units that either improves system operation or makes it no worse.

8. Print out a trace of the inventory at the retailer. For each day, include the inventory at the start of the day, the deliveries, the demand, and the inventory at the end of the day.

9. Modify the model so that product occurs Monday through Friday at the current rate and Saturday at half the current rate. This implements the conclusion of the analysis that 5.5

days per week production on the average is sufficient. At the same time, customer demand at the retailer occurs 7 days per week. Would you expect more or less inventory to be needed at the retailer? Defend your expectation.

Case Problem

A product inventory changes daily due to customer demand that withdraws from it and production that replenishes it on the days when it is operating. Customer demands and production are random variables. Production is subject to down times of random frequency and duration. Customers due not backorder. Thus, any customer demand that cannot be met results in a lost sale.

Periodically, an analyst can review the inventory and make adjustments by purchasing or selling product on the spot market. The time between these reviews is called the review period. At each review, the analyst can do the following:

a. If the current inventory is less than the safety stock, buy a quantity of product equal to (safety stock – current inventory) on the spot market.
b. If the current inventory is greater than the maximum inventory, sell a quantity of product equal to (current inventory – maximum inventory) on the spot market.

The safety stock level is an operating parameter of the inventory system set such that the probability of meeting all customer demand between periodic reviews is at least a specified value, typically 90%, 95%, or 99%. This probability is called the effective service level.

The maximum inventory level is less that the physical limit on inventory storage, called the capacity, to avoid having no place to store items.

Figure 13-6 summarizes the inventory control system. Note the following definitions and notation.

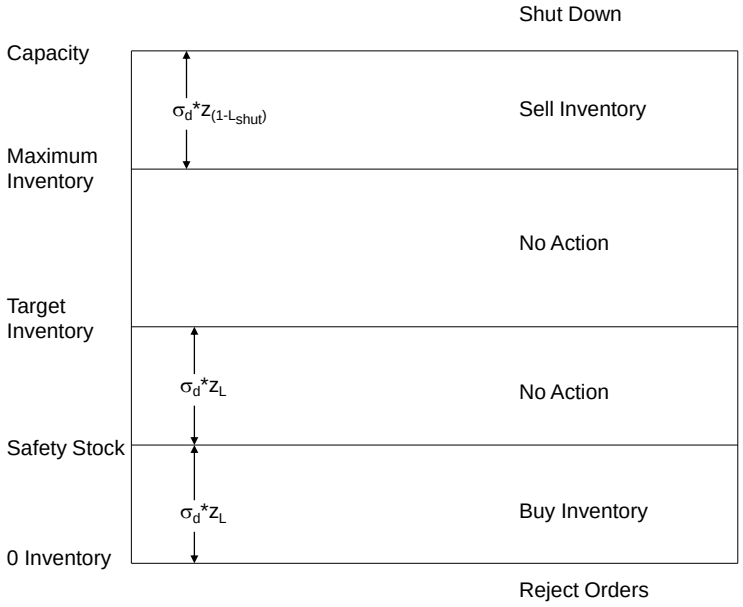


Figure 13-6: Inventory Management System Summary

Target inventory – The desired inventory to avoid purchases and sales on the spot market, in the range [safety stock, maximum inventory].

Delta inventory – The change in inventory each day due to production (+) and demand (-).

Nominal service level – The probability that the inventory will be greater than the safety stock at the time of a review given that it was equal to the target at the time of the last review. As well, the probability that the inventory will be greater than zero at the time of a review given that it was equal to the safety stock at the time of the last review.

d = number of days in the review period

μ_d = the mean of the delta inventory distribution for a review period of d days

σ_d^2 = the variance of the delta inventory distribution for a review period of d days

μ_p = the mean of the conditional daily production distribution (given that production is greater than zero).

σ_p^2 = the variance of the daily production distribution

μ_c = the mean of the daily customer demand distribution

σ_c^2 = the variance of the daily customer demand

A = the percent of days that production occurs (the availability of production)

L = the nominal service level as a percent

L_{eff} = the effective service level

$$L = 1 - \sqrt{(1 - L_{eff})}$$

L_{shut} = the probability of exceeding the capacity during the next review period given that the inventory level is equal to the maximum inventory at the current review

z_L = The L% point of the standard normal distribution

The average demand is equal to the average production over a long period of time such as a year. This means that $\mu_c = \mu_p * A$. In other words, the expected production each day (including an allowance for down time) is equal to the expected daily demand.

Ignoring down times, the delta inventory can be viewed approximately normally distributed with parameters:

$$\mu_d = 0$$

$$\sigma_d^2 = \sum_{i=0}^{d} \left[\sigma_p^2 + \sigma_c^2\right] = d * \left[\sigma_p^2 + \sigma_c^2\right]$$

The problem is to determine the review period, safety stock, maximum inventory, and target inventory given an effective service level. Performance measures should include the service level as well as the number of purchases and sales made to the spot market by the analyst.

Consider the following specifications.

Capacity:	10,000.
Distribution of customer demand:	Normal (1000, 250).
Distribution of production when production occurs:	Normal(1000/A, 300).
A=Availability:	30/(30+1.92)
Review period:	7, 10, or 14 days.
Effective service level:	0.95 or 0.99.

First, determine the required quantities assuming that there is no production downtime. Use analytic models to compute the safety stock, target inventory, and maximum inventory for each review period. Use simulation to validate the analytic computations. Make adjustments to the analytically computed quantities if necessary. Validate your final recommendation using simulation. Validate means to show that the required effective service level is met.

Next, consider production downtime. Establish and validate quantities for the safety stock, target inventory, and maximum inventory using simulation. The average time between periods of no production is 30 days. The average length of each period of no production is 1.92 days. The distribution of the period of no production follows.

Distribution of the Length of Periods of No Production

Days Down	Percent
1	50%
2	25%
3	13%
4	7%
5	5%
Total	100%

Case Problem Issues

1. Identify the processes that are needed in the model.

2. Specify all of the combination of the values of the parameters that should be simulated.

3. Compute the safety stock, target inventory level, and maximum inventory level for each parameter value combination using a spreadsheet.

4. Specify the initial conditions for the simulation.

5. In addition to service level, define the performance measures.

6. Determine how production downtime should be modeled.

7. Discuss how verification and validation evidence will be obtained.

The time period of interest is one year.

Chapter 14

Logistics

14.1 Introduction

Logistics has to do with the procurement, storage, transportation, and delivery of goods or people. A logistics system deals with the way finished products move from producer to customer or raw material moves from a supplier to the producer. Logistics systems must be responsive to customer requirements for short lead times. That is the amount of time between the placement of an order and its delivery should be minimized. Using excessive amounts of inventory or capital equipment to accomplish this objective increases costs and thus is inconsistent with lean principals.

Movement of goods may involve truck, rail, water, and air transportation. Facilities for loading and receiving products by each mode of transporation employed are necessary. Evaluating trade-offs between using various transporation modes can be a part of a simulation study.

Inspection and repair of transportation equipment is important. Inspection is often required after each round trip to a customer and returning to the shipping site. Inspection delays and subsequent repair times if necessary must be included in a simulation model.

Determining how many trucks, rail cars, or aircraft are needed must be accomplished. This is known as fleet sizing. Fleet size estimates are often made using simple algebra based on the expected round trip time to a customer, including inspection and repair as well as the number of round trips needed per planning period. This result is the lower bound on the fleet size. Simulation allows the effects of variability on meeting customer requirements for timely deliveries to be considered when sizing a fleet so that a more precise esitimate is obtained. Variability sources include transportation times and customer demands.

Staffing plans for logistics systems are necessary. A lack of staff may prove to be a constraint on the number of loads shipped. Staff may work only certain shifts during the day and only certain days of the week. Such scheduling may result in structural variability that causes the need for additional inventory or capital equipment.

Logistics systems add no value to products. Thus, their cost needs to be minimized. On the other hand, they are critical to making sure customers are satisfied by receiving products on time.

A simple logistic system is shown in Figure 14-1. A factory creates product which is stored in an inventory. Customer demands result in shipments via truck to the customer site. After the shipment is unloaded into the customer's inventory, the truck returns to the factory for inspection and repair as well as to await its next shipment.

This chapter discusses a basic and straightforward logistic system with an emphasis on fleet sizing. A more complex logistic system as required in a supply chain is discussed in the next chapter.

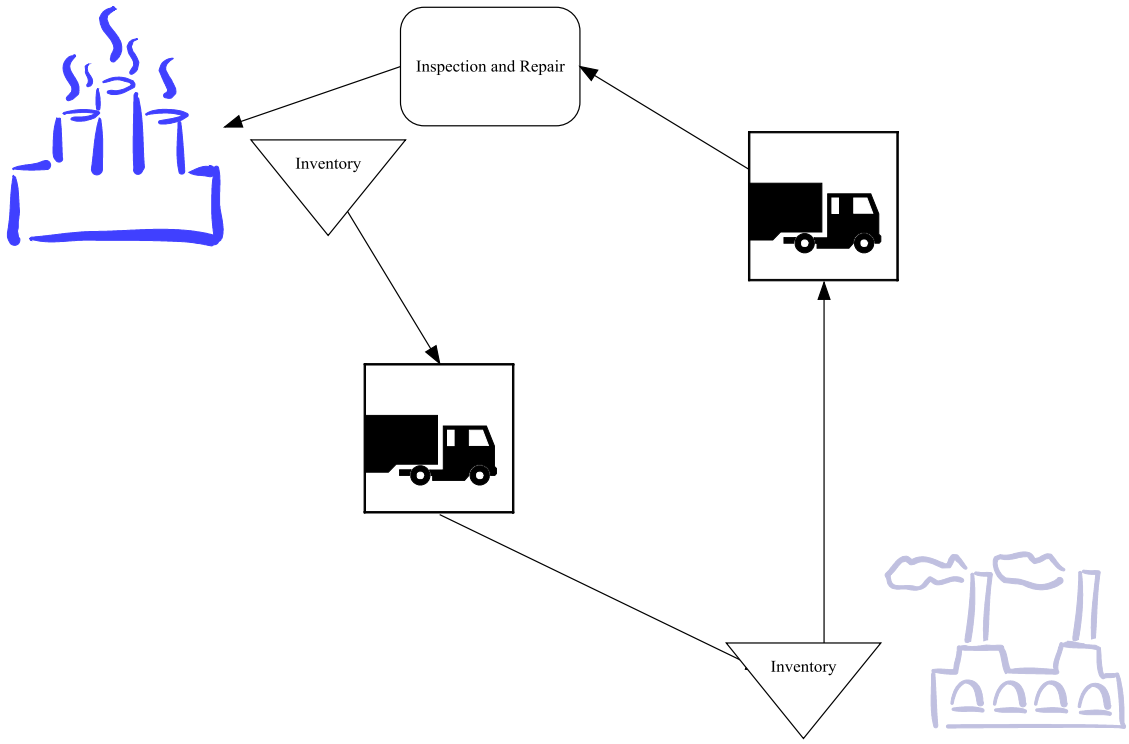


Figure 14-1: Simple Logistics System

14.2 Points Made in the Case Study

Arrivals represent daily shipping demand. There is one arrival per day at the beginning of the day. The number of shipments per day is modeled as a random variable.

Trucks used in shipping are modeled with a single resource. Each unit of the resource represents a truck. The number of trucks (resource units) required can be determined dynamically in the model during execution. An additional truck (resource unit) is created whenever needed to make a shipment, subject to an upper limit. Thus, the number of trucks needed to meet a performance level can be determined.

A sequence of experiments can be used to determine the relationship between the maximum number of trucks (resource units) available and system performance. The values of the number of trucks depend on the results of previous simulation experiments. After the number of trucks is established, the number of workers is set in the same way.

Simulation experiments can be run to determine the affect of structural variability on system performance. Various staff schedules with regard to shifts worked per day and days worked per week can be tested to determine their relationship to system performance and to estimate the number of workers required. This is left as an exercise for the reader.

14.3 The Case Study

Resource requirements in a logistics system include capital expenditures for transportation mechanisms such as trucks and operating costs such as personnel salaries. Minimizing capital and operating expenditures while providing the level of service demanded by customers is a fundamental issue.

14.3.1 Define the Issues and Solution Objective

A new logistics system is being designed to deliver truck loads of finished product over a large area from a main terminal supporting a manufacturing plant. The logistics system works in a similar way to the one shown in Figure 14-1. Truck loads are shipped seven days per week every day of the year. For the next year, the daily shipping volume is estimated as follows: a minimum of 20, a mode of 35 and a maximum of 65. Thus, the average daily shipping volume is 40 loads per day $\left(= \frac{(20 + 35 + 65)}{3} \right)$. This means that there are 14600 loads shipped per year on the average.

A truck waits at the terminal until it is loaded. Loading time is uniformly distributed between 2 and 4 hours. A sufficient number of workers are available for loading. The truck will make all of its deliveries and then return to the terminal. The time from the terminal to the customer site, in either direction, is triangularly distributed with a minimum of 4 hours, a mode of 12 hours and a maximum of 30 hours. The time at the customer site is triangularly distributed with a minumum of 2 hours, a mode of 4 hours, and a maximum of 8 hours.

Upon its return to the terminal, the truck must be inspected by a worker. Inspection time is uniformly distributed between 1 and 2 hours. Approximately 90% of the trucks pass inspection or require only minor adjustments and are then ready for another load. The other 10% require significant maintenance that is performed by the same worker. Repair time is triangularly distributed with a minimum of 4 hours, a mode of 8 hours and a maximum of 12 hours. Workers are available 16 hours per day.

Management does not want to significantly constrain the number of loads delivered each year. At the same time the number of trucks and number of workers needs to be minimized for cost reasons. Management has determined that differences of more than 1% in the number of round trips completed are operationally significant. This difference can affect company profitability. Difference of less than 1%, even if statistically significant, are considered to be operationally unimportant.

The objective is to determine the number of trucks and workers required for the effective operation of the product delivery system. Effective operation requires minimizing costs without significantly reducing the number of loads delivered.

14.3.2 Build Models

The expected number of trucks and workers needed can be estimated using simple algebra. This is a lower bound on the actual number of trucks and workers needed Simulation is used to determine if additional trucks and workers are needed to meet the delivery criteria established by management.

The expected time for a truck to complete the delivery process and be ready to begin another delivery is shown in Table 14-1.

Table 14-1: Expected Time to Complete Delivery Process

	Expected Time (Hours)
Load Truck	3.0
Travel to Customer	15.3
At Customer	4.7
Travel to Terminal	15.3
Inspection	1.5
Repair	0.8
Total	40.6

There are approximately 8760 hours in a year. Thus, a truck could be expected to make 215 deliveries per year (= 8760 / 40.6). Thus, the number of trucks required to make 14600 deliveries is 68 (= 14600 / 215).

The expected number of workers needed can be determined in the same way. A worker is required for inspection and repair with an expected time of 2.3 hours per delivery. Thus, a single worker who works 16 hours per day could inspect and repair on the average of 2539 trucks per year. Thus, 6 workers on each shift (= 14600 / 2539) are needed.

The model of the logistics system can be divided into the following processes.

1. Daily generation of loads.
2. Truck loading and round trip to the customer site.
3. Truck inspection and repair.
4. Worker shift changes.

The first process Daily Loads operates as follows. The number of loads per day is generated as a sample from a triangular distribution with the appropriate minimum, mode, and maximum: 20, 35, 65. While the daily number of loads is an integer, it is also sufficiently large to model as a continuous random variable.

In order to avoid “loosing” fractional loads, the fractional part of the sample for one day is added to the number of loads for the next day. For example, suppose the value for the number of loads is 30.6. Then, 30 loads are created today and 0.6 is added to the number of loads for the following day. Suppose the value for the number of loads on the following day is 40.7. Next, 0.6 is added. Thus, 41 loads are created and 0.3 is added to the number of loads for the next day.

In this process, the variable LoadsWaiting contains the quantity described above and the variable NLoads contains the integer portion of LoadsWaiting. NLoads loads are sent to the second process, RoundTrip, each day.

```
Define Arrivals:
        Time of first arrival:      0
        Time between arrivals:  1 day
        Number of arrivals:         Infinite

Define Variables
        LoadsWaiting            // Number of loads to ship
        NLoads: Integer         // Integer number of loads to ship

Process Daily
Begin
        Set LoadsWaiting += Triag 20, 35, 65
        Set NLoads = LoadsWaiting
        Set LoadsWaiting -= NLoads
        Clone NLoads to RoundTrip
End
```

The process model for truck loading and load delivery, RoundTrip, consists of four time delays, one each for truck loading, movement to the customer site, time at the customer site, and return to the terminal. Then the truck goes to the inspection process. Preceding the time delays, each load acquires a truck.

Preceding truck acquisition, the model determines whether an additional unit of the truck resource should be created. If the number of truck resource units is less than a specified limit, contained in the variable MaxTrucks, and there are no free units of the truck resource, then a new unit is created. Thus, the load would not wait for a truck. Else, the load must wait for a truck to return from a delivery as well as being inspected and repaired.

```
Define Variables
        MaxTrucks               // Maximun number of trucks

Define Resources
        Truck                   // Trucks

Process RoundTrip
Begin
        If Truck/1 is Idle is FALSE then
        Begin
        // Add another truck if possible
                If Truck Units < MaxTruck then
                        Increment Truck Units by 1
        End
        Wait Until Truck/1 is Idle in QTruck
        Make Truck/1 Busy
        Wait for        uniform         2,  4 hours             //Loading Time
        Wait for        triangular      4, 12, 30 hours  //To Customer
        Wait for        triangular      2,  4,  8 hours    //At Customer
        Wait for        triangular      4, 12, 30 hours  //From Customer
        Send to Inspect
End
```

Similar logic is used to model the worker resource in the truck inspection and repair process, Inspect. After an entity acquires the worker resource, there is time delay for inspection. Ten percent of the entities also require a time delay for repair.

```
Define Variables
        MaxWorkers              // Maximun number of workers

Define Resources
        Worker                  // Inspection and repair workers

Process RoundTrip
Begin
        If Worker/1 is Idle is FALSE then
        Begin
        // Add another worker if possible
                If Worker Units < MaxWorker then
                        Increment Worker Units by 1
        End
        Wait Until Worker/1 is Idle in QWorker
        Make Worker/1 Busy
        Wait for uniform 3,  1 hr                   //Inspection
        If uniform 0, 1 < 10% then
                Wait for triangular 4, 8, 12        //Repair
        Make Worker/1 Idle
        Make Truck/1   Idle
End
```

The worker shift process is as was discussed in chapter 2. All units of the worker resource are put into the off-shift state after 16 hours of work and returned to the idle state after 8 hours.

Notice that this results in an approximation in the model. If a worker is inspecting or repairing a truck at the beginning of the off shift period, the inspection or repair will continue until completed. This worker will again be available for work at the beginning of the on-shift period. Thus, the number of workers required could be underestimated.

14.3.3 Identify Root Causes and Assess Initial Alternatives

The experimental strategy to determine the number of trucks and workers is as follows. First the number of trucks will be determined. After the number of trucks is established, the number of workers will be determined for that number of trucks.

The minimum number of trucks is the expected number, 68, as determined in the previous section. The maximum number will be determined through simulation by not constraining the number of units of the truck resource used, that is requiring that no load ever waits for a truck. Various values of the number of trucks between the minimum and the maximum will be simulated.

The number of workers is not constrained so that no returning truck waits for a worker.

For each of these values, 20 replicates will be made and the average number of completed round trips over the replicates computed. A graph showing the average number of completed trips versus the number of trucks can be constructed. Thus, the number of trucks to use is determined.

The experimental design for determining the number of trucks is shown in Table 14-2.

A terminating experiment of duration 1 year (8760 hours), the planning period for the logistics system is used. There is one random number stream for each time delay modeled as a random variable as well as a random number stream for determining the daily number of loads. An

additional random number stream is needed to model the random choice as to whether a truck passes inspection.

The primary system performance measure is the number of round trips completed. The utilization of trucks and workers are of interest. The experiment will begin with all trucks at the terminal waiting to make a trip.

Table 14-2: Simulation Experiment Design for Determining the Number of Trucks

Element of the Experiment	Values for This Experiment
Type of Experiment	Terminating
Model Parameters and Their Values	1. Number of trucks – Various values between the minimum needed and the maximum used.
Performance Measures	1. Number of round trips completed 2. Utilization of trucks 3. Utilization of workers
Random Number Streams	1. Number of pallets each day 2. Truck loading time 3. Travel time to customer 4. Time at customer site 5. Travel time from customer to terminal 6. Time to inspect returning truck 7. Decision: Did truck pass inspection? 8. Time to repair truck
Initial Conditions	Empty buffers and idle resources
Number of Replicates	20
Simulation End Time	1 year

First the simulation is run with 68 trucks and then with maximum number of trucks used as determined by the simulation to be 144 with an approximate 99% confidence interval of (141.3, 146.3).

The maximum number of trucks case results in an average of 662 more round trip completions per year. This is an increase of 4.8% over case where 68 trucks are used. Furthermore, this difference is statistically significant as seen by the approximate 99% confidence interval for the difference.

Thus, it can be concluded that more than 68 trucks are needed and that the number of trucks needed is between 68 and 144.

Table 14-3: Number of Round Trips – Average Trucks and Maximum Trucks

Replicate	Maximum Trucks	Average Trucks	Difference
1	14738	13863	875
2	14374	13850	524
3	14697	13839	858
4	14564	13800	764
5	14345	13853	492
6	14527	13823	704
7	14278	13804	474
8	14421	13877	544
9	14638	13825	813
10	14357	13854	503
11	14611	13858	753
12	14791	13828	963
13	14507	13755	752
14	14477	13754	723
15	14442	13872	570
16	14447	13868	579
17	14397	13810	587
18	14308	13839	469
19	14501	13820	681
20	14469	13860	609
Average	14494.5	13832.6	661.9
Std. Dev.	142.7	35.0	147.5
99% CI Lower Bound	14403.2	13810.2	567.5
99% CI Upper Bound	14585.7	13855.0	756.2

Furthermore simulation experiments are run with 68, 70, 75, …, 140, and 144 trucks. Results are shown in the following graph, Figure 14-3.

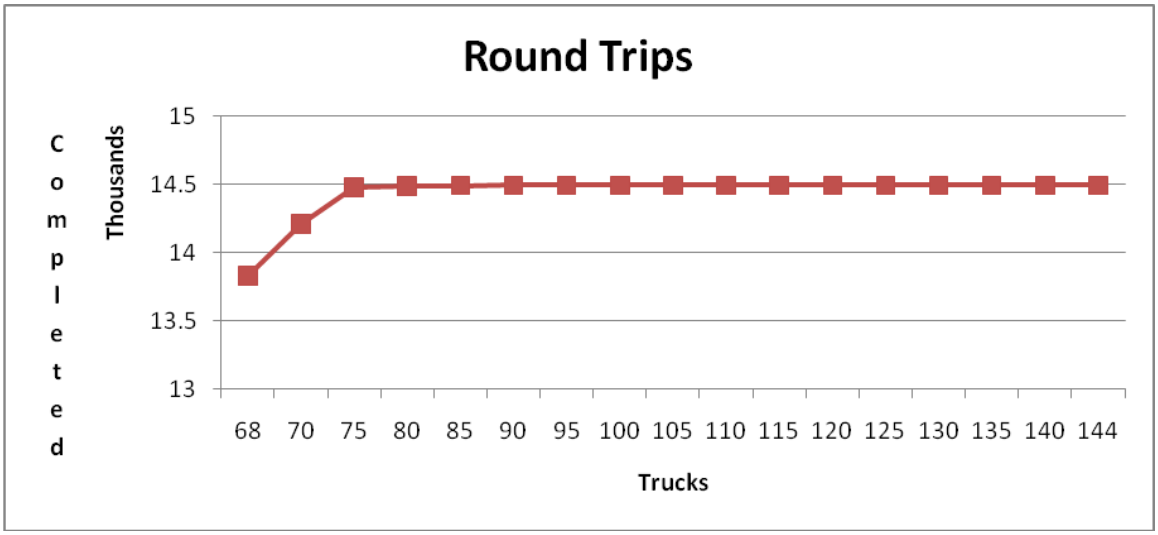


Figure 14-3: Round Trips versus Number of Trucks

It appears from the graph that the number of roundtrips increases significantly up to 75 trucks. The difference in round trips when 80 trucks are used instead of 75 is only 14 on the average and thus is not operationally significant. Thus, 75 trucks will be used if there is a statistically and operationally significant difference in the number of roundtrips versus when 70 trucks are used. The simulation results are summarized in Table 14-4.

Table 14-4: Number of Round Trips – 75 Trucks versus 70 Trucks

Replicate	75 Trucks	70 Trucks	Difference (75 vs 70 Trucks)
1	14268	14715	447
2	14197	14351	154
3	14238	14669	431
4	14198	14558	360
5	14235	14341	106
6	14222	14527	305
7	14104	14273	169
8	14258	14385	127
9	14208	14601	393
10	14169	14320	151
11	14226	14604	378
12	14212	14704	492
13	14123	14497	374
14	14174	14477	303
15	14276	14433	157
16	14262	14439	177
17	14189	14397	208
18	14231	14307	76
19	14191	14486	295
20	14239	14435	196
Average	14211.0	14476.0	265.0
Std. Dev.	45.1	132.9	127.4
99% CI Lower Bound	14182.1	14390.9	183.5
99% CI Upper Bound	14239.9	14561.0	346.4

On the average, the number of roundtrips increases by 265, 1.9%, when 75 trucks are used versus 70 trucks. Thus, the difference is operationally significant since it is greater than 1%. The approximate 99% confidence interval for the difference is (183.5, 346.4). Thus, the difference is statistically significant.

Thus 75 trucks should be used. For this case, the truck utilization is 94.9% with an approximate 99% confidence interval of (94.1%, 95.3%). Utilization includes time spent in inspection and repair.

Given that 75 trucks should be used, the number of workers must be determined. The average maximum number of workers determined by the simulation experiments using 75 trucks is 30. This is the maximum number of workers that could be needed. The average number of workers computed with algebra was 6. The actual number of workers needed is somewhere between these two values. The simulation experiment to determine the number of workers is the same as that shown in Table 14-2 except that the model parameter is the number of workers instead of the number of trucks. Conducting this experiment is left as an exercise for the reader.

14.3.4 Review and Extend Previous Work

Management was pleased with the results as presented above and 75 trucks will be acquired.

14.3.5 Implement the Selected Solution and Evaluate

The number of completed roundtrips will be monitored. Additional trucks can be obtained and workers can be hired if needed.

14.4 Summary

This case study emphasizes a sequentially designed simulation experiment to determine the level of resources needed to operate a truck based logistics system. Minimizing the cost of the system in terms of trucks and workers trades off with the need to meet delivery targets. The idea of a level of indifference is employed. Alternatives may statistically differ significantly, but the difference may not be large enough, greater than the level of indifference, to impact system operations.

Problems

1. Validate the computation of the expected time a truck spends in repair per roundtrip.

2. Tell what the entity in each of the processes in the model discussed in this chapter represents.

3. Give verification evidence based on the information resulting from one replicate of the simulation experiment as follows:

Number of truck round trips started:	14335
Number of truck round trips completed:	14203
Number of truck round trips on going at the end of the simulation:	104
Number of trucks waiting or in inspection and repair at the end of the simulation:	28

4. Compare the modeling and experimental issues of the logistics system discussed in this chapter to those concerning the serial line discussed in chapter 7.

5. Tour the operation of the local office of an overnight delivery service. Write down a process model of their logistics system for organizing and delivering in bound packages.

6. Modify the model presented in this chapter so that no worker inspects or repairs a truck during the off-shift period. Assess the effect of making the model more precise on the number of workers required.

7. Suppose that workers were available 24 hours per day but the total number of hours worked per day could not increase. That is, there would be 2/3rds of the number of workers determine above would work each shift. Use the model developed in this chapter to determine if the number of trucks needed could be lowered.

8. Modify the model and simulation experiment to estimate the needed capacity of the parking area for trucks at the terminal. Include trucks that are in inspection or repair.

9. Modify the model and simulation experiment to give a profile of truck location. Estimate the average number of trucks in each possible location: in route to the customer, at the customer, in route to the terminal, in inspection, in repair, and waiting for a load.

10. Conduct a simulation experiment to determine the number of workers needed.

Case Study

A new logistics system is being designed to transport one product from a factory to a terminal by rail. A simulation study is needed to estimate the following:

1. The rail fleet size.
2. The size of the rail yard at the factory.
3. The size of the rail yard at the terminal.
4. The size of the inventory needed at the terminal.

Customer demand is satisfied each day from the terminal. Demand is normally distributed with a mean of 1000 units and a standard deviation of 200 units. Production at the factory is sufficient to meet demand on a daily basis. Policy is to ship an average of 1000 units each day from the factory to the terminal. Each rail car holds 150 units. Partial rail car loads are not shipped but included with the demand for the the next day.

The customer service level provided at the terminal should be at least 99%. The time period of interest is one year.

Transportation time from the factory to the terminal is triangularly distributed with a minimum of 3 days, a mode of 7 days, and a maximum of 14 days. At the terminal, a car must wait for a single unload point to unload. Unloading takes one hour. Upon return to the plant, a rail car is inspected. Inspections take 2 hours. Maintence is required for 3% of returning cars. Maintenance requires 4 days.

Embellishment: All cars leaving the factory in a day join a single train leaving at 4:00 A.M. the next morning and have the same transportation time to the terminal. A single train containing all empty cars leaves the terminal at 4:00 A.M. each morning.

Case Study Issues.

1. What initial conditions concerning the arrival of trains to the terminal should be used?

2. What target inventory level should be used?

3. How is the policy to ship 1000 units each day from the factory implemented if rail cars hold 150 units?

4. Embellishment: How is the requirement that all rail cars leaving the factory or terminal join a single train with a single transporation time to the other site modeled?

5. How is the unloading constraint for rail cars modeled?

6. In what order should the system parameters listed above be determined by the simulation experiments?

7. What is the primary performance measure for the simulation experiments?

8. How will verification and validation evidence be obtained?

9. How is the size of a rail yard modeled?

10. How is the size of the rail fleet modeled?

11. Computed the expected fleet size and use the result in providing validation evidence.

12. Define the processes that comprise the model.

Chapter 15

Integrated Supply Chains

15.1 Introduction

A supply chain integrates the efforts of geographically dispersed production and distribution facilities that acquire raw material, make intermediate or finished products, and deliver finished products to customers. Transportation links provide for product and raw material movement between facilities. Integration is accomplished by information technology that shares production, inventory, and customer demand data among the facilities.

Integration implies that the operation of each facility affects the operation of all other facilities. The volume of production at a facility is determined by the need for its products at subsequent facilities in the supply chain. The fundamental purpose of the supply chain is to meet customer demand for finished products. Thus, customer demand drives all of the work of the supply chain.

A simple two facility supply chain is shown in Figure 15-1. At the right side of the figure, customer demand is satisfied from finished goods inventory at facility B. Facility B production levels are set so that the finished goods inventory is replenished. Facility B production requires an intermediate product made by facility A that is stored in an inventory at facility B. Facility A ships the intermediate product to facility B so that just enough inventory is available to meet production requirements at facility B. Shipments are made from an inventory at facility A that is replenished by production at facility A. Thus, customer demand indirectly drives production at facility A. Facility A needs to be constantly knowledgable about customer demand, production levels, and inventory levels at facility B to set its own production levels.

Many supply chains are much more complicated than the one shown in Figure 15-1. There are multiple kinds of facilities: some for production only and some for movement or transfer of materials like the facilities that will be discussed in a later chapter. More than one finished product may be delivered to customers. Facilities may supply products to and receive many products from many other facilities. More than one mode of transporation may be involved. The expected demand of a customer for a product may vary over time, that is be subject to seasonal variations.

Modeling an integrated supply chain involves modeling the flow of information from the end of the supply chain where product is delivered to customers to the beginning of the supply chain where the first intermediate product is produced from raw materials. The flow of product between facilities must be modeled as well as inventory management and production. Understanding and modeling of customer demand is essential.

This case study shows the simulation approach to evaluating integrated supply chain performance and how all aspects of a supply chain are integrated into one model. It is based in part on the work described in Standridge and Heltne [2000].

15.2 Points Made in the Case Study

In previous chapters, emphasis has been placed on one aspect of the operation of a system. Modeling an integrated supply chain requires integrating many components in one model: shipments, inventory management, customer demand, production, and information flow. Simulation has a unique ability to provide such integration. A model integrating these components is illustrated in this case study.

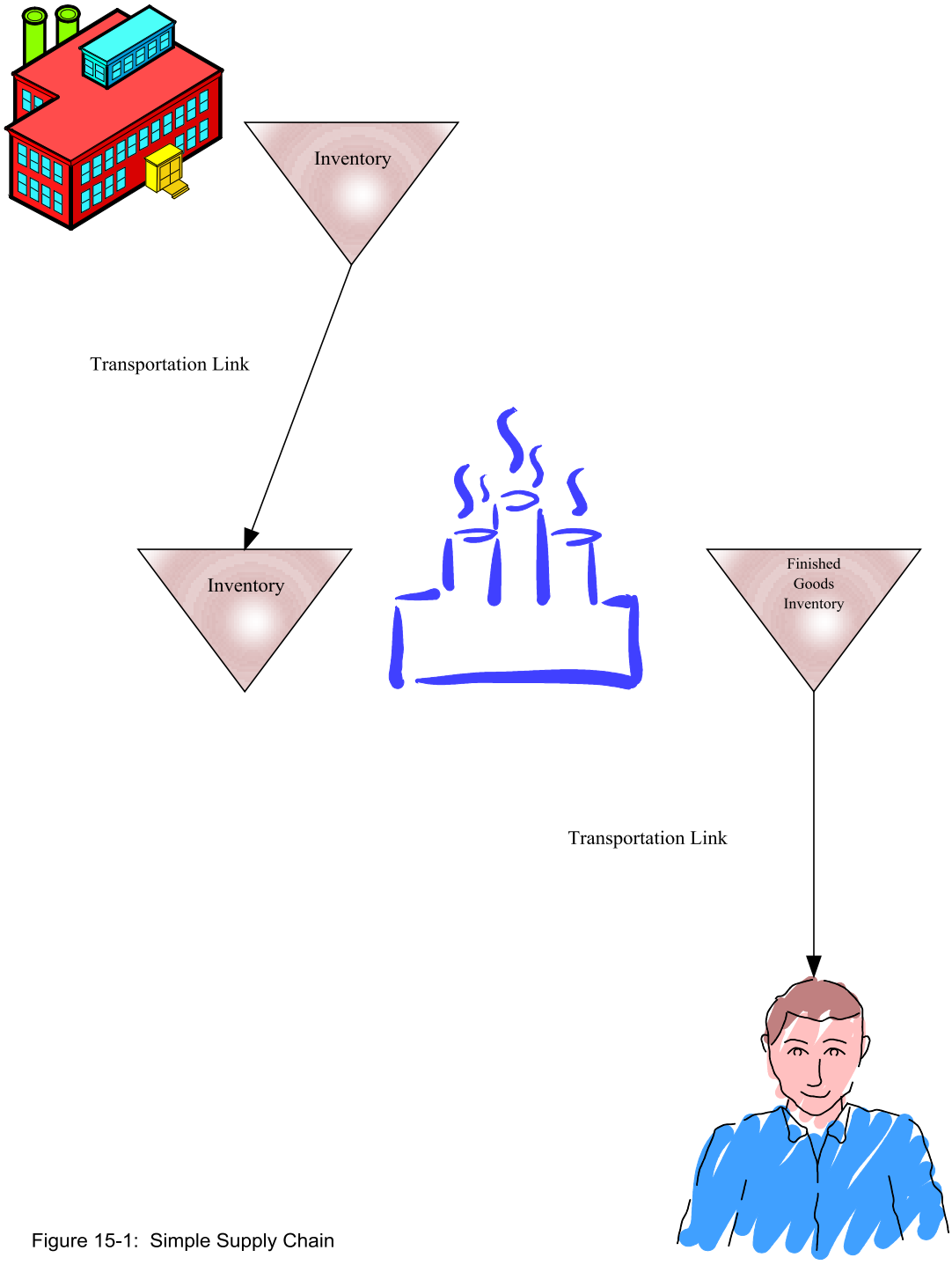


Figure 15-1: Simple Supply Chain

Expected customer demand can vary by month or season of the year. At some times, demand may be far less than production capacity and at some time more. Thus, creating inventory to buffer against demand seasonality is necessary. One approach to doing this is illustrated in this case study.

The effect of the operations of one facility on the decisions made by another facility must be modeled. In this study, equations are used to compute production levels at precedessor plants in a supply chain based on inventory levels at the successor plant, customer demands, and the amount of product in route between the two plants.

The model of a complex system can be implemented using multiple processes. The processes share resources and variables to interact with each other. In this application, nine processes are used to model the supply chain. Variables and resources modeling inventory levels at plant and in route between plants as well as rail fleets are shared between them.

Decisions made within a model may be a function of time. In this application, inventory may be produced in the months prior to peak demand and used only at the time of peak demand.

Supply chain performance is best measured by the service level provide to the customers of retail products. The occasional lack of intermediate inventory for production is acceptable if the customer service level is still satisfactory.

Initial conditions in a supply chain model must include shipments between facilities. In this case, trains are scheduled to arrive at each of the production facilities each day before the expected arrival time of the first train generated by the simulation experiment as a part of the initial conditions.

15.3 The Case Study

A company owns three plants. Two of the plants, Baker and Chauncey, produce retail products for delivery to customers. A third plant, Able, produces two intermediate products for delivery to the Baker and Chauncey plants. This supply chain is pictured in Figure 15-2.

Product is shipped from the Able plant by rail. There is a separate rail fleet for Able to Baker shipments and for Able to Chauncey shipments.

Customer demand for the retail product made by the Baker plant is triangularly distributed with a minimum of 15 rail cars, a mode of 20 rail cars, and a maximum of 40 rail cars per day. Thus, the average daily demand is 25 rail cars.

Customer demand for the retail product made by the Chauncey plant is seasonal. The average daily demand varies by month of the year as shown in Table 15-1. This data is valid for the next year.

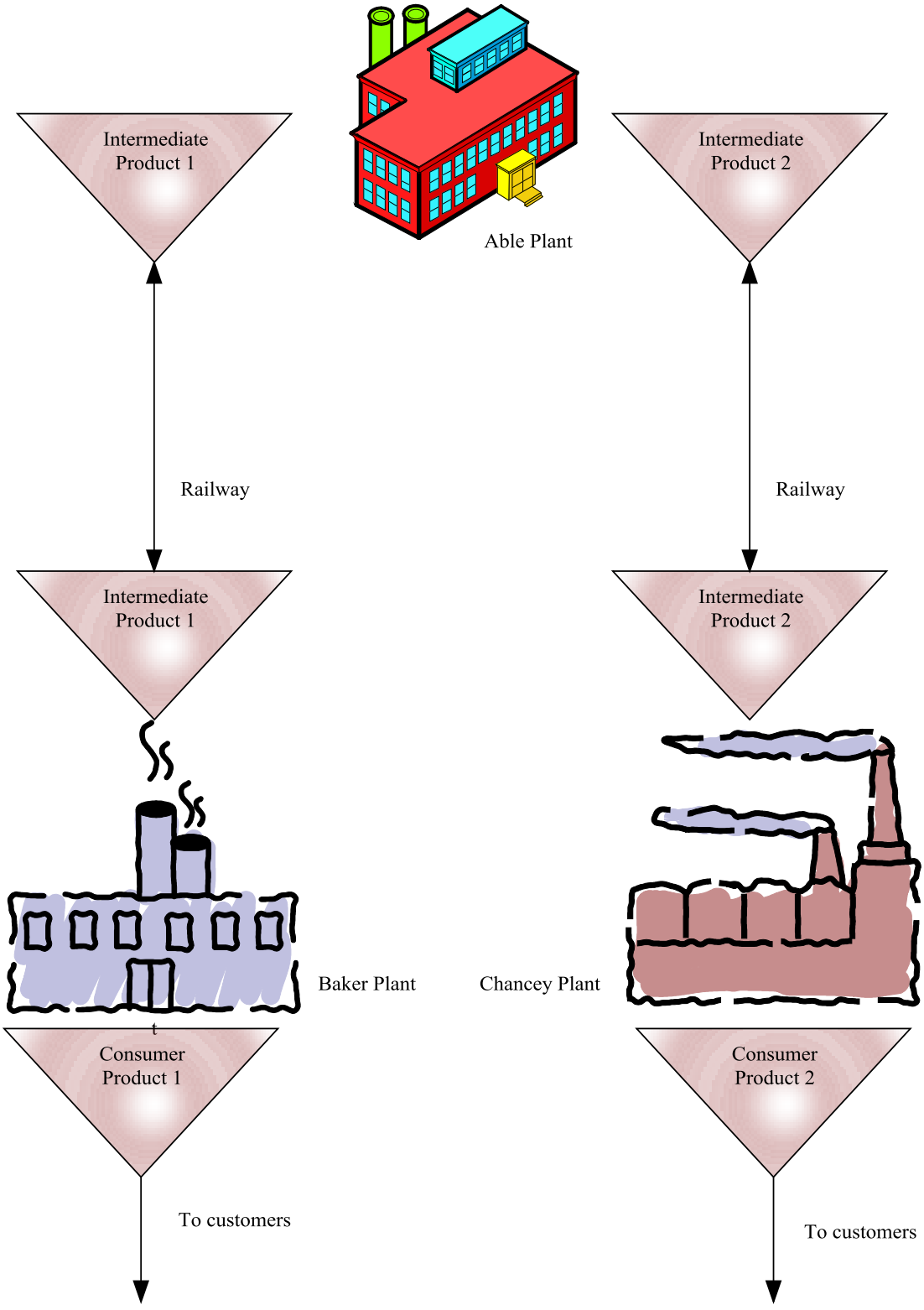


Figure 15-2: Application Study Supply Chain

Table 15-1: Average Demand for the Chauncey Plant Retail Product by Month

Month	Average Daily Demand (Rail Cars)
January	17
February	18
March	18
April	22
May	23
June	24
July	22
August	21
September	21
October	18
November	18
December	18

The average of the average daily demands is 20 rail cars. The minimum demand is 70% of the average and the maximum is 130% of the average.

Daily customer demand can include a fractional number of rail cars. However, only full rail cars are shipped with the fractional demand carried over until the next day.

Production capacity at the Able plant is not an issue as sufficient quantities of each intermediate product can be made each day. Production capacity at the Baker and Chauncey plants is constrained. The Baker plant can produce only 35 rail cars per day. The Chauncey plant can produce 27 cars per day.

Production levels are determined daily. Production at the Baker and Chauncey plants can be viewed as occurring in batches equal to one rail car. A rail car of intermediate product sent from the Able plant is required before a batch can be produced. Production of a batch can be modeled as taking 24 hours / daily plant capacity.

Each day at 4:00 A.M. rail cars leave Able plant for the other two plants. There is one train to each plant. All rail cars sent to a plant travel on the same train. Arriving cars at Baker and Chauncey plants are moved into the plant railyard at 12:00 P.M for use the next day. Empty cars leave these plants for return to Able plant at 4:00 A.M. Travel time between Able plant and Baker plant is triangularly distributed with a mean of 7 days, a minimum of 3 days and a maximum of 10 days. Travel time between Able plant and Chauncey plant is triangulary distributed with a mode of 10 days, a minimum of 7 days and a maximum of 20 days. Rail car maintenance will not be modeled.

Rather than construct inventory facilities at the Baker and Chauncey plants, intermediate product remains in rail cars until needed. One rail car at a time is unloaded in preparation for the start of the next batch. Retail product is loaded directly into rail cars for shipment to customers.

15.3.1 Define the Issues and Solution Objective

The objective of the simulation study is to establish values for the operating parameters of the supply chain for the next year, January through December. These include:

1. The number of cars in each rail fleet: Able plant to Baker plant as well as Able plant to Chauncey plant.
2. The capacity of each inventory: Each of the two intermediate products at Able plant as well as the intermediate and retail product inventories at Baker and Chauncey plants.
3. The target retail inventories at Baker and Chauncey plants.

The primary measure of performance is the service level to customers at Baker and Chauncey plants, defined as the number of days when customer demand was met from existing inventory.

15.3.2 Build Models

The first step in analyzing the supply chain is to set initial target retail inventory levels. One way to do this is as follows, remembering that the simulation experiments can be used to find better values for the target inventory level if necessary.

Consider the target retail inventory level at Baker plant. Suppose there was no variation in customer demand or transportation times. The target inventory level would be equal to one day's demand. Product to meet customer demand would be removed from the retail inventory. The day's production would be used to replenish the inventory to meet the next day's demand.

Because of variation, additional inventory is needed to meet customer demands to a specified service level. Suppose a 95% service level is desired. Then the target inventory can be set such that the probability that customer demand is less than the target is 95%. For Baker plant this is 35 rail cars.

For Chauncey plant, the target will vary by month as shown in Table 15-2. Note that the target inventory levels are at or above plant capacity in 4 of 12 months. This may reduce customer service levels below 95%.

Table 15-2: Target Retail Inventory Levels by Month

Month	Target Inventory Level (Rail Cars)
January	21
February	22
March	22
April	27
May	28
June	29
July	27
August	26
September	26
October	22
November	22
December	22

In addition, the average customer demand at Chauncey plant exceeds the plant capacity in May and June. Thus, management has decided to increase daily production by one rail car per day in January, February, and March to prepare for May and June demand. This inventory will be set aside for use starting in April.

It seems prudent to set each of the intermediate product target inventory levels to the same value as the corresponding retail level, at least initially.

Production levels at all three plants are set using the following relationship:

$$\text{Production} = \text{Target Inventory} - (\text{Current Inventory} + \text{Amount in production}) \qquad (15\text{-}1)$$

In other words, enough units of a product are sent into production so that the sum of these units, the current inventory and the number of units still in production from previous days is equal to the target inventory.

Capacity constraints are applied at Baker and Chauncey plants. In the number of units sent into production is greater than the daily capacity, some of the units will be produced on subsequent days.

The extra production amount is added at the Chauncey plant as well to help meet customer demand in the months where the target inventory is greater than or equal to the plant capacity. This implies the need for additional intermediate inventory that must be shipped from Able plant.

Shipping volumes are set using the following relationship:

$$\begin{aligned}\text{Shipping} = {} & (\text{Target Inventory} - \text{Current Inventory}) + \\ & (\text{Expected customer demand in expected transportation time} - \\ & \text{Amount in route}) \end{aligned} \qquad (15\text{-}2)$$

In addition, the extra production amount is added for shipping between Able and Chauncey plants.

The model consists of nine processes as defined in Table 15-3

Table 15-3: Model Process Definitions

Process Name	Description
Able	Daily operation decisions at Able Plant
Baker	Daily operation decisions at Baker Plant, including customer service
Chauncey	Daily operation decisions at Chauncey Plant, including customer service
BakerMake	Production at Baker Plant
ChaunceyMake	Production at Chauncey Plant
Move2Baker	Train shipment from Able Plant to Baker Plant
Move2Chauncey	Train shipment from Able Plant to Chauncey Plant
Move2AbleBaker	Train shipment from Baker Plant to Able Plant
Move2AbleChauncey	Train shipment from Chauncey Plant to Able Plant

Important variables in the model are shown in Table 15-4.

Table 15-4: Model Variable Definitions

Variable Name	Description
Avg2*	Average transporation time from Able plant to * plant (days)
AvgRetail*	Average daily customer demand
Capacity*	Plant capacity
Cars2Cust*	Number of rail cars demanded by customers currently
Cars2*	Number of rail cars to be shipped from Able plant currently
InRoute*	Number of rail cars currently in route from Able plant
ProductionAdd	Number of additional rail cars of retail product to produce daily at Chauncey plant to meet peak demand. The amount varies by month.
TargetInvRetail*	Target retail (customer) inventory
TargetInvInt*	Target intermediate inventory
TargetInvIntAble*	Target intermediate inventory at Able plant
*toAble	Number of rail cars currently in route to Able plant

* = a plant name (Baker, Chauncey)

The Able process is given in the following pseudo-code. This process models the initiation of the shipment of railcars to Baker plant and Chauncey plant as well as the production of intermediate product at Able plant. Entities in this process represent trains and have one attribute:

CarsinTrain: The number of cars in a train

The two intermediate product inventories, one for Baker plant (IntInvAbleBaker) and the other for Chauncey plant (IntInvAbleChauncey), are modeled as resources. The units of each resource correspond to rail cars. The initial number of units of each inventory resource is equal to the target value for that inventory. The same strategy is used to model the retail inventories at Baker (RetailInvBaker) and Chauncey (RetailInvChauncey) plants.

The two rail fleets are modeled as variables: FleetBaker and FleetChaucey. The model is allowed to create as many rail cars in each fleet as needed. Thus, an estimate of the size of each fleet is obtained. The initial size of each rail fleet is zero.

First consider the shipment of rail cars to Baker plant. The number of cars that need to be shipped is incremented using equation 15-2. Suppose the inventory of intermediate product for Baker plant has at least as many cars as the number that need to be shipped. Then all cars that need to be shipped are shipped, the number remaining to be shipped is zero, and the inventory is reduced by the number of cars shipped.

Suppose more cars need to be shipped than are in inventory. Then the train consists of the cars that are in that are inventory. The number remaining to be shipped is reduce by the number in inventory and the number in inventory is set to zero.

In either case, a clone (copy) of the train entity is sent to process Move2Baker.

The modeling logic for a shipment to the Chauncey plant is identical except for the consequences of the expected customer demand varying month to month. All target inventory values for the intermediate product also vary by month.

Define Arrivals:

Time of first arrival:	0
Time between arrivals:	1 day
Number of arrivals:	Infinite

Define Attributes

CarsinTrain	// Rail cars in a train

Define Variables

AddInv	// Number of additional rail cars needed
Avg2Baker	// Average number of transit days to Baker
AvgRetailBaker	// Average daily customer demand at Baker
Cars2Baker	// Current number of rail cars to ship from Able to Baker
Cars2CustBaker	// Current demand in rail cars at Baker
InRouteBaker	// Current number of rail cars in route between Able and Baker
TargetIntInvAbleBaker	// Target intermediate inventory at Able for Baker
TargetIntInvBaker	// Target intermediate inventory at Baker
Avg2Chauncey	// Average number of transit days to Chauncey
AvgRetailChauncey	// Average daily customer demand at Chauncey
Cars2Chauncey	// Current number of rail cars to ship from Able to Chauncey
Cars2CustChauncey	// Current demand in rail cars at Chauncey
InRouteChauncey	// Current number of rail cars in route -- Able and Chauncey
TargetIntInvAbleChaunceyBaker	// Target intermediate inventory at Able for Chauncey
TargetIntInvChauncey	// Target intermediate inventory at Chauncey

Define Resouces

FleetBaker	// Number of rail cars in the Able to Baker fleet
FleetChauncey	// Number of rail cars in the Able to Chauncey fleet
IntInvBaker	// Number of rail cars in intermediate inventory at Baker
IntInvChauncey	// Number of rail cars in intermediate inventory at Chauncey
IntInvAbleBaker	// Number of rail cars in intermediate inventory at Able for Baker
IntInvAbleChauncey	// Number of rail cars intermediate inventory Able for Chauncey
ProductionChauncey	// Production facility at Chauncey
RetailInvChauncey	// Number of rail cars in finished goods inventory at Chauncey
SavedInvChauncey	// Number of rail cars in build ahead inventory at Chauncey

```
Process AblePlant
Begin
        Cars2Baker += TargetInvIntBaker - #IntInvBaker/IDLE +
                        (Avg2Baker*AvgRetailBaker-InRouteBaker)
        If Cars2Baker <= #IntInvAbleBaker/IDLE then
        Begin
                CarsinTrain = Cars2Baker
                Reduce #IntInvBaker/IDLE by CarsinTrain
                Cars2Baker = 0
        End
        Else
        Begin
                CarsinTrain = #IntInvBaker/IDLE
                Reduce #IntInvBaker/IDLE by CarsinTrain
                Cars2Baker -= CarsinTrain

        End
        Clone to Move2Baker

        Cars2Chauncey += TargetInvIntChauncey - #IntInvChauncey/IDLE +
                            (Avg2Chauncey*AvgRetailChauncey-InRouteChauncey)
        If Cars2Chauncey <= #IntInvAbleChauncey/IDLE then
        Begin
                CarsinTrain = Cars2Chauncey
                Reduce #IntInvChauncey/IDLE by CarsinTrain
                Cars2Chauncey = 0
        End
        Else
        Begin
                CarsinTrain = #IntInvChauncey/IDLE
                Reduce #IntInvChauncey/IDLE by CarsinTrain
                Cars2Chauncey -= CarsinTrain
        End
        Clone to Move2Chauncey

        Wait until Midnight
        AddInv = TargetIntInvBaker - #IntInvAbleBaker/Idle
        If (#FleetBaker/IDLE < AddInv) Then
                increase #FleetBaker/IDLE by (AddInv - #FleetBaker/IDLE)
        Make FleetBaker/AddInv BUSY
        Reduce #IntInvAbleBaker/IDLE by AddInv

        AddInv = TargetIntInvChauncey(Month) - #IntInvAbleChauncey/Idle
        If (#FleetChauncey/IDLE < AddInv) Then
                increase #FleetChauncey/IDLE by (AddInv - #FleetChauncey/IDLE)
        Make FleetChauncey/AddInv BUSY
        Reduce #IntInvAbleChauncey/IDLE by AddInv

End
```

After the train shipments are initiated, time is delayed until midnight when the inventories are updated. Since there is no constraint on production at Able plant, each inventory is simply reset to the target value. In addition, each unit in inventory is stored in a rail car. If there are insufficient idle rail cars at Able plant, additional units of each fleet resource are created.

The remaining discussion of the model will focus on the Chauncey plant. Baker plant operates in an identical way except that time varying average demand is not a factor.

The process Move2Chauncey is shown in the following pseudo-code. The number of rail cars in route to Chauncey is incremented by the number of cars in the train, CarsinTrain. The time delay for movement from Able to Chauncey is determined as a sample from the triangular distribution with minimum 7, mode 10, and maximum 20 days. All trains arrive at midnight. The number of cars in the intermediate product inventory at the Chauncey plant is recorded by increasing the number of idle units of the resource IntInvChauncey. The arriving cars are subtracted from the number of cars in route to the Chauncey plant.

```
Process Move2Chauncey
Begin
        InRouteChauncey +- CarsinTrain
        Wait for Triangualar 7, 10, 20 days        // Train from Able to Chauncey
        Wait until Midnight
        Increase #InvIntChauncey by CarsinTrain
        InRouteChauncey -= CarsinTrain
End
```

Next consider the daily operations at the Chauncey plant. This involves determining the number of rail cars of product demanded by customers, the number of cars that can be shipped from inventory to meet this demand and the number of rail cars of the retail product to produce to replenish the inventory. Additional cars of retail product may need to be produced and saved to meet peak demand. Such cars already in inventory may or may not be available to meet current demand.

The process begins by adding the customer demand for the current day to the currently unfilled customer demand (the variable Cars2Cust). The demand is a sample from a triangular distribution whose mode depends on the month of the year, whose minimum is 70% of the mode and whose maximum is 130% of the mode and can result in a fractional number of rail cars. Only whole rail car loads are shipped so fractional demand, as well as unmet demand, is carried forward to the next day.

If the number of rail cars in the regular inventory is sufficient to meet the customer demand, then the inventory is reduced by the number of rail cars demanded and the remaining customer demand is reduced by the same quantity. If the demand is greater than the number of rail cars in regular inventory, the entire inventory is used to partially meet the demand. The inventory and demand variables are updated accordingly. If the month is April through December, the saved inventory can used to meet the remaining demand, partially or completely.

Service level observations are recorded. If all demand is met, the service level for the day is 100. Otherwise, the service level is zero.

The regular inventory is replenished to the target level by creating an order to produce more rail cars of retail product. The number of rail cars to produce is given by equation 15-1. The number of rail car loads in production is incremented by the right hand side of the same equation.

The saved inventory is built up each day by the number cars depends on the month of the year and is specified in the variable ProductionAdd(Month). Thus, an order for ProductionAdd(Month) additional rail cars is created.

Each order entity corresponds to a single rail car's volume of production and has one attribute.

IsSaved: Whether or not the rail car is a part of the saved inventory (1 Yes; 0 No or regular inventory.)

The Chauncey plant process is given in the following pseudo-code.

```
Define Attributes
        IsSaved         // Is rail car part of saved inventory

Define Variables
        WholeCars       // Integer portion of demand in rail cars
        OrderSize       // How much to produce in rail cars

Process ChaunceyPlant
Begin
        Cars2CustChauncey += triangular 70%*Mode(Month), Mode(Month),
        130%Mode(Month)
        WholeCars         = Integer(Cars2CustChauncey)
        If WholeCars <= #RetailInvChauncey/IDLE Then
        Begin   // Enough Inventory to Meet Demand
                Reduce #RetailInvChauncey/IDLE by WholeCars
                Cars2CustChauncey -= WholeCars
                Tabulate 100 in ServiceLevel
        End
        Else
        Begin   // Not enough inventory to meet demand
                Whole Cars -= #RetailInvChauncey/Idle
                Cars2CustChauncey -= WholeCars
                Reduce #RetailInvChauncey/IDLE by WholeCars
                If (Month is not April through December) Then tabulate 0 in ServiceLevel
                Else
                Begin //Try to use pre-made cars in inventory
                        If (WholeCars <= #SavedInvChauncey/IDLE) Then
                        Begin // Enough pre-made cars to meet demand
                                Reduce #SavedInvChauncey/IDLE by WholeCars
                                Cars2CustChauncey -= WholeCars
                                Tabulate 100 in ServiceLevel
                        End
                        Else
                        Begin // Not enough pre-made cars to meet demand
                                WholeCars -= #SavedInvChauncey/IDLE
                                Cars2Cust Chauncey = -= #SavedInvChauncey/IDLE
                                Reduce #SavedInvChauncey/IDLE by WholeCars
                                Tabulate 0 in ServiceLevel
                        End
                End
        End
        OrderSize = TargetInvRetailChauncey - #InvRetailChauncey/IDLE
        RetailProdChauncey += Ordersize
        IsSaveInv = 0
        Clone OrderSize to MakeChauncey
```

```
        IsSaveInv = 1
        Clone AddProduction(Month) to MakeChauncey
End
```

Chauncey plant production is modeled by process MakeChauncey, which is shown in the following pseudocode. Each entity represents an order to produce one rail car. The entity waits for one rail car sized unit of the intermediate product inventory. After the intermediate inventory is obtained, the entity waits for its turn in the Chauncey production facility. The production time is 1440 minutes (in a day) / 27 (the daily production capacity). Thus, the number of units made per day is limited to the capacity. The newly made unit is added to the appropriate inventory (regular or saved). The rail car containing the intermediate product is sent to wait for the next train to Able plant by adding one to the count of the number of rail cars on the train.

```
Process MakeChauncey
Begin
        Wait until IntInvChauncey/1 to be IDLE
        Make IntInvChauncey/1 Busy
        Wait until ProductionChauncey/1 is IDLE
        Make ProductionChauncey/1 Busy
        Wait for 1440/27 minutes
        Make ProductionChauncey/1 IDLE
        Reduce #IntInvChauncey/Busy by 1
        If IsSavedInv = 0 then
        Begin
                Increase #RetailInvChauncey/IDLE by 1
                RetailProdChauncey -=1
        End
        Else Increase #SavedInvChauncey/IDLE by 1
        Chauncey2Able +=1
End
```

The movement of empty cars from Chauncey plant to Able plant is modeled by process MoveChauncey2Able as shown in the following pseudocode. The number of cars in the train is the number cars containing intermediate inventory that was consumed since the last train departed. The trip is made and the train arrives at midnight to Able plant. One unit of the FleetChauncey resource is freed for each car in the train.

```
Define Arrivals:
        Time of first arrival:      0
        Time between arrivals:  1 day
        Number of arrivals:         Infinite

Process Move2AbleChauncey
Begin
        CarsinTrain = Chauncey2Able
        Chauncey2Able = 0
        Wait for 7, 10, 20 days
        Wait until Midnight
        Make FleetChauncey/CarsinTrain IDLE
End
```

It is important to note when and how each process is initiated. An entity is sent to each of the plant processes: Able, Baker, and Chauncey once each day at midnight. An entity is sent to each process that moves trains to Able plant: Move2AbleBaker and Move2AbleChauncey at the time of daily train departure, 4 A.M. The MakeBaker and MakeChauncey processes are initiated by the Baker and Chauncey plant processes respectively after the number units to make to replenish the inventory has been determined. The Move2Baker and Move2Chauncey processes are initiated by the Able plant process after the number of rail cars to ship to each has been determined.

15.3.3 Identify Root Causes and Assess Initial Alternatives

The design of the initial simulation experiment is shown in Table 15-5. Since the customer demand data is valid for one year, a terminating experiment of length one year is used.

Model parameters are the inventory target levels. Establishing inventory target levels is a primary objective of the simulation study. This will done by setting the target levels in the manor previously described and detemining the resulting system performance. The performance in measured by the customer service level at Baker and Chauncey plants as well as the size of each fleet. In addition, the waiting time of orders for intermediate product at Baker and Chauncey plants so production can begin will be measured. Excessive waiting time could lower customer service levels. Only the waiting time for orders that had to wait is recorded.

There are four random streams, two for transportation times to and from Able plant and two for customer demand at Baker and Chauncey plants. Twenty replicates will be made.

Ideally, the level of each inventory at the end of each day should be the target value. Thus, the target value is used for the initial inventory level.

Trains arrive to Baker and Chauncey plant daily on the average. However, the first shipments from Able plant will not arrive to Baker and Chauncey plants until day 7 and 10 on the average. Thus, shipments must be scheduled to arrive to Baker and Chauncey plants on the preceding days as part of the initial conditions. Shipment size is the average number of rail cars arriving to the plant per day. This is equal to the average customer demand at that plant.

Table 15-5: Simulation Experiment Design for the Supply Chain

Element of the Experiment	**Values for This Experiment**
Type of Experiment	Terminating
Model Parameters and Their Values	1. Retail inventory target levels set to the 95% point of the customer demand distribution 2. Intermediate inventory target levels at Baker and Chauncey plants initially set to the same value as corresponding retail inventory target level 3. Intermediate inventory target levels at Able plant initially set to the same value as the corresponding inventory at Baker or Chauncey plant
Performance Measures	1. Service level to customers at Baker plant 2. Service level to customers at Chauncey plant 3. Fleet size: Able to Baker 4. Fleet size: Able to Chauncey 5. Order waiting time for intermediate inventory at Baker plant 6. Order waiting time for intermediate inventory at Chauncey plant
Random Number Streams	1. Transportation time between Able plant and Baker plant 2. Transportation time between Able plant and Chauncey plant 3. Customer demand at Baker plant 4. Customer demand at Chauncey plant
Initial Conditions	1. All inventory levels set equal to their target 2. Intermediate inventory arrivals to Baker and Chauncey plants as discussed in the text
Number of Replicates	20

Simulated End Time	1 year

Simulation results are shown in Table 15-6.

Table 15-6: Simulation Results for the Initial Experiment

	Fleet Size (Rail Cars)		Service Level		Wait for Inventory (Hours)	
Replicate	**Baker**	**Chauncey**	**Baker**	**Chauncey**	**Baker**	**Chauncey**
1	501	666	43	5	31	33
2	500	722	55	18	29	35
3	502	671	49	7	30	36
4	468	724	68	26	30	33
5	467	700	85	72	26	35
6	501	704	61	33	29	31
7	500	684	48	9	34	36
8	502	719	54	44	31	34
9	494	724	39	6	32	33
10	495	698	43	21	30	35
11	486	732	61	39	28	33
12	484	709	60	28	28	32
13	481	749	64	7	29	32
14	489	675	68	51	28	32
15	534	717	61	39	31	32
16	472	722	28	24	33	34
17	501	737	54	9	32	37
18	489	717	44	38	33	32
19	488	736	52	26	30	33
20	476	695	51	18	32	33
Average	492	710	54	26	30	33
Std. Dev.	15	23	13	18	2	2
99% CI Lower Bound	482	695	46	15	29	32
99% CI Upper Bound	501	725	62	37	32	34

Service level values are unexceptably low. Order waiting time for intermediate inventory averages greater than one day at each plant. An average of 1339 orders per replicate waited for intermediate inventory at the Baker plant with an approximate 99% CI of (1152, 1526) while an average of 1094 orders per replicate waited for intermediate inventory at the Chauncey plant with an approximiate 99% CI of (949, 1238).

These results lead to a second alternative. The target intermediate inventory at Baker plant is increased by the expected customer demand in one day as the waiting time for intermediate inventory averages about 1.25 days. Similarly, the target inventory at Chauncey plant is increased by the expected customer demand in two days as the waiting time for intermediate inventory is about 1.4 days. Otherwise the simulation experiment is the same as shown in Table 15-5. Results are shown in Table 15-7.

Table 15-7: Simulation Results with Higher Intermediate Inventory Targets

	Fleet Size (Rail Cars)		Service Level		Wait for Inventory (Hours)	
Replicate	**Baker**	**Chauncey**	**Baker**	**Chauncey**	**Baker**	**Chauncey**
1	520	770	93	97	24	30
2	483	795	97	96	24	47
3	501	771	93	90	24	33
4	492	763	93	99	24	26
5	516	778	91	94	25	27
6	499	744	93	95	30	25
7	502	750	95	97	24	26
8	532	819	91	91	25	30
9	499	801	94	100	24	48
10	494	846	96	97	25	39
11	516	782	90	93	25	29
12	504	784	95	97	24	28
13	497	813	94	96	29	34
14	499	774	92	99	27	36
15	505	799	97	93	26	28
16	487	773	95	96	24	41
17	492	748	96	96	28	32
18	524	808	93	96	25	33
19	511	808	94	95	25	25
20	502	779	93	95	25	32
Average	504	785	94	96	25	32
Std. Dev.	13	26	2	3	2	7
99% CI Lower Bound	496	769	92	94	24	28
99% CI Upper Bound	512	802	95	97	26	37

Results show that the approximate 99% confidence intervals for the service level at both the Baker plant and the Chauncey plant contain the target service level of 95%. The fleet size required for the Chauncey plant is 785 cars and the fleet size required for Baker plant is 504 cars. An average of 122 orders per replicate waited for intermediate inventory at the Baker plant with an approximate 99% confidence interval of (83, 161) while an average of 124 orders per replicate waited for intermediate inventory at the Chauncey plant with an approximate 95% confidence interval of (100, 148). Note that number of orders waiting at each plant has dropped by about an order of magnitude.

Since service levels are acceptable for this alternative, inventory capacity can be examined. The retail inventories at Baker and Chauncey plant can by design not exceed the target. The same is true for the intermediate inventories at Able plant. Thus, only the inventory capacities to be set are the intermediate inventories at Baker and Chauncey plant. The approximate 99% confidence interval for the maximum number of rail cars in the intermediate inventory at Baker plant is (135, 145) with an average of 140. The approximate 99% confidence interval for the same quantity at Chauncey plant is (155, 170) with an average of 162.

15.3.4 Review and Extend Previous Work

Management was willing to except a slightly less than 95% service level at Baker plant. Fleet sizes of 504 for Able to Baker and 785 for Able to Chauncey will be used. The number of orders waiting for intermediate inventory as well as the average waiting time were felt to be acceptable.

The target inventory levels are as follows. Note that target inventory values associated with Chauncey plant vary by month.

Retail inventories:	35 at Baker plant, the 95% point of the demand distribution as shown in Table 15-2 at Chauncey plant.
Intermediate inventories:	60 (35 + 25 = the expected demand in one day) at Baker plant and for the Baker plant intermediate inventory at Able plant 35 + 2 * the monthly value shown in Table 15-1 for the Chauncey plant

Inventory capacities were set as follows.

Customer inventories:	Same as target inventories.
Intermediate inventories:	Able plant – Same as target inventories. Baker plant – Same as the average maximum of 140 rail cars Chauncey plant – Same as the average maximum of 162 rail cars

15.3.5 Implement the Selected Solution and Evaluate

The supply chain will be operated with the above parameters. Service level performance will be monitored.

15.4 Summary

This chapter discusses the use of simulation to analyze complex systems in general and supply chains in particular. Some components of such systems have time varying characteristics such as the expected customer demand for products. The behavior of one component may depend on the behavior of other components. Customer demand at one facility is a factor in determining shipping quantities at another facility.

Models can be constructed by viewing the complex system as a set of semi-independent processes that share information using modeling constructs such a variables and resources. Simulation experiments include initial conditions that specify time dynamic behavior such as the arrival of shipments over time at a facility. A variety of simulation results can be collected and the behavior of many of aspects of such a system can be assessed.

Problems

1. Distinguish the integrated supply chain approach from the automated inventory management approach discussed in chapter 13.

2. Compare the flow of information in an integrated supply chain to the flow of work in a kanban system such as the one discussed in chapter 10.

3. What other factors should be taken into account in setting the initial target inventory levels for the intermediate products at each plant in addition to the variation in customer demand?

4. What information would be lost if two models were used instead of the one model in this chapter? One model would represent the supply chain between Able plant and Baker plant and the other between Able plant and Chauncey plant.

5. State the model for each of the following processes:

   a. Baker
   b. BakerMake
   c. Move2Baker
   d. Move2AbleBaker

6. Instead of scheduling daily train arrivals at Baker and Chauncey plants as a part of the initial conditions, discuss the effects of simply increasing the initial inventory at each plant by an amount equal to the total volume arriving due to the initially scheduled shipments.

7. Why is the fleet size larger for Chauncey plant than Baker plant when the customer demand is smaller?

8. Modify the model discussed in this chapter to determine how many rail cars would be saved if there was only one rail fleet used to ship product from the Able plant.

9. Evaluate the policy of shipping an amount equal to the expected customer demand from Able plant to each of the other two plants each week. For Chauncey plant this means the expected time of train arrival should be used in determining which expected demand to use.

10. Modify the model in this chapter to estimate the rail yard size needed at Able plant.

11. Suppose that the capacity of Able plant is 62 rail cars per day in sum total over all products produced. Analyze the supply chain for this case. Generate additional inventory needed to support Chauncey plant ahead of time if needed.

12. Analyze the supply chain for the following case: Allow the saved inventory at Chauncey to be used to meet demand during the first three months of the year. Replace all saved inventory that is used in this way.

13. Modify the model so that the size of each rail fleet can be constrained to a pre-specified upper limit. Find the smallest rail fleet size that results in an acceptable service level.

14. Currently the model described in this chapter assumes that customers will accept backorders and late deliverys. Modify the model so that demand which can not be met on time results in lost sales. Conduct simulation experiments to estimate the volume of lost sales and reset the operating parameter values of the supply chain to minimize lost sales.

15. Modify the model described in this chapter so that fractional demand is met if inventory is available.

16. Rerun the simulation model to collect verification and validation evidence. This evidence could include:
    a. One table for each inventory. The rows of the table correspond to days. There is one column for each of the following: Level at the beginning of the day, additions, removals, and level at the end of the day.
    b. The number of orders at each plant that way for intermediate inventory.
    c. Month by month service levels at Chauncey plant

Case Problem

A company supplies a customer product for which daily demand, expressed in truck loads, is normally distributed with a mean of 10 and a standard deviation of 2. Production capacity is 14 truck loads per day. Delivery time is equally distributed between one day and two days that is it takes either one or two days for the truck to travel to the customer, deliver the load, and return to the company site.

There is one truck load of raw material per truck load of final product. Raw material is obtained from a supplier. Travel time from the company site for each truck is as follows: one day to the supplier, one day (80%) or two days (20%) at the supplier, and one day to return to the company site. .

The same truck fleet is used for both product delivery and raw material acquisition.

Determine the size of the truck fleet. In addition, determine the target inventory level for raw material and the inventory target level for the consumer product needed for a 95% service level for delivery to customers.

Generate a trace of the dynamics of each the two inventories that shows the following information by day.

- Simulated day
- Inventory level at start of day
- Inventory consumed during the day
- Inventory added during the day
- Inventory level at the end of the day

The time period of interest is one year (365 days).

Case Problem Issues

1. Compare this problem to the case problem in chapter 14.

2. How could a lower bound on the truck fleet size be computed?

Part V
Material Handling

In previous chapters, entity movement between stations was not included in models and movement times were implicitly assumed to be negligible. However, these assumptions are not always satisfactory. Material movement or handling may be a signicant component of a manufacturing system. Note that such movement and the devices required to perform it do not add value to a service or product. Thus material handling is inconsistent with the lean philosophy, increasing both capital equipment cost and lead time. Efficient and cost effective material handling is essential to successful operations.

It has been claimed that the most effective tool for evaluating the performance of material handling system designs is simulation. Alternative strategies can be evaluated. Competing equipment can be compared. The ability of a particular design to meet performance criteria such as a throughput target can be assessed.

Chapter 16 discusses transfer hubs. Such hubs are an integral part of the operations of shipping companies that transport packages. A hub provides for the sorting and routing of voluminous packages within a short time frame. This is most often accomplished using a series of conveyor systems. A transfer hub routes the in bound packages to their final destination or another hub.

Chapter 17 deals with issues concerning automated guide vehicles (AGV's). An AGV system is used to move loads of parts along predetermined paths between workstations without manual operation. Simulation is used to confirm the operational effectiveness of an AGV system that has been designed by other means.

Chapter 18 discusses the use of an automated storage and retrieval system (AS/RS) to manage inventory. An AS/RS system provides for the high-speed storage and movement of part and other materials. The computer system that is part of an AS/RS provides for real time inventory management. Simulation is used to evaluate alternative AS/RS storage configurations.

Chapter 16[1]

Transfer Hubs

16.1 Introduction

Companies such as FedEx and United Parcel Service specialize in the delivery of packages often when time is critical. The network of trucks and airplanes employed by such a company transports millions of packages to both business and personal customers each year.

The ground based shipping methods employed by these companies typically rely on a network of terminals and hubs to move packages throughout the country. Vans are used to pick up packages from customers and deliver them to a small terminal. If a package needs to be sent outside of the terminal's delivery area, it is loaded onto a tractor-trailer truck and sent to a hub.

Most hubs are located in major cities with hundreds of the smaller terminals located in smaller cities. Tractor-trailers containing packages to be shipped a great distance across the country can be loaded onto railcars to reduce cost. When the tractor-trailer arrives at a hub, the packages it contains are sorted by destination. Outbound packages can be loaded into vans for local delivery or sent to other hubs throughout the network.

At the heart of the hub is the material handling system usually a conveyor system. The conveyor system is used to unload, sort and transport packages throughout the hub. Hub facilities may be of enormous size, some containing 8 miles of conveyor. The hub material handling system is built up in phases. Each phase typically adds a copy of the original system as well as expanding it. Phased development reduces the financial risk associated with installing the complete system before the demand to support it exists.

The material handling structure employed by a typical hub is shown in Figure 16-1. A truck arrives to one of many docks that comprise the unload area. A large conveyor is extended into the truck. A worker manually unloads each truck and places the packages it contains on the conveyor.

A set of conveyors, usually four in number, used in truck unloading is called a bank. Typically, each pair of unload banks feeds a primary sorter, which processes packages from multiple unloading doors at once. A variety of logic is used to merge packages on to a single conveyor before reaching the primary sorter.

Each of the primary sorters routes packages to one of many secondary sorters. A secondary sorter routes each package to a particular lane and hence to an outbound truck. A worker removes each package from a lane and places it in the proper truck. A lane corresponds to a particular zip code or truck destination. A typical secondary sorter supports 20 lanes.

[1] Mr. Joel Oostdyk assisted with the development of this application study.

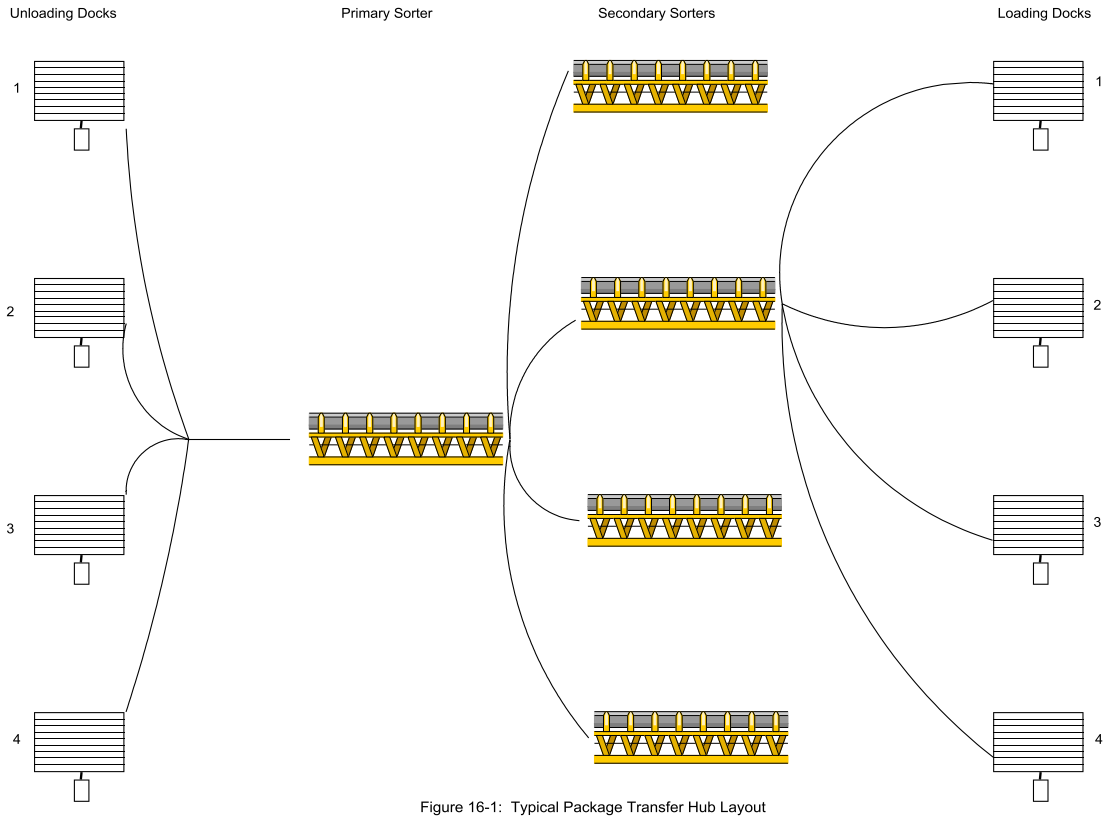


Figure 16-1: Typical Package Transfer Hub Layout

16.2 Points Made in the Case Study

This case study deals with modeling issues concerning conveyors. A conveyor is viewed as consisting of multiple segments. Certain segments, such as the exit points for work stations, are key. Key segments are modeled as resources with the number of units equal to the capacity of the segment. When all resource units are busy, the key segment is full and other items "back up" along preceding segments of the conveyor. The non-key segments are modeled as time delays only.

Key segments and operations are modeled similarly. A scarce system object is modeled as a resource constraining the movement of an entity. An entity uses this object for a length of time and then releases it for use by other entities.

Package travel time on a conveyor is determined from specifications of the speed of the conveyor and the distance the package must travel.

Many simulation languages have special modeling constructs for representing conveyor systems in a model. These constructs contain the logic for modeling key and other segments. Thus, this logic can be included in a model transparently to its developer.

In some cases, an operation may be performed by any of several workers or machines. In the model, this implies a choice between resources. The logic for making this choice must be specified.

Multiple individually distinguishable resources may be identified by the same name. The individual resources each have a unique ID number or index. For example a model could represent 10 workers with the resource WORKER and worker 7 could be referenced by WORKER(7).

Ergonomic considerations can be included in a model. In this case, worker walking time as well as allowances for rest and other personal time are taken into account.

Performance measures can be computed from other performance measures. In this case, the average utilization for a group of workers is computed from the utilization of each individual worker.

16.3 The Case Study

The following case study is a subset of one described by Warber and Standridge (2002). A package sorting hub is entering an expansion phase. The number of unloading banks, primary sorters and secondary sorters is increasing to support processing an increased volume of packages. Secondary sorter operations are of particular interest.

16.3.1 Define the Issues and Solution Objective

The level of staffing is a significant cost component for a transfer hub. Thus, the number of workers assigned to loading out bound trucks is at issue. Management believes that a worker can support more than one secondary sorter lane at a time. For example, a worker supporting two lanes would wait until a package arrives to one of the two lanes, walk to that lane, place the package on the truck, and return to look for the next arriving package on either lane. Note that in addition to the time to load a package into a truck, the walking time to a lane must be taken into account.

The number of workers to assign to the secondary sorter must be determined. The number of workers should be minimized to reduce costs. At the same time, loading delays are detrimental to hub operations. Thus, the time to load a package should be minimized. These two operating criteria are in conflict and a suitable balance between the two must be found.

A simulation study will be done to determine the number of workers to assign per secondary sorter. Trucks containing packages arrive to the terminal between 4:00 P.M. and 8:00 P.M. each day. It is estimated that on the average 8000 of these packages will be processed by the secondary sorter of interest. Since many packages are also sent to other secondary sorters, the time between arrivals the secondary sorter of interest is considered to be an exponentially distributed random variable with mean 4 hours / 8000 packages or 1.8 seconds.

The secondary sorter serves 20 loading lanes each leading to a loading dock. A package is equally likely to be routed to any of the loading lanes. The distance between loading lanes is 10 feet measured from the center point of one loading lane to the center point of the next. A detailed drawing of the secondary sorter of interest is given in Figure 16-2.

The distance from the secondary sorter to a loading door is 37 feet. The total length of the secondary sorter conveyor is 250 feet. Conveyor speed is 1 foot per second.

Loading time consists of two components: the time for a worker to remove a package from the end of the loading lane and place it properly in the a truck and the time for the worker to walk to a loading lane. The former can be modeled as a random variable since the location of a particular package in a truck depends on the packages currently in the truck. Experience has found the loading time to be highly variable with a mean of 8 seconds. Thus, loading time is considered to be exponentially distributed.

The time for a worker to walk to a loading lane depends on how many lanes the worker serves. If the worker serves two lanes and waits half way between them for an arriving package, the walking distance is five feet. Assuming the average walking speed is 2 miles per hour, the

average walking time is about 1.7 seconds. This time is about 21% of the average time to place a package on a truck and thus is a significant factor in determining system performance.

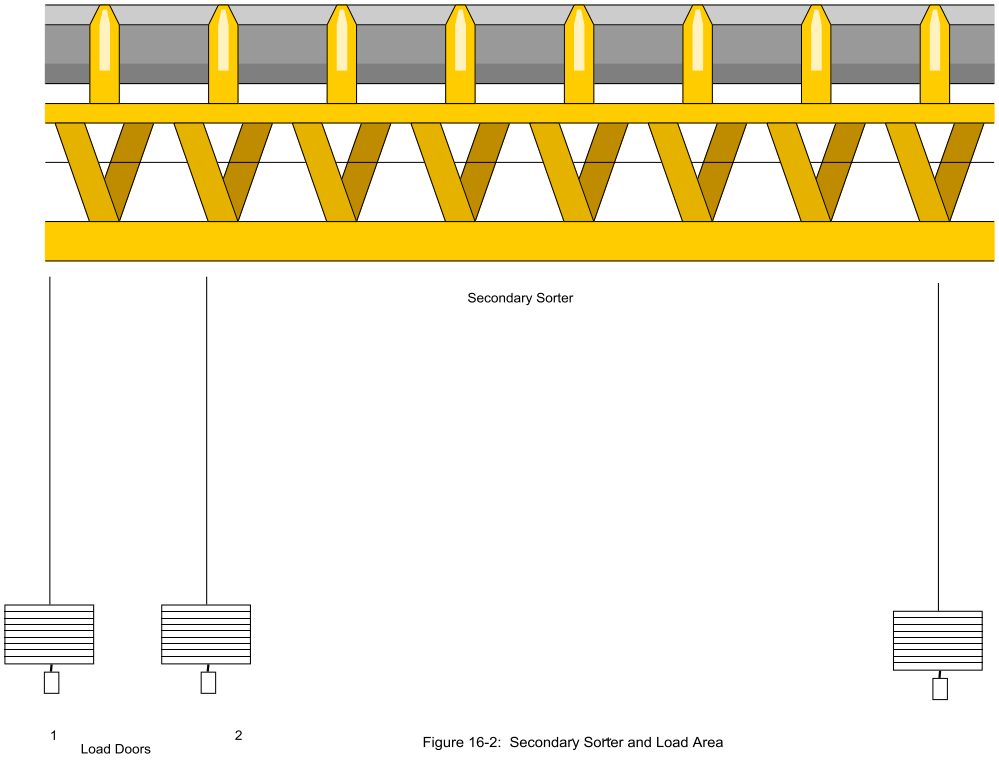


Figure 16-2: Secondary Sorter and Load Area

16.3.2 Build Models

The model of the secondary sorter operations must take into account the following system components.

1. Arrival of packages to the secondary sorter between 4 P.M. and 8 P.M. with an exponentially distributed time between arrivals with a mean of 8 seconds.
2. Package movement along the secondary sorter conveyor until the lane corresponding to the loading door is reached.
3. Package movement to the end of the lane.
4. Loading of the package on the truck.
5. Worker assignment to lanes including walking time to a lane.

Arriving entities model packages and have the following attributes.

1. Lane: Loading lane assignment, 1, 2, …, 20.
2. TimeArriveLane: Time of arrival to the end of a lane.
3. LaneWorker: ID of the particular worker resource assigned to lane Lane.

The latter attribute allows the time a package waited for a worker for loading to be collected.

Model logic is shown in the following pseudocode. Packages arrive according to an exponential distribution with mean 1.8 seconds. The lane from which the package will be loaded is computed as a sample from a uniform distribution between 1 and 21. Thus, each of the lanes 1 through 20 is equally likely. The package moves on the secondary sorter conveyor at the rate of 1 foot per

second to the selected lane. Then the package moves down the lane to its end at the same rate. The arrival time at the end of the lane is noted. The package waits at the end of the lane for the worker serving that lane. The waiting time is collected. The worker walks to the lane in 1.7 seconds and then loads the package in an exponentially distributed time with a mean of 8 seconds. After this task, the worker becomes IDLE again.

```
Define Arrivals
        Time of first arrival:    0
        Time between arrivals:    Exponential 1.8 seconds
        Number of arrivals:       Infinite

Define Attributes
        Lane                      // Loading lane assignment, 1, 2, …, 20.
        TimeArriveLane            // Time of arrival to the end of a lane.
        LaneWorker                // ID of the particular worker resource assigned to lane Lane.

Define Resources
        Worker(2)                 // Lane workers

Process SecondarySorter
Begin
        Lane = Integer (Uniform 1, 21)                      // Select Lane
        Wait for (1 sec * distance to lane in feet)         // Move to lane
        Wait for (1 sec * length of lane conveyor in feet)  // Move to load area
        TimeArriveLane = Clock
        LaneWorker = (Lane+1)/2                             // Select lane worker
        Wait until Worker(LaneWorker)/1 is IDLE
        Make Worker(LaneWorker)/1 BUSY
        Tabulate (Clock-LaneArrivalTime) in WaitforWorker
        Wait for 1.7 seconds                                // Worker walks to lane
        Wait for Exponential 8 seconds                      // Worker loads truck
        Make Worker(LaneWorker)/1 IDLE
End
```

Model logic for a conveyor deserves more detailed discussion. Consider a lane conveyor. The conveyor is divided into segments. Each segment can contain one package so each segment is the size of a package. The segment at the end of the lane is called a key segment. The key segment is modeled as a resource so that only one package can occupy the key segment at a time. Packages waiting for the key segment to become idle occupy the segments physically preceding the key segment. If enough packages are waiting, the lane could become full and block the secondary sorter conveyor.

When modeling a conveyor, the size of entities traveling on the conveyor and the key segments must be specified along with the conveyor speed. The use of the non-key segments as queuing space for a key segment must be included in the model. Figure 16-3 summarizes these ideas. An entity moves on the lane until it reaches the non-key segment closest to the key segment that is not occupied by another entity. Each entity waits to enter the key segment. As an entity departs the key segment, all remaining waiting entities move one non-key segment closer to the key segment.

Fortunately, the above logic is included in the modeling constructs of many simulation languages. Thus, the modeler is required only to specify the conveyor parameters, for example package size, conveyor speed, conveyor length, and key segment location.

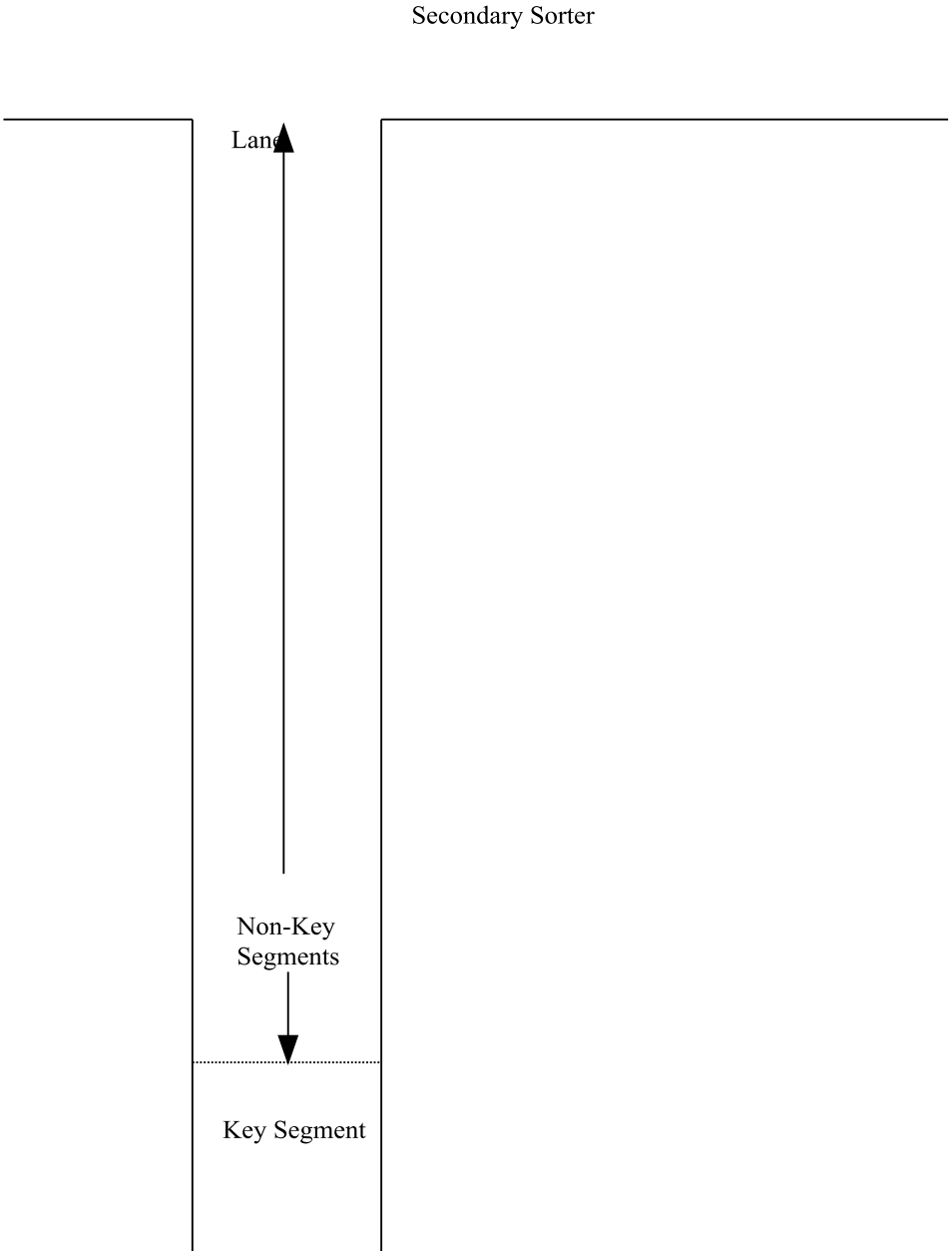


Figure 16-3: Model of a Lane Conveyor

16.3.3 Identify Root Causes and Assess Initial Alternatives

Management desires that the workers be kept as busy as possible. On the other hand, ergonomic considerations require worker rest and personal time to be about 20% of the work period. Thus, an average worker utilization of 80% is sought and this quantity is one performance measure. The time a package waits for a worker before loading is also of interest.

One model parameter will be varied, the number of lanes server by a worker, either 2 or 3. Note that worker walking time to a lane will increase when 3 lanes are served. The worker will stand at the middle lane of the three being served. The walking distance to the middle lane is therefore neglibile. The walking distance to each of the other two lanes is 10 feet. Thus, the average walking distance increases from 5 feet to 6.67 feet and the average walking time increases from 1.7 seconds to 2.3 seconds. Having each worker serve 3 lanes instead of 2 reduces the number of workers from ten to seven. Six of the seven workers serve 3 lanes and the seventh server the remaining two lanes.

Since trucks arrive with packages between 4 P.M. and 8 P.M. each day, a terminating simulation experiment of duration 4 hours is employed. Twenty replicates will be made. Since there are no packages at the secondary sorter at 4 P.M., the initial conditions are all lanes empty and all workers idle.

There are three random number streams used in the model, one for package arrivals, one for lane assignments, and one for package loading time onto trucks.

The experiment is summarized in Table 16-1.

Table 16-1: Simulation Experiment Design for the Secondary Sorter

Element of the Experiment	**Values for This Experiment**
Type of Experiment	Terminating
Model Parameters and Their Values	Number of lanes served by one worker (2 or 3)
Performance Measures	1. Average utilization over all workers 2. Waiting time for a worker
Random Number Streams	1. Time between arrivals 2. Lane assignment for a package (1-20) 3. Loading time
Initial Conditions	Empty and idle
Number of Replicates	20
Simulation End Time	4 hours

Simulation results for the cases where a worker serves 2 and 3 lanes are shown in Table 16-2. Average worker utilization is the average utilization of all workers in the first case and of only those workers serving 3 lanes in the second case.

Table 16-2: Average Worker Utilization and Package Waiting Time for a Worker at the Secondary Sorter

	Worker Serves Two Lanes		Worker Serves Three Lanes	
Replicate	**Average Package Waiting Time (sec)**	**Average Worker Utilization**	**Average Package Waiting Time (sec)**	**Average Worker Utilization**
1	3.1	0.533	10.7	0.843
2	2.8	0.520	9.8	0.832
3	2.9	0.529	10.3	0.839
4	2.9	0.535	10.8	0.845
5	2.8	0.529	10.4	0.845
6	2.9	0.528	10.4	0.842
7	2.9	0.527	10.4	0.837
8	3.0	0.527	10.1	0.839
9	3.0	0.535	10.3	0.844
10	3.1	0.538	10.6	0.855
11	3.0	0.530	10.2	0.841
12	2.9	0.527	9.9	0.835
13	3.1	0.533	10.4	0.844
14	3.2	0.546	11.3	0.870
15	2.9	0.537	10.8	0.853
16	2.9	0.530	10.3	0.843
17	2.9	0.536	10.7	0.852
18	3.1	0.536	11.4	0.858
19	2.9	0.534	10.7	0.849
20	3.0	0.526	10.3	0.836
Average	3.0	0.532	10.5	0.845
Std. Dev.	0.096	0.00560	0.398	0.00903
99% CI Lower Bound	2.9	0.528	10.2	0.839
99% CI Upper Bound	3.0	0.535	10.7	0.851

Note that in neither case does the approximate 99% confidence interval contain the target utilization of 80%. Package average waiting time increases by about 3.5 times when a worker serves three lanes instead of 2.

16.3.4 Review and Extend Previous Work

Management was disappointed that neither assigning 2 or 3 lanes to a worker produced the desired utilization of 80%. The slightly higher utilization of 84.5% on the average was deemed unacceptable since upper bound of the 99% confidence interval was 85.1%. A worker utilization of 53.2% was deemed too low and thus too costly.

The following alternative was proposed. Each worker would serve 2 lanes alone plus sharing the responsibility for a third lane with another worker. This would increase the number of workers from seven serving 3 lanes each to eight serving 2.5 lanes each. Thus, the simulation project process was restarted at the Build Models step.

16.4.1 Build Models

The average walking time when a worker serves two lanes and shares responsibility for a third lane was computed as follows. A worker stands in the same position as when serving 2 lanes. Thus, the average walking time is 1.7 seconds for 80% of the package loading operations. For the other 20% of the package loads, the walking distance is 15 feet, which requires 5.1 seconds on the average. Thus, two walking times must be included in the model.

A new version of the model was created to model two workers sharing responsibility for every third lane. The shared lanes are 3, 8, 13, and 18. No changes to model logic are required for non-shared lanes. For shared lanes, the changes to model logic are as follows.

1. Wait for either lane worker to perform the loading operation, whichever one becomes IDLE first.
2. Use the walking time to a shared lane, 5.1 seconds.
3. Free whichever worker performed the loading operation.

16.4.2 Identify Root Causes and Assess Initial Alternatives

The experiment is the same as the one define in Table 16-1 except for the performance measures. Waiting time for each of two types of packages is required: those using lanes served by one worker alone and those using lanes servered by two workers.

Simulation results comparing the two cases are shown in Table 16-3.

In the shared lanes scenario, all workers serve the same number of lanes, 2.5. The average worker utilization is 66.4%, less than the desired 80% target but more than in the case where each worker serves only two lanes. Average package waiting time is about half of that in the workers serve 3 lanes case. Average package waiting time is less on the shared lanes than on the lanes that do not share a worker.

16.4.3 Implement the Selected Solution and Evaluate

Management was disappointed that the target worker utilization of 80% could not be achieved but satisfied with the using eight workers instead of the ten required by the case in which a worker servers two lanes only. Average package waiting time was deemed satisfactory.

16.5 Summary

This chapter discusses the modeling and analysis of a package transfer hub. Specifically techniques for modeling conveyor systems have been presented. The choice between alternative resources for performing an operation has been illustrated. Ergonomic considerations have been included in the model. The number of workers to serve a loading operation was determined.

Table 16-3. Average Worker Utilization and Package Waiting Time for a Worker at the Secondary Sorter – Shared Lanes Case

	Worker Serves Three Lanes		Worker Serves Two Lanes Plus a Shared Lane		
Replicate	Average Package Waiting Time	Average Worker Utilization	Average Package Waiting Time Non-Shared Lanes	Average Package Waiting Time Shared Lanes	Average Worker Utilization
1	10.7	0.843	5.3	5.0	0.664
2	9.8	0.832	4.9	4.2	0.650
3	10.3	0.839	5.1	3.8	0.661
4	10.8	0.845	5.2	4.5	0.669
5	10.4	0.845	5.3	4.4	0.661
6	10.4	0.842	5.3	4.4	0.660
7	10.4	0.837	5.1	4.1	0.659
8	10.1	0.839	5.2	4.3	0.659
9	10.3	0.844	5.2	4.1	0.669
10	10.6	0.855	5.2	4.2	0.671
11	10.2	0.841	5.2	4.2	0.663
12	9.9	0.835	5.2	4.2	0.658
13	10.4	0.844	5.4	4.5	0.666
14	11.3	0.870	5.7	4.8	0.683
15	10.8	0.853	5.5	4.3	0.670
16	10.3	0.843	5.5	4.7	0.662
17	10.7	0.852	5.2	4.1	0.670
18	11.4	0.858	5.7	4.7	0.670
19	10.7	0.849	5.2	4.2	0.667
20	10.3	0.836	5.2	4.2	0.657
Average	10.5	0.845	5.3	4.3	0.664
Std. Dev.	0.398	0.00903	0.193	0.284	0.00702
99% CI Lower Bound	10.2	0.839	5.1	4.2	0.660
99% CI Upper Bound	10.7	0.851	5.4	4.5	0.669

Problems

1. Explain how sampling from the continuous uniform distribution with minimum 1 and maximum 21 gives equal probability to the integers 1 through 20 and no probability to the integer 21 when samples from truncated to integer values.

2. Why is the time between the arrival of a package to the secondary sorter and completion of loading on a truck not a good performance measure? Supply an improved definition for this performance measure.

3. Develop a model for a lane served by either of two workers.

4. Perform a formal statistical analysis using paired confidence intervals and the data in Table 16-3 to confirm that package waiting time is less in the workers share lanes case than in the case where a worker serves 3 lanes.

   a. Compare average package waiting time with each worker serving 3 lanes (2nd column) with the average waiting time for lanes served by only one worker, the non-shared lanes (4th column).

   b. Compare average package waiting time in the shared lines (5th column) and the non-shared lanes (4th column).

5. Explain why the average waiting time for packages for a shared lane served by two workers (5th column) is less than for lanes served by one worker (4th column) as seen in Table 16-3.

6. Perform a formal statistical analysis using paired confidence intervals and the data in Tables 16-2 and 16-3 to compare the average package waiting time between the worker serves two lanes scenario (2nd column) and the shared lanes scenario (5th column).

7. Explain why average package waiting time increases in a non-linear fashion as the utilization of the workers increases.

8. Go to a manufacturing lab, transfer hub, or a local manufacturing plant to observe a conveyor system in operation. List the number of different conveyor types found.

9. Embellish the model to make package loading time a function of how many packages are on a truck. Assume 8 seconds is the mean time to load the package in the center of the truck and each truck holds 200 packages. The mean loading time varies linearly from 12 seconds for a completely empty truck to 4 seconds for the last package on a truck. After the 200th package is loaded on a truck, the fully loaded truck swaps positions with an empty truck in 3 minutes. No package loading can occur during this time. Determine the number of workers needed under these conditions.

10. Suppose that packages are not uniformly distributed across final destinations but the distribution by destination is shown in the following table. Use simulation to assign the package destinations to lanes as well as workers to lanes. The destinations may be assigned to lanes in any way that is helpful.

Package Destination	Percent of Packages	Package Destination	Percent of Packages
1	0.48%	11	5.24%
2	0.95%	12	5.71%
3	1.43%	13	6.19%
4	1.90%	14	6.67%
5	2.38%	15	7.14%
6	2.86%	16	7.62%
7	3.33%	17	8.10%
8	3.81%	18	8.57%
9	4.29%	19	9.05%
10	4.76%	20	9.52%

Case Study

Some packages that pass through a primary sorter cannot be routed to a secondary sorter for a variety of reasons and must be manually processed. Suppose such packages are routed to a circular conveyor as shown in Figure 16-4. Packages proceed around the conveyor to a workstation. There is no package waiting area or buffer at a workstation. If a package arrives to a station that is processing another package, it stays on the conveyor to the next station. If the package is not processed by the last station, it recirculates to the first station.

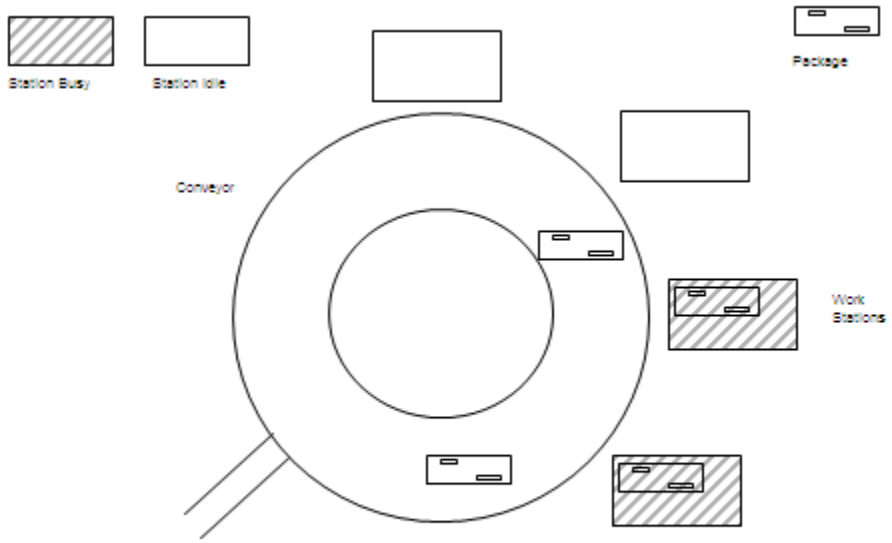


Figure 16-4: Circular Conveyor System

The purpose of the simulation study is to specify the parameters of the manual system to minimize package lead time. There may be either 1, 2, 3, or 4 workstatons employed. In addition, waiting areas for up to three packages may be placed in any fashion among the workstations. Cost considerations make more buffer spaces and fewer workstations the preferred design. Determine the number of workstations, the number of buffer spaces, and the location of the buffer spaces.

Relevant information is as follows:

Time between package arrivals:	Exponentially distributed with mean 1.6 minutes.
Package processing time:	Exponentially distributed with mean 4.0 minutes.

Conveyor Segments (Assuming a Four Workstation Configuration).

Conveyor Segment	**Conveyor Distance (Feet)**
Arrival Point to First Work Station Exit	18
Station Exit Segment	2
Inter-Station Segment (to Exit Segment)	13
Last Station to Arrival Point (4 stations case)	45

Assume that conveyor speed is 0.25 feet / second and that packages are 2 feet in length. The time period of interest is 40 hours.

Case Problem Issues

1. Count the number of possible alternatives. Is it reasonable to simulate all of these?

2. Which alternatives should be simulated to make sure the best or at least a good alternative is identified?

3. What performance measures in addition to package lead time are of interest?

4. What operating rules could be added to the system to guard against excessive lead times for individual packages?

5. What is the minimum number of workstations required by the system?

6. Discuss how verification and validation evidence can be obtained.

7. What is the purpose of having buffer space in front of workstation?

8. How is the arrival of a package to a workstation modeled if:

    a. There is no buffer space at the workstation.

    b. There is at least one buffer space at the workstation.

9. What is your initial guess as to the best placement for the buffer spaces? Does the simulation study confirm your guess?

10. Tell how to compute the lead time for a package as a function of the number times it travels completely around the conveyor within the simulation.

11. What is the radius of the conveyor: radius = circumference / 2 π ?

Chapter 17

Automated Guided Vehicle Systems

17.1 Introduction

An automated guided vehicle (AGV) system can transport material between a finite number of pre-defined locations at work stations with little or no human assistance. Barrett Electronics Corporation invented the world's first AGV for industrial applications in 1954. In the United States there have been over 3,000 AGV systems installed during the last 50 years. These systems range from one vehicle to well over 100 vehicles.

An AGV system consists of vehicles that move along predetermined paths to move loads between workstations and storage areas. Vehicles operate without the need for an onboard operator or driver, pick up loads at designated pick-up points and transport them to designated drop-off points. Each workstation has a pick-up point and a drop-off point. These two points can be the same.

There are several major categories of vehicles:

1. **Tow Type** vehicles that pull carts, trailers, dollies and the like.
2. Self-contained **Unit Load Type** vehicles that carry products on their built-in load decks.
3. **Fork Type** vehicles that utilize a fork/mast lift mechanism for interfacing with loads at various elevations.
4. Smaller **Commercial/Office Type** vehicles that have capacities of less than 500 pounds.
5. **Heavy Carrier Type** vehicles designed to transport large or very heavy loads such as dies, rolls, coils, ingots weighing in excess of 250,000 pounds.

Vehicles move between work stations by traversing control segments. Each control segment is relatively short. The intersection point between control segments is a control point. Pickup and dropoff points are control points as well.

Vehicles in most existing systems follow an inductive guide path consisting of a wire embedded in the floor carrying alternating current that induces a magnetic field detected by antenna mounted on the bottom of the vehicles. Other control mechanisms include surface mounted magnetic or optical strips as well as inertial or laser guidance. Vehicles have controllers that respond to instructions and ensure safety.

AGV systems must be able to perform routing, traffic control and communications functions. Routing is the method by which an AGV determines how to go from its current location to a designated destination. Different approaches to routing logic can be implemented such as shortest time, shortest distance and fixed pattern. Traffic control assures that AGVs do not collide with each other. Either fixed or variable distances between vehicles can be used.

Communication is needed between vehicles, between a vehicle and a central device or for local interfaces. The communication mechanism provides the means by which vehicles are informed of routing and traffic control decisions.

A simple AGV system is shown in Figure 17-1. There are four control segments that form a loop in the shape a rectangle with rounded corners. Rounded corners allow the AGV to continue at full speed instead of stopping to make a 90 degree turn as would be the case if square corners were used. There are four stations each with its own control point indicating the place where loads are picked up or dropped off. AGV's move in only one direction, clockwise, around the loop.

Requests come to move loads from one workstation to another. In response, an idle AGV moves from the parking area to the pickup point of the workstation where the load currently resides. The AGV moves from this pickup point to the drop off point of the destination workstation. After unloading, the AGV remains idle at the dropoff point.

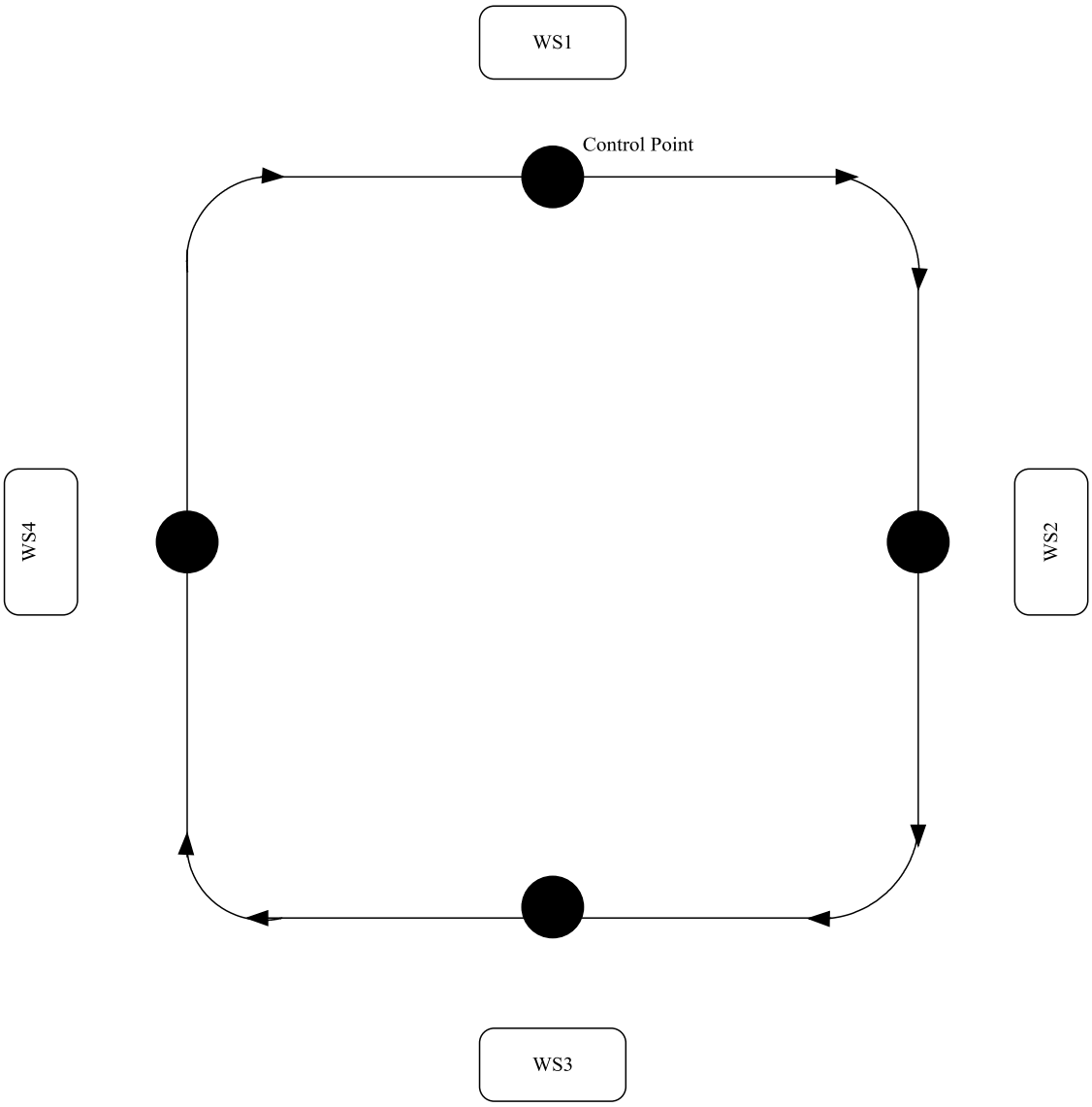


Figure 17-1: Simple AGVS Layout.

17.2 Points Made in the Case Study

Simulation can be used to assess the operational behavior of a system designed by other means. In this case, the AGV system is designed using standard, analytic methods. Simulation is used to assess the behavior of the system as designed relative to operational performance criteria as well as to determine the number of vehicles the system needs.

The structure of a system can be described using data inputs to a model. This allows changes in system structure to be assessed without changing the model. In this case, the control segments and control points are defined by input data. This data input most often takes the form of a graphical drawing.

The models originally developed for operations can be directly applied to material handling situations as well. This illustrates how models developed for one domain may be directly applied to another domain where system components behavior and interact in an analogous way. In an AGV system, the control points constrain the movement of the vehicles to assure that there are no collisions at interactions. Thus, control points can be modeled as a single machine station where the processing time is the time to traverse the control point.

Increasing the number of resources that can perform an activity, such as the number of machines at a workstation, normally lessens entity waiting time for that activity. Thus, it might be assumed that increasing the number of vehicles in an AGV system would increase the responsiveness to movement requests. This might not be the case since the contention between the AGV's for control segments, intersections, and control points will increase.

17.3 The Case Study[1]

The case study has to do with confirming the operational effectiveness of the design of a new AGV system as well as determining the number of vehicles needed. Load movement requests can be modeled as having a constant time between arrivals. However, the orgin and destination points are stochastic. That is not all requests for material movement can be predetermined. The discussion and examples of AGV systems in Askin and Standridge (1993) form the basis for this case study.

17.3.1 Define the Issues and Solution Objective

The design of a new AGV system to serve nine workstations as shown in Figure 17-2. Each shorter edge corresponds to 50 feet and each longer edge to 100 feet. AGV's move in one direction only on each bold edge as indicated by the arrows. There is no AGV movement on dashed edges. The letters in the center of a square are the work station ID's. The numbers near the edges are the control segment ID's. Idle AGV's wait where at the dropoff point of their last load.

The pickup and dropoff points for each workstation are indicated using the letters P and D respectively. Note that stations 5 and 6 share these points.

Table 17-1 gives the average number of material moves between workstations per 16 hour day. This information forms the distribution of pickup point to dropoff point AGV movements. Each individual movement can be determined as a random sample from this distribution. The time between material moves is a constant 90 seconds (57600 seconds per day / 640 moves).

A material move requires an AGV to move from it current location to the pickup point and then from the pickup point to the dropoff point. Each AGV moves at the rate of 5 feet per second and takes 30 seconds for each drop-off and each pick-up.

[1] Todd Frazee assisted with the development of this case study.

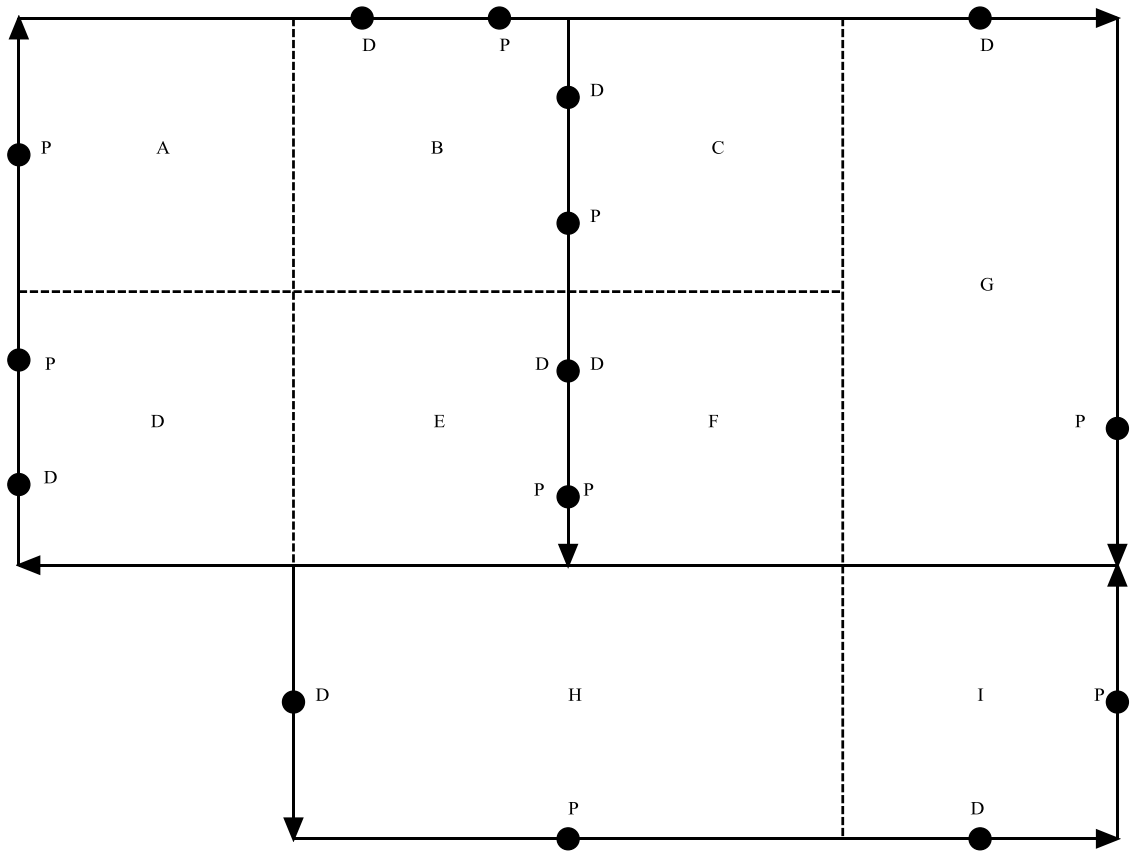


Figure 17.-2: AGV System Layout

The design shown in Figure 17-2 was developed using analytic methods. The following principles were applied.

1. Vehicles move in only one direction on path.
2. The dropoff point for a station should precede the pickup point with respect to vehicle movement.
3. Dropoff and pickup points should be placed on control segments with low utilization to avoid other vehicles waiting for dropoffs and pickups to be completed.
4. Movement of empty vehicles should be minimized. Thus after a dropoff is completed, the vehicle should wait on the same control segment for a possible pickup on that segment.

Other analytic methods were used to estimate that 2 AGV's would be needed in the system. These analytic methods were used to compute each of the five components of total vehicle utilization time: loaded travel time, travel time while empty, blocked time, load time, and unload time. These computations are based on knowledge of the number of loads to be moved between each pair of workstations (the information shown in Table 17-1) as well as AGV travel speeds and the shortest path between each pair of workstations.

Loaded travel time, load time, and unload time are straightforward to compute. A lower bound on travel time while empty can be computed using an optimization algorithm. Blocked time was assumed to be zero for this system since the number of AGV's need was only 2.

Table 17-1: Average Number of Material Moves between Work Stations

From Work Station	To Work Station	Average Number of Moves
A	B	40
A	C	25
A	D	30
A	E	10
A	F	10
A	G	20
A	H	5
A	I	10
B	C	40
B	E	30
B	G	10
B	H	10
C	G	50
C	I	10
D	B	5
D	C	10
D	F	10
E	D	100
F	D	60
G	F	40
G	I	40
H	D	10
H	F	5
I	E	60
Total		640

Management wishes to confirm the operationally effectiveness of AGV system as designed. The primary performance criteria is time between the request for a load to be moved and the completion of the move. Both the maximum and average time are interest. Assessing the number of AGV's needed is also important since blocked time was ignored and only a lower bound on travel time while empty was obtained. There is concern that 2 AGV's are not sufficient.

17.3.2 Build Models

It is helpful to take a generic perspective to modeling AGV systems. The control segments and control points that comprise the paths taken by the vehicles between workstations can be data input, expressed most often as a graphical drawing. In this case, the graphical drawing used for input this information is the one in Figure 17-2. Other inputs include where vehicles park when they become idle and vehicle speed. Vehicles can be viewed as resources. This generic view is implemented in some simulation environments.

In addition, a process model describing the movement of loads through the AGV system, perhaps including processing at workstations, is needed. A request to move a load is the entity flowing through the process. The following are the major steps in the process model.

1. Arrival of a request for an AGV to move a load from one workstation to another.
2. Waiting for an idle AGV.
3. Selection of the idle AGV nearest to the pickup point for the load.
4. Movement of that AGV from where it is parked to the pickup point.
5. Movement of the same AGV from the pickup point to the dropoff point.

Attributes of the entity are the following:

FromStation	The station where the load is to be picked up.
ToStation	The station where the load is to be dropped off.
ArriveTime	Simulation time that the request for load movement is made.

Analytic algorithms for determining the shortest path from one workstation to another are known and can be implemented within a simulation environment that supports modeling AGV systems. In most cases, the number of feasible paths between any pair of workstation should be few in number. Otherwise, the system would be too complex to operate. For example, consider the number of paths from workstation A in Figure 17-2 to each of the other eight workstations. There is only one path to workstations B, C, E, F, and G. There are two paths to the other workstations: D, H, and I. However, one of the two paths is obviously shorter.

One issue that is unique to modeling AGV systems is contention among the vehicles for the same control segment or control point. All vehicles travel at the same speed so one cannot overtake another as long as both are moving in the same direction. Contention occurs when one vehicle is stopped at a pickup or dropoff point and another vehicle needs to pass through such a point enroute somewhere else. In this case, the second vehicle needs to stop to wait for the first vehicle to leave the pickup or dropoff point.

In addition, contention can occur when two vehicles coming from opposite directions arrive at the same intersection at the same time. One vehicle needs to stop or slow down to let the other vehicle pass. There are two such intersections in the AGV system shown in Figure 17-2. One is at the right side of the boundary between workstations G and I. The other is at the center of the upper boundary of workstation H where the path dividing workstations E and F ends.

One system performance criterion is the time between the request for moving a load and completion of the move. Thus, it may seen desirable to have as many AGV's in the system as possible to minimize this time. This strategy is similar to increasing the number of machines at a workstation to minimize cycle time at the station that was employed in previous chapters. However, increasing the number of AGV's also increases the contention for control points and control segments. Thus, such increases may be counter productive and must be tested using simulation.

The modeling logic described above follows in pseudo English. AGV's are modeled as resources as are pickup and dropoff points. Each AGV has an attribute, CurrentLoc, giving its current location. Resources are also used to model intersections where vehicles can enter from more than one direction.

Travel along a path is comprised of a series of steps as modeled by Process MoveOnPath with parameters FromLoc and ToLoc. Each step represents travel between the current AGV location and the next pickup point, dropoff point, or intersection on the path. Each of these is modeled as resource that must be acquired to traverse that part of the path and freed after such movement is accomplished.

The next pickup point, dropoff point, or intersection and the distance to it are exacted from the data input describing the AGV system that was given as a graphical drawing. In this case, travel time can be modeled as distance traveled * AGV speed. It is possible to include acceleration and deacceleration if desired. When the destination control point is reached, travel ends. Otherwise travel to the next pickup point, dropoff point, or intersection commences.

The process AGV System makes use of the process MoveOnPath. Arrivals to the process are requests for load movement that occur every 90 seconds in this case. Entity attributes are assigned: the workstation where the load currently resides, the workstation to which the load

must be transported, and the simulation time the request arrives. The idle AGV closest to the workstation station where the load currently resides is chosen. If there are no idle AGV's the movement request must wait. The AGV moves empty workstation to the where the load is residing, picks of the load, moves to the destination station, and drops off the load. The AGV become IDLE and the current location of the AGV is recorded.

```
Define Resources
        AGV/2                           // Two AGV's
        ControlPointIntersection (n)/1  // Control Points and Path Intersections

Define Attributes
        FromStation                     // The station where the load is to be picked up.
        ToStation                       // The station where the load is to be dropped off.
        ArriveTime                      // Time the request for load movement is made.

Define Variables
        StartTrip (NStations)           // Distribution of trip starting point stations
        EndTrip  (NStations, NStations) //  Distribution of trip end point stations by starting station
        CurrentLoc(2)                   // Current location of an AGV

Process AGV_System
Define Arrivals
        Time of first arrival:      0
        Time between arrivals:      90 seconds
        Number of arrivals:         Infinite
Begin
        Set TimeArrive = Clock
        FromStation = Sample (StartTrip)
        EndStation = Sample(EndTrip(FromStation))
        Wait until AGV is IDLE in WaitforAGV    // IDLE AGV closest to From Station is chosen
        Make AGV Busy
        Send to MoveOnPath (CurrentLoc, FromStation) with return
        Wait for 30 seconds                     // Pick up load
        Send to MoveOnPath (FromStation, ToStation) with return
        Wait for 30 seconds                     // Drop off load
        Make AGV IDLE
        CurrentLoc (AGV) = ToStation
        Tabulate Clock – TimeArrive in CompleteMovementTime
End

Process MoveOnPath (FromLoc, ToLoc)
Begin
        While CurrentLoc(AGV) != ToLoc
        Begin
                CurrentLoc (AGV) = FromLoc
                Wait for Distance*AGVSpeed to
                        Next Control Point or Intersection from CurrentLoc
                CurrentLoc (AGV) = Next Control Point or Intersection
                Wait until ControlPointIntersection (CurrentLoc(AGV)) is IDLE
                Make ControlPointIntersection (CurrentLoc(AGV)) BUSY
                Wait for Distance through Control Point or Intersection * AGVSpeed
                Make ControlPointIntersection (CurrentLoc(AGV)) IDLE
        End
End
```

17.3.3 Identify Root Causes and Assess Initial Alternatives

The simulation experiment can be described as follows. The system operates for one 16 hour day. Thus, a terminating simulation of length one day is appropriate. The proper initial conditions are all AGV's idle since no load movement requests occur before the work day begins. Their initial location is randomly assigned. There is one random number stream to aid in selecting the pair of workstations for pickup and dropoff. Twenty replicates are made.

Management wishes to minimize the time to complete a movement request. Thus, performance measures include this quantity as well as the utilization of AGV's and AGV capacity lost to contention for control segments and control points. AGV congestion will be measured as the average number of AGV's waiting due to contention for control points and intersections.

The number of AGV's required must be determined, either the 2 previously recommend or 3 to improve the time to complete a movement request. The model parameter is the number of AGV's to employ. Table 17-2 summarizes the experimental design.

Table 17-2: Simulation Experiment Design for the AGV System

Element of the Experiment	**Values for This Experiment**
Type of Experiment	Terminating
Model Parameters and Their Values	1. Number of AGV's (2 or 3)
Performance Measures	1. Time to complete a move request 2. AGV utilization 3. AGV congestion
Random Number Streams	1. From-to pair of workstations
Initial Conditions	AGV's randomly assigned to control points
Number of Replicates	20
Simulated End Time	57600 seconds (one day)

Tables 17-3 through 17-5 give the simulation results for the above experiment, including a comparison between system operations when 2 and 3 AGV's are used.

Table 17-3: Simulation Results for Two AGV's

	AGV's		Time to Complete a Move (min)	
Replicate	**Idle Percent**	**Percent Congested**	**Maximum**	**Average**
1	0.7%	2.6%	80.9	8.7
2	1.4%	2.0%	60.8	5.9
3	0.5%	2.1%	114.6	15.5
4	0.4%	2.6%	92.3	11.5
5	1.6%	2.2%	99.1	9.8
6	1.4%	2.2%	71.0	6.8
7	1.3%	2.5%	88.8	9.0
8	0.6%	2.2%	100.6	10.0
9	1.2%	2.3%	75.9	8.4
10	0.4%	2.0%	120.2	14.4
11	2.8%	2.5%	43.9	5.1
12	0.5%	2.0%	105.5	13.0
13	0.4%	2.3%	94.9	11.3
14	1.9%	1.9%	45.4	5.5
15	1.9%	2.6%	60.7	6.0
16	2.5%	2.5%	24.9	4.4
17	1.2%	2.1%	125.3	14.3
18	1.2%	2.2%	52.1	5.8
19	0.9%	2.3%	93.2	12.9
20	0.4%	2.2%	79.7	10.4
Average	1.2%	2.3%	81.5	9.4
Std. Dev.	0.7%	0.2%	27.2	3.4
99% CI Lower Bound	0.7%	2.1%	64.1	7.2
99% CI Upper Bound	1.6%	2.4%	98.9	11.6

The following can be noted from Table 17-3 when 2 AGV's are used.

1. The AGV's are almost always busy.
2. There is very little congestion.
3. The average time to complete a move is 9.4 minutes with an approximate 99% confidence interval for the true average of (7.2, 11.6) minutes.
4. The maximum time to complete a move is over an hour with an approximate 99% confidence interval of (64.1, 98.9) minutes.

Thus it can be concluded from Table 17-3 that using only 2 AGV's is ineffective since the average time and maximum times to complete a move are too high. This is not unexpected since the AGV's are almost always busy. On the other hand, there is very little contention.

Table 17-4: Simulation Results for Three AGV's

	AGV's		Time to Complete a Move (min)	
Replicate	**Idle Percent**	**Percent Congested**	**Maximum**	**Average**
1	21.3%	12.8%	7.8	3.1
2	21.0%	12.0%	7.5	3.1
3	22.8%	11.7%	9.0	3.0
4	21.5%	11.5%	12.0	3.1
5	21.3%	12.1%	8.5	3.1
6	21.1%	12.0%	9.9	3.1
7	20.8%	11.8%	8.1	3.1
8	21.5%	11.5%	9.3	3.1
9	21.3%	12.0%	9.3	3.1
10	22.1%	11.5%	9.0	3.1
11	20.7%	12.6%	9.1	3.1
12	22.1%	10.9%	8.2	3.1
13	21.2%	12.0%	8.8	3.1
14	20.5%	12.6%	8.5	3.1
15	22.3%	11.4%	9.5	3.1
16	22.9%	11.8%	8.5	3.0
17	21.4%	11.8%	9.2	3.1
18	21.4%	12.0%	10.5	3.1
19	21.2%	11.8%	9.3	3.1
20	23.0%	10.5%	8.4	3.0
Average	21.6%	11.8%	9.0	3.1
Std. Dev.	0.7%	0.5%	1.0	0.04
99% CI Lower Bound	21.1%	11.5%	8.4	3.1
99% CI Upper Bound	22.0%	12.2%	9.7	3.1

The following can be noted from Table 17-4 when 3 AGV's are used.

1. AGV utilization is near 80%.
2. Significant congestion occurs since about 1/3 of the available time of 1 AGV is lost (11.8 % * 3 AGV's = 1/3 of 1 AGV).
3. The average time to move a load is about 3 minutes.
4. The maximum time to move a load is about (8.4, 9.7) minutes with approximately 99% confidence.

Thus it can be concluded from Table 17-4 that using 3 AGV's allows movement to occur in a sufficiently small amount of time. AGV utilization is neither too high or too low. However, contention among the three AGV's is significant.

Table 17-5: Comparison of Simulation Results for Two and Three AGV's

	AGV's (3-2)		Time to Complete a Move (min) (2-3)	
Replicate	**Idle Percent**	**Percent Congested**	**Maximum**	**Average**
1	20.6%	10.2%	73.1	5.6
2	19.6%	10.0%	53.3	2.8
3	22.3%	9.6%	105.6	12.5
4	21.1%	8.9%	80.3	8.4
5	19.7%	9.9%	90.6	6.7
6	19.7%	9.8%	61.2	3.7
7	19.5%	9.3%	80.7	5.8
8	20.9%	9.3%	91.3	6.9
9	20.1%	9.7%	66.6	5.3
10	21.7%	9.5%	111.2	11.4
11	17.9%	10.1%	34.8	2.0
12	21.6%	8.9%	97.3	9.9
13	20.8%	9.7%	86.2	8.1
14	18.6%	10.7%	36.9	2.4
15	20.4%	8.8%	51.3	2.9
16	20.4%	9.3%	16.4	1.4
17	20.2%	9.7%	116.2	11.1
18	20.2%	9.8%	41.6	2.7
19	20.3%	9.5%	83.9	9.8
20	22.6%	8.3%	71.4	7.4
Average	20.4%	9.6%	72.5	6.3
Std. Dev.	1.1%	0.5%	27.2	3.4
99% CI Lower Bound	19.7%	9.2%	55.1	4.1
99% CI Upper Bound	21.1%	9.9%	89.9	8.5

Table 17-5 shows that the difference between using 2 AGV's and 3 AGV's is statistically significant with approximately 99% confidence for all performance measures. AGV utilization is lowered when 3 AGV's are used as well as the average and maximum time to move a load. Congestion increases as well.

17.3.4 Review and Extend Previous Work

Management was pleased with the results of the simulation experiment. It was decided that three AGV's should be used.

The amount of contention between the three AGV's was of concern. It was felt that if load volumes increased and the addition of a fourth AGV was necessary that contention might cause the AGV system to take too long to respond to and complete transportation requests.

Thus, a redesign of the AGV system was proposed. The pickup and dropoff points for each workstation would be located within the station. Workstations E and F would have distinct pickup and dropoff points.

17.4 Assessment of Alternative Pickup and Dropoff Points

The impact of alternative pickup and dropoff points was assessed as follows.

17.4.1 Identify Root Causes and Assess Initial Alternatives

The assessment of the new AGV system design can be done in the following way. Note from Figure 17-2 that there are two types of workstations. The pickup and dropoff points for workstations B, C, D, E, and F are located near each other. The pickup and dropoff points for workstations G, H, and I are separate. Workstation A has only a pickup point.

Figure 17-3 shows the redesign of the pickup and dropoff points for workstations B, C, D, E, and F. Figure 17-4 shows the redesign for the remaining stations. Note that the AGV's have a greater distance to travel to both pickup and dropoff a load since a loop of about 15 feet must be traversed into each workstation.

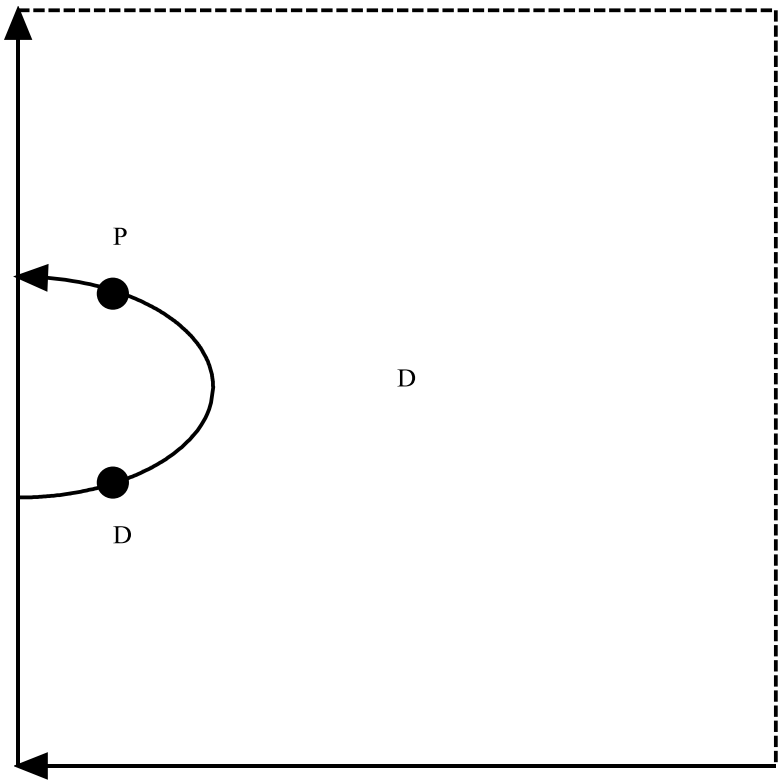


Figure 17-3: Example Workstation Layout with Pickup and Dropoff Points within the Workstation -- Style 1 .

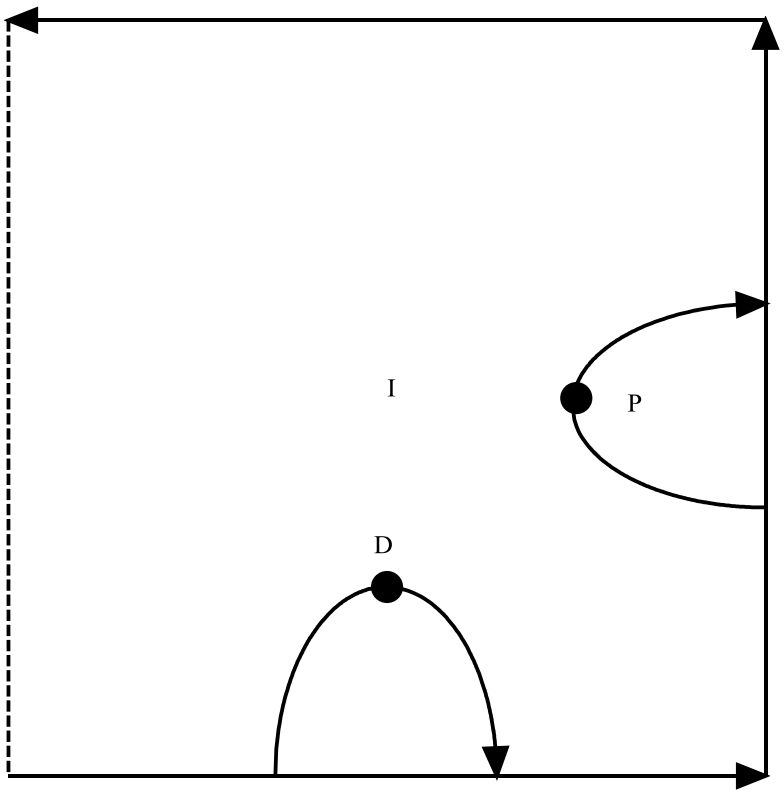


Figure 17-4: Example Workstation Layout with Pickup and Dropoff Points within the Workstation -- Style 2.

A new simulation experiment can be executed. The design is the same as shown in Table 17-2 except that the operation of the modified AGV layout with three AGV's only will be assessed. Note that the model does not need to be modified since the AGV layout is input data expressed as a graphical drawing. Simulation results for this experiment are shown in Table 17-6.

The following can be noted from Table 17-6.

1. AGV utilization is near 72%.
2. Only a little congestion occurs since about 13% of the available time of 1 AGV is lost.
3. The average time to move a load is about 3 minutes.
4. The maximum time to move a load is about (10.3, 28.1) minutes with approximately 99% confidence. The average maximum is 19.2 minutes. The range of the maximum times across the replicates is (5.3, 56.8) minutes.

Table 17-6: Simulation Results for Three AGV's with Dropoff and Pickup Points within Each Workstation

	AGV's		Time to Complete a Move (min)	
Replicate	**Idle Percent**	**Percent Congested**	**Maximum**	**Average**
1	26.4%	4.9%	3.3	38.3
2	27.6%	3.9%	2.8	10.0
3	28.1%	4.4%	2.8	12.2
4	28.6%	4.5%	2.9	33.2
5	27.9%	4.5%	2.8	5.3
6	28.4%	3.6%	2.8	12.5
7	28.6%	3.5%	2.7	5.3
8	28.2%	4.0%	2.9	23.3
9	28.6%	4.1%	2.9	32.9
10	28.3%	3.6%	2.8	16.3
11	28.3%	4.3%	2.9	22.4
12	26.6%	3.9%	3.2	32.5
13	27.3%	4.9%	3.5	56.8
14	28.4%	4.2%	2.8	11.5
15	28.4%	4.8%	2.8	18.6
16	28.4%	4.2%	2.7	7.5
17	28.3%	5.8%	2.9	25.5
18	28.5%	3.9%	2.7	7.8
19	27.6%	3.7%	2.8	5.4
20	27.8%	4.4%	2.8	6.8
Average	28.0%	4.3%	2.9	19.2
Std. Dev.	0.6%	0.6%	0.2	13.9
99% CI Lower Bound	27.6%	3.9%	2.8	10.3
99% CI Upper Bound	28.4%	4.6%	3.0	28.1

Table 17-7 contains a comparison of AGV system operations using the initial system design and the new system design each employing 3 AGV's.

Table 17-7. Comparison of Simulation Results for the Original and Modified System Designs

	AGV's (New-Original)		Time to Complete a Move (min) (New-Original)	
Replicate	**Idle Percent**	**Percent Congested**	**Maximum**	**Average**
1	5.1%	-7.9%	-4.5	35.2
2	6.6%	-8.1%	-4.8	6.9
3	5.3%	-7.3%	-6.3	9.2
4	7.1%	-7.0%	-9.1	30.1
5	6.6%	-7.6%	-5.8	2.2
6	7.3%	-8.4%	-7.1	9.4
7	7.8%	-8.3%	-5.4	2.2
8	6.7%	-7.5%	-6.4	20.2
9	7.3%	-7.9%	-6.4	29.8
10	6.2%	-7.9%	-6.2	13.2
11	7.6%	-8.3%	-6.2	19.2
12	4.5%	-7.0%	-5.1	29.4
13	6.1%	-7.1%	-5.3	53.7
14	7.9%	-8.4%	-5.7	8.3
15	6.1%	-6.6%	-6.7	15.5
16	5.5%	-7.6%	-5.8	4.5
17	6.9%	-6.0%	-6.2	22.3
18	7.1%	-8.1%	-7.8	4.7
19	6.4%	-8.1%	-6.5	2.3
20	4.8%	-6.1%	-5.6	3.8
Average	6.4%	-7.6%	-6.1	16.1
Std. Dev.	1.0%	0.7%	1.0	13.8
99% CI Lower Bound	5.8%	-8.0%	-6.8	7.2
99% CI Upper Bound	7.1%	-7.1%	-5.5	25.0

The following can be noted from Table 17-7.

1. AGV utilization increases as the average congestion increases for the new system configuration versus the original configuration. These difference are both significant with approximately 99% confidence. Note that both differences are small.
2. There is little difference in the average time to move a load between the two designs, though the difference is statistically significant.
3. The difference in the maximum time to move a load is operationally and statisically significant. The approximate 99% confidence interval is wide. The maximum difference is at least 29 minutes in 5 of 20 replicates.

17.4.2 Review and Extend Previous Work

Management rejected the new AGV system design. It was recognized that this design is more complex than the orginal which will require a more complex control system. AGV utilization and congestion as well as the average load delivery time were about the same for either design. The possible increase in maximum delivery time was of concern.

17.4.3 Implement the Selected Solution and Evaluate

The original system configuration with three AGV's will be implemented and the maximum time to move a load monitored.

17.5 Summary

The modeling and analysis of an AGV system design has been discussed in this chapter. The use of the graphical representation of the pathways traveled by the AGV's as data input to a simulation model has been presented. The conflict between improving response time to load movement requests and congestion by increasing the number of AGV's in the system has been examined. The need to confirm designs developed using analytic methods through simulation has been illustrated.

Problems

1. Compare and contrast the approach to modeling conveyors discussed in chapter 16 with the approach to modeling AGV systems presented in this chapter.

2. Tell why bi-directional AGV movement on a path is not desirable.

3. Tell why the dropoff point for a workstation should preceed the pickup point.

4. Visit a manufacturing facility and observe the automated material handling equipment that is in use.

5. Make a list of the automated material handling equipment you have observed in the service systems you encounter regularly.

6. List the advantages and disadvantages of adding additional AGV's to a system.

7. List the advantages and disadvantages of having distinct pickup and dropoff points at each workstation versus having a single pickup-dropoff point.

8. Consider the following modification to the original configuration with pickup and dropoff points on the main AGV path. All AGV's return to a parking area where maintenance and recharging is performed immediately after completing the movement of a load. Assess this design.

9. Consider the following modification to the new system configuration with pickup and dropoff points within each department. The pickup and dropoff points for each station are the same. Assess this design.

10. Reassess each design proposed in this chapter for the case where the time between request to move loads is exponentially distributed with mean 90 seconds.

11. Generate a customized trace of events and state variable values for the new design to determine why the maximum time to move a load sometimes becomes large.

Case Problem

Consider the following manufacturing system described in Askin and Standridge (1993) and shown in Figure 17-5. There are five departments. Material movement between departments is performed using an AGV system.

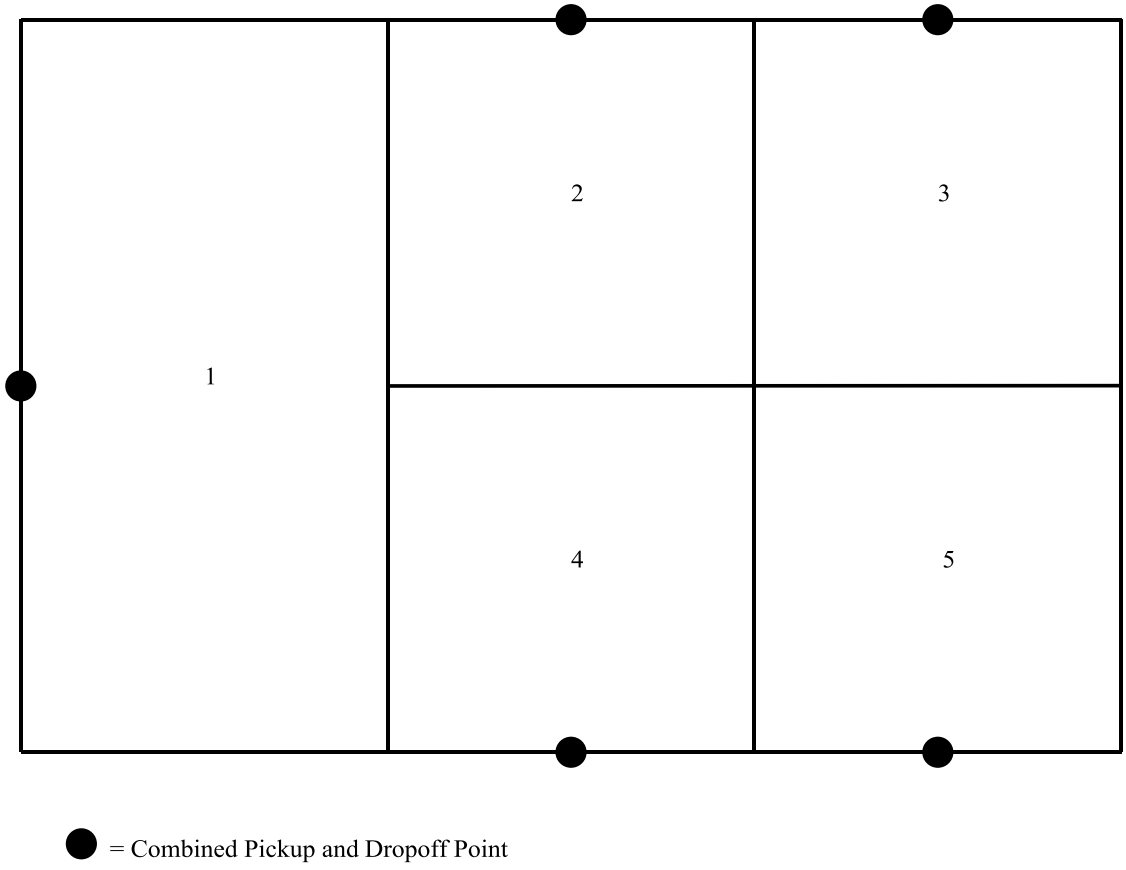


Figure 17-5: AVG Path for Application Problem

Material flow volumes between departments per eight hour day are shown in the following table.

Material Flow Volumes Between Departments per 8 Hour Day

From/To	1	2	3	4	5	Total From
1		10	25			35
2			10		25	35
3	15			10		25
4		40			20	60
5	24	10		50		84

Note that each department uses the same point for dropoffs and pickups. AGV travel time is 3 feet/second. Assume that the sequence of interdepartmental moves is essentially random but that the time between move requests can be modeled as a constant value.

Each department is 30 feet by 30 feet in size except department 1 which is 30 feet by 60 feet. Pickup and dropoff times are 15 seconds.

The problem is to determine the routes taken by AGV's between each pair of stations. In addition, the number of AGV's required in this system as well as the effectiveness of the system as measured by the time from the request to move a load until load movement is completed must be determined. Both the average and maximum times are of interest.

If the AGV system as designed proves ineffective, it may be improved by moving pickup and dropoff points. The redesign could include having separate pickup and dropoff point. A new AGV path could be defined as well.

The simulation study should answer the following questions:

1. Is one AGV sufficient for the current demand or are two AGV's necessary?
2. If demand increases uniformly across all stations by 20%, what adjustments to the system are necessary?

Case Problem Issues

1. Can the same model developed in this chapter be used as is or slightly modified to apply to the case problem? What would the modifications be?

2. What alternative AGV paths should be considered?

3. For each department, what pickup and dropoff point locations should be considered?

4. Are there any other performance measures besides load movement time and AGV utilization that could be important?

5. How will verification and validation evidence be obtained?

6. What is the expected number of AGV's required?

7. How will the arrival of load movement requests be modeled?

Chapter 18

Automated Storage and Retrieval

18.1 Introduction

Much of the time that material is in a plant, it is being moved or stored. In this chapter, the dynamics of how an automated storage and retrieval system (AS/RS) organizes and maintains an inventory are examined. An AS/RS system provides the following benefits: space efficient storage of materials, high speed controlled transportation of materials, and real-time inventory control. Thus an AS/RS system helps reduce inventory, labor, floor space, and material control costs.

A typical AS/RS system has several components as is shown in Figure 18-1. A storage / retrieval (S/R) machine places pallets (or another standard carrier) having one or more standard sizes in a high rise rack system. A rack consists of a matrix of storage locations. Racks are separated by aisles. There is one S/R machine per aisle. An S/R machine moves in the horizontal direction on a track located in the floor of an aisle and rises vertically via an imbedded mechanism. Typically, vertical speed is about 1/3 of horizontal speed.

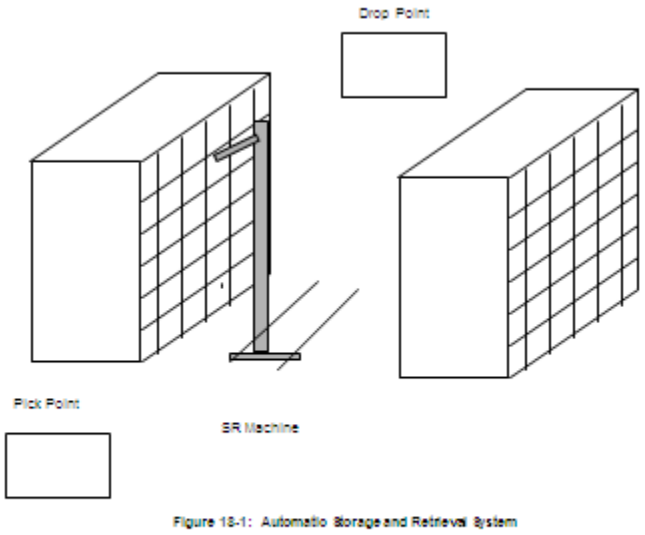


Figure 18-1: Automatic Storage and Retrieval System

Items to be stored arrive to a pick point. Retrieved items are transported by a the S/R machine to a drop point.

A computer-based control system is an important part of an AS/RS system. The computer keeps track of the exact location of all items in the racks. The control system directs the movement of the S/R machine by providing timely instructions concerning what items to retrieve or store in the racks. These instructions are in response to external requests for storage and retrieval.

The computer-based control system can be tested using simulation. Alternative rack sizes can be assessed. Various storage and retrieval strategies can be compared. In this way, movement of the S/R machine when it is empty, as well as the capital investment in racks, can be minimized.

18.2 Points Made in the Case Study

The control algorithm for an automated system, such as an AS/RS, can be included in a simulation model. The control algorithm may be coded in a general purpose programming language and interfaced within the model.

A simulation model can consist of multiple processes. These processes can share the same resources. In this application, an S/R machine, represented by a resource, is used by both an inventory storage process and by an inventory retrieval process.

A resource may have multiple BUSY states. Each BUSY state indicates that the resource is occupied in a unique way. In this application, each rack space is either empty or full of a particular type of item. BUSY states correspond to item types.

Sometimes it is necessary to select a resource to employ from a set of resources with similar or identical characteristics. A criteria for making the selection must be specified. The set of resources cannot be modeled as units of a single resource since the state of each individual resource must be tracked. In this application, the state of each individual rack storage space is important. The model must use the AS/RS control logic to select from among the storage spaces in the IDLE state when a carrier is stored. In the same way, the model must select from among items of the same type when a retrieval is required.

18.3 The Case Study

A particular manufacturing plant assembles finished goods from subassemblies that are produced in another area of the plant or delivered to the plant from external suppliers. A subassembly consists of component parts that have been joined together. Subassemblies arrive to the area preceding the final assembly operation as completed or as delivered.

Thus, a buffer before final assembly is required. The buffer is implemented using an AS/RS system. The storage area consists of two rectangular racks of bins with an aisle between them. Each bin holds one subassembly, which may be of one of four types. Subassemblies are delivered to a pick point where they are picked up one at a time by the S/R machine and placed in the nearest, with respect to S/R machine movement time, available bin.

The final assembly process requests subassemblies one at time. Each request specifies a particular type of subassembly. The S/R machine retrieves the nearest, with respect to its movement time, subassembly of the requested type and places it at the drop point. The subassembly is subsequently moved from the drop point to the final assembly area.

To minimize unproductive movements, the S/R machine remains at the bin in which it last placed a subassembly or at the drop point when it completes a task and becomes idle.

Subassemblies arrive from 6:00 A.M. to 2:00 P.M. each day. The final assembly process operates from 8:00 A.M. to 4:00 P.M. each day or until all of the subassemblies in the AS/RS have been consumed.

18.3.1 Define the Issues and Solution Objective

A fundamental issue in the AS/RS control algorithm is into what free bin to store a subassembly and from what occupied bin to retrieve a subassembly. The algorithm to select a bin is an intrinsic component of the operation of the AS/RS system and must be included in the simulation model. Each bin is in one of nine states:

1. Idle
2-5. Occupied with a subassembly of type one, two , three, or four
6-9. Occupied with a subassembly of type one, two, three, or four that is committed to the final assembly process

The selected bin is the one in the specified state that requires the least travel time for the S/R machine. The idle S/R machine waits at the pick point or at the last bin in which a subassembly was stored.

The S/R machine moves 6 feet per second horizontally and 2 foot per second vertically. Each bin is 1 foot square including the rack structure. Thus, the time to reach any bin is the sum of the number of bins traversed horizontally * 1/6 second per bin and the number traversed vertically * 1/2 second per bin. This sum is illustrated for a rack 8 bins high and 7 bins long in Figure 18-2 assuming the S/R machine starts at the pick point which is to the left of the bin structure on the floor level.

The search for a bin is performed by the AS/RS control software. Bins are searched in order of the movement time values shown in Figure 18-2, least to greatest until a bin in the state desired is located. Among bins with the same value, those closer to the floor are preferred.

The same search strategy can be applied if the S/R machine is waiting at a particular bin. The control algorithm searches in four directions, one at a time. These directions are:

1. Right and up from the current location, as shown in Figure 18-2.
2. Right and down from the current location.
3. Left and up from the current location.
4. Left and down from the current location.

After all the searches have been completed, the storage location nearest the S/R machine with respect to movement time is chosen.

The search strategy is worthy of discussion. Consider the movement time of $7/6^{th}$ second. This is the movement time to the seventh bin in the first row, the fourth bin in the second row, and the first bin in the third row. Thus, the bin search order for the time $7/6^{th}$ second is as listed previously.

Consider searching up and right from the current SR machine location in general. Bins are examined in order of movement time, least to greatest, until a bin in the desired state is found. Bins with equal movement times are searched as follows. The search begins at the bin to the right of the current location and procedes to the bin in the next higher row and three columns preceding (since the vertical movement time is three times the horizontal movement time). This part of the search stops when either a bin in the desired state is found or the next bin to be examined would be to the left of the current location of the SR machine or the next bin to be examined does not exist.

It takes the S/R machine 6 seconds to store or retrieve a subassembly from a bin. The time between requests to store a subassembly is 20 seconds, exponentially distributed, as is the time between requests to retrieve a subassembly.

Two configurations of the AS/RS system have been proposed. In the first, each rack has 180 bins, 10 bins high and 18 bins long. In the other, each rack has 225 bins, 9 bins high and 25 bins long. Thus, extra storage space requires more floor space. The problem is to select between these two alternatives.

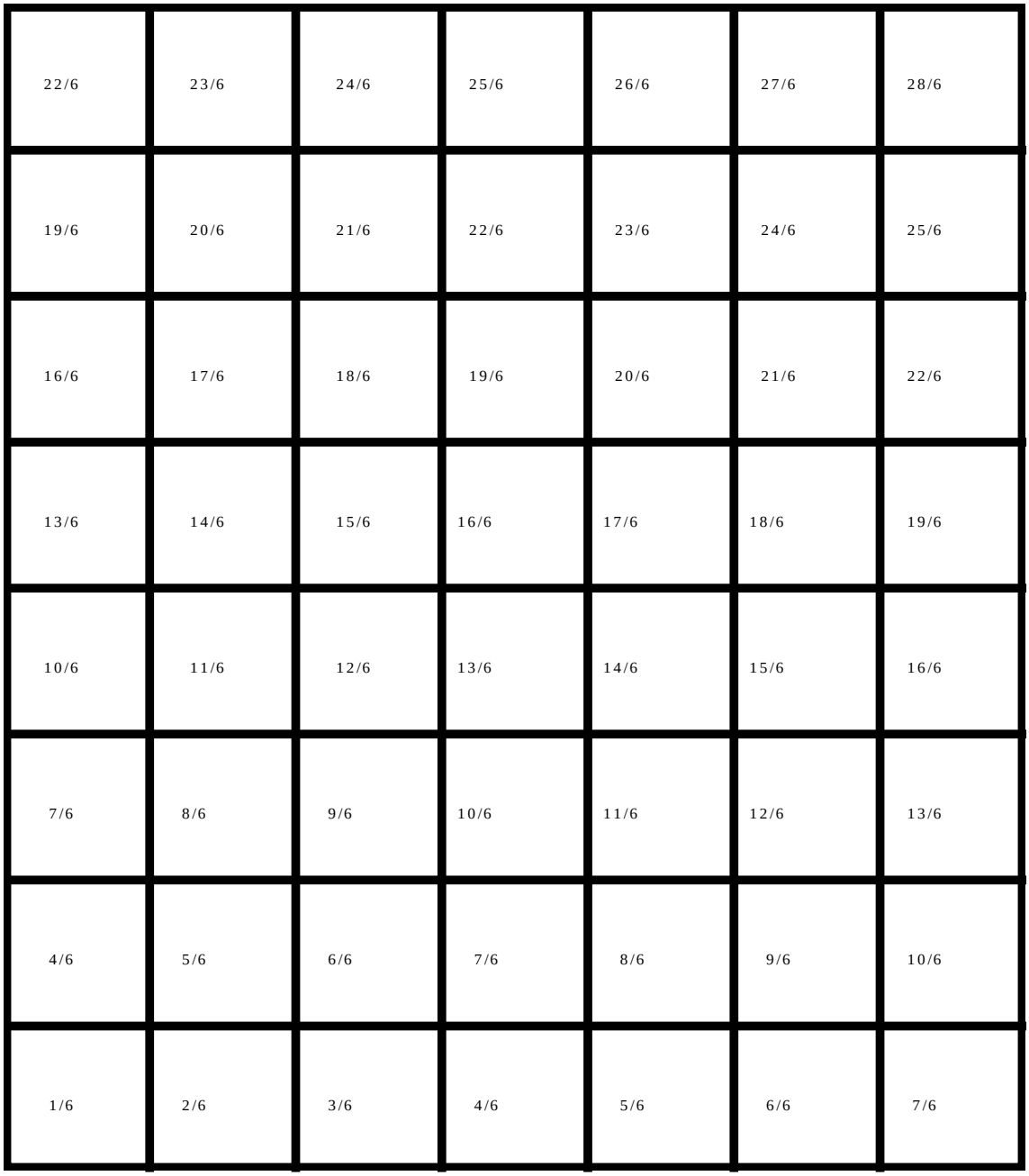


Figure 18-2: S/R Machine Movement Time (Seconds)

18.3.2 Build Models

The two operations performed by the AS/RS system are modeled as two separate processes. The first operation stores a subassembly in a bin. The second retrieves a subassembly from a bin.

Entities represent subassemblies to be stored or retreived and have five attributes:

Type =	Type of subassembly: 1, 2, 3, 4.
ArriveTime =	Time of arrival to the AS/RS system.
Rack =	Rack in which to store the subassembly: 1 or 2.
Row =	Horizontal position of the bin in which to store the subassembly.
Column =	Vertical position of the bin in which to store the subassembly.

A state variable is used to track the state of each bin:: idle, filled with a subassembly of a particular type, or filled with a subassembly of a particular type that is committed to the second manufacturing process. In addition, there is a state variable for each subassembly type modeling the number of units of that type in the racks.

A resource represents the S/R machine. The resource modeling the S/R machine has two attributes indicating its Row and Column location in the rack structure.

The storage of an arriving subassembly is handled as follows. The subassembly waits at the pick point until at least one bin is idle. Which particular idle bin to use is determined by the AS/RS control algorithm which is implemented in the model. Information identifying the location of the bin is recorded in the attributes (Rack, Row, Column) of the subassembly entity.

The subassembly continues to wait until the S/R machine is idle. The S/R machine moves from its current location to the pick point. The S/R machine picks up the subassembly, moves to the selected idle bin, and stores the subassembly in that bin. The S/R machine waits at that bin for its next assignment.

Finally, the state of the system updated. The state of the bin is changed to the type of subassembly stored in the bin. The location of the S/R machine is recorded in its Row and Column attributes. The number of subassemblies of the type just stored is incremented by one.

The process of retrieving a subassembly from a bin is similar to the storage process just described. The request for a subassembly of a particular type waits until there is a subassembly of that type in the AS/RS. The AS/RS system control algorithm selects the bin closest to the current location of the S/R machine containing a subassembly of the desired type. The S/R machines moves to that bin, retrieves the subassembly and procedes to the drop point. The S/R machine becomes idle and remains at the drop point.

Again, the state of the system is updated. The state of the bin from which the subassembly was retrieved is changed to idle. The number of subassemblies of the type just retrieved is decremented by one. The location of the SR machine is recorded.

When it becomes idle, the SR machine resource may need to choose between two jobs: storing a subassembly or retrieving a previously stored one. Management decided that it was most important to keep the second manufacturing process working. Thus, priority is given to requests to retrieve previously stored subassemblies.

The AS/RS control algorithm is shown in Figures 18-3 a, b, and c.

```
function SearchOne
/* routine to search in a given direction for a bin in a given state
   inputs:
     RowDir      = Row Direction (1 or -1)
     ColDir      = Column Direction (1 or -1)
     RowStart    = Row Location of First Cell
     ColStart    = Column Location of First Cell
     ColDiff     = Number of columns to move before moving rows
     RowMax      = Number of Rows in a Rack
     ColMax      = Number of Columns in a Rack
     Rack        = Rack ID number (1 or 2)
     TargetState = Required State of Location
     LocState    = State of each rack
   outputs:
     Row = Row of required bin
     Col = Column of required bin  */
/* Search Equivalent Bins */
    Row = 0
    Col = 0
    RowIndex = RowStart
    ColIndex = ColStart
    RowBase  = RowStart
    ColBase  = ColStart
/* Stay within the boundaries of a rack */
   while RowIndex <= RowMax and RowIndex > 0 do
   begin
      while ColIndex <= ColMax and ColIndex > 0 do
      begin
/* Stay within the boundaries of the search direction from the
   starting point */
       while (RowIndex <= RowMax and RowIndex > 0)   and
                    (ColIndex <= ColMax and ColIndex > 0)   and
                    ((RowIndex >= RowStart and RowDir > 0)   or
                     (RowIndex <= RowStart and RowDir < 0)) and
                    ((ColIndex >= ColStart and ColDir > 0)   or
                     (ColIndex <= ColStart and ColDir < 0))  do
       begin
                  if(LocState(Rack,RowIndex,ColIndex) = TargetState) then
                  begin
                 /* Bin in desired state found.  Set attributes */
                     Row = RowIndex
                     Col = ColIndex
                     return
                  end
                 /* Go to next bin having same movement time */
                  RowIndex = RowIndex + RowDir
                  ColIndex = ColIndex - ColDir*ColDiff
       end
   /* Go to bin with next smallest movement time in the initial row */
          RowIndex = RowBase
          ColBase  = ColBase + ColDir
          ColIndex = ColBase
      end
  /* Go to bin in the next row with the next smallest movement time */
      RowBase  = RowBase + RowDir
      RowIndex = RowBase
      ColBase  = ColStart + ColDir*ColDiff
      ColIndex = ColBase
   end
end
```

Figure 18-3a: Control Algorithm Function for Searching in One Direction

```
begin S_SearchRack
/* routine to search for a storage state in a given state in a given
direction in each of two racks
   inputs:
     RowDir      = Row Direction (1 or -1)
     ColDir      = Column Direction (1 or -1)
     RowStart    = Row Location of First Cell
     ColStart    = Column Location of First Cell
     ColDiff     = Number of columns to move before moving rows
     RowMax      = Number of Rows in a Rack
     ColMax      = Number of Columns in a Rack
     TargetState = Required State of Location
     LocState    = State of each rack
   outputs:
     RackA = Rack ID of the required bin
     Row   = Row of required bin
     Col   = Column of required bin  */
/* Search the first rack */
     Rack  = 1
     RackA = 1
     call S_SearchOne
     RowTemp = Row
     ColTemp = Col
/* Seach the second rack */
     Rack  = 2
     RackA = 2
     call S_SearchOne
/* See if the location in the first rack is closer than the one in the
second rack */
/* return second rack info if no bin was found in the first rack */
     if(RowTemp = 0 or ColTemp = 0) then return
/* return first rack info if no bin was found in the second rack */
     if(Row     = 0 or Col     = 0) then
     begin
       Row   = RowTemp
       Col   = ColTemp
       RackA = 1
       return
     end
/* bin found in both racks; return info for bin with shorter movement
time */
     if(abs(RowTemp-RowStart)*RowSpeed+abs(ColTemp-ColStart)*ColSpeed<
        abs(Row    -RowStart)*RowSpeed+abs(Col    -ColStart)*ColSpeed)
     then
     begin
/* movement to rack one bin is shorter */
       Row   = RowTemp
       Col   = ColTemp
       RackA = 1
       return
     end
end
```

Figure 18-3b: Control Algorithm Function for Determining the Shortest Move Time Among Two Racks

The control algorithm is implemented as three functions. The first searches from the current location of the SR machine given by RowStart and ColStart in any of the four directions given above as specified by RowDir and ColDir. Each of these two variables takes on the values –1 or +1 to define the search direction. The dimensions of a rack are specified in the variables RowMax and ColMax. Which of the two racks to search is specified in the variable Rack which has the value 1 or 2. The variable TargetState gives the state of interest, zero for idle or 1, 2, 3, or 4 for a subassembly of that type. The state of each bin is stored in the three dimensional array LocState(Rack, Row, Col). The variable ColDiff stores the ratio of the horizontal speed to the vertical speed of the SR machine which is three in this case.

The entity attributes Row and Col store the location of the bin in the desired state. If no such bin is found both Row and Col have the value zero.

The other two functions use the same variables as SearchOne. The function SearchRack, shown in Figure 18-3b, searches each rack in one of the directions listed above for a bin and returns the location of the bin that is closest with respect to movement time to the current position of the SR machine. The rack ID number (1 or 2) is returned in the entity attribute RackA.

The function SearchAll, shown in Figure 18-3c, searches in all four directions from the current SR machine location to find the nearest bin in the desired state. The directions are searched one at a time using function SearchRack. After each search, the nearer location so far is determined. The nearest location is returned using the entity attributes Row, Col, and RackA.

The model also contains two processes, Arrival and Retrieval, whose steps were previously described. Pseudo-code for the two processes follows. The same variables defined above for the function SearchOne are also used in the processes. Note that in the Arrival process, the function SearchRack is used instead of SearchAll since the only search direction is up and right of the pick point.

```
begin SearchAll
/* routine to search for a storage state in each of two racks in all
directions
   inputs:
     RowStart    = Row Location of First Cell
     ColStart    = Column Location of First Cell
     ColDiff     = Number of columns to move before moving rows
     RowMax      = Number of Rows in a Rack
     ColMax      = Number of Columns in a Rack
     TargetState = Required State of Location
     LocState    = State of each rack
   outputs:
     RackA = Rack ID of the required bin
     Row = Row of required bin
     Col = Column of required bin  */
/* Search right and up */
     RowDir = 1
     ColDir = 1
     call S_SearchRack
     RowTemp1 = Row
     ColTemp1 = Col
     RackTemp = RackA
/* Search right and down */
     RowDir = 1
     ColDir = -1
     call S_SearchRack
/* Select which is closer */
    if(RowTemp1 = 0 or ColTemp1 = 0) then
    begin
        RowTemp1 = Row
        ColTemp1 = Col
        RackTemp = RackA
    end
    else
    begin
        if((abs(RowTemp1-RowStart)*RowSpeed+
            abs(ColTemp1-ColStart)*ColSpeed >
            abs(Row     -RowStart)*RowSpeed+
            abs(Col     -ColStart)*ColSpeed)   and
            (Row > 0 and Col > 0)) then
        begin
            RowTemp1 = Row
            ColTemp1 = Col
            RackTemp = RackA
        end
    end
```

Figure 18-3c: Control Algorithm Function for Determining the Shortest Move Time in Any Direction

```
/* Search left and up */
    RowDir = -1
    ColDir = 1
    call S_SearchRack
/* Select which is closer */
    if(RowTemp1 = 0 or ColTemp1 = 0) then
    begin
        RowTemp1 = Row
        ColTemp1 = Col
        RackTemp = RackA
    end
    else
    begin
        if((abs(RowTemp1-RowStart)*RowSpeed+
            abs(ColTemp1-ColStart)*ColSpeed >
            abs(Row     -RowStart)*RowSpeed+
            abs(Col     -ColStart)*ColSpeed)   and
            (Row > 0 and Col > 0)) then
        begin
            RowTemp1 = Row
            ColTemp1 = Col
            RackTemp = RackA
        end
    end
/* Search left and down */
     RowDir = -1
     ColDir = -1
     call S_SearchRack
/* Select which is closer */
    if(RowTemp1 = 0 or ColTemp1 = 0) then
    begin
        RowTemp1 = Row
        ColTemp1 = Col
        RackTemp = RackA
    end
    else
    begin
        if((abs(RowTemp1-RowStart)*RowSpeed+
            abs(ColTemp1-ColStart)*ColSpeed >
            abs(Row     -RowStart)*RowSpeed+
            abs(Col     -ColStart)*ColSpeed)   and
            (Row > 0 and Col > 0)) then
        begin
            RowTemp1 = Row
            ColTemp1 = Col
            RackTemp = RackA
        end
     end
/* return closest location */
     Row   = RowTemp1
     Col   = ColTemp1
     RackA = RackTemp
end
```

Figure 18-3c: Concluded

```
Define Resources
        SRMach              // Storage / retrieval (S/R) machine

Define Attributes
        ArriveTime          // Time of arrival of subassembly
        Type                // Type of subassembly

Define Variables
        Bin                 // Bins currently available
        InvSA1              // Subassemblies of type 1
        InvSA2              // Subassemblies of type 2
        InvSA3              // Subassemblies of type 3
        InvSA4              // Subassemblies of type 4
        RowStart            // Current location of S/R machine – row
        ColumnStart         // Current location of S/R machine – column
        RowDirection        // Direction of row search
        ColumnDirection     // Direction of column search
        TargetState         // State of requested bin
        Rack                // Rack with bin in requested state
        Row                 // Row of bin in requested state
        Column              // Column of bin in requested state
        VerticalSpeed       // Vertical (column) speed of S/R Machine
        HorizontalSpeed     // Horizontal (row) speed of S/R Machine
        LocState            // State of each bin
        ColumnMax           // Number of columns in a rack

Process SubAssembly_Arrivals
Define Arrivals
        Time of first arrival:   0
        Time between arrivals:   Exponential 20 seconds
        Number of arrivals:      Infinite
Begin
        Set ArriveTime = Clock
        Set Type = Integer Uniform (1, 4)
        Wait until Bin > 0
        Increment Bin by 1
        Wait until SRMachine is IDLE
        Make SRMachine BUSY
        Wait for RowStart*VerticalSpeed + ColumnStart*HorizontalSpeed
                // Move SRMachine to Pick Point
        Set RowStart = 1
        Set ColumnStart = 1
        Set RowDirection = 1
        Set ColumnDirection = 1
        Set TargetState = 0
        Call SearchRack returning Rack, Row, Column
        Wait for Row*VerticalSpeed + Column*HorizontalSpeed
                // Move SRMachine to Selected Bin
        Wait for 6 seconds      // Store Carrier in Bin
        Make SRMach IDLE
        LocState (Rack, Row, Column) = Type
        Increment InvSA<Type> by 1
End
```

```
Process SubAssembly_Retrievals
Define Arrivals
        Time of first arrival:      2 hours
        Time between arrivals:  Exponential 20 seconds
        Number of arrivals:         Infinite
Begin
        Set ArriveTime = Clock
        Set Type = Integer Uniform (1, 4)
        Wait until InvSA<Type> >0
        Decrement InvSA<Type> by 1
        Set TargetState = Type
        Call SearchAll returning Rack, Row, Column
        Set LocState (Rack, Row, Column) = Type +4
        Wait until SRMachine is IDLE
        Make SRMachine BUSY
        Wait for abs ((Row – RowStart)*VerticalSpeed) +
                abs ((Column-ColumnStart)*HorizontalSpeed
                // Move SRMachine to Select Bin
        Wait for 6 seconds                  // Remove Carrier from Bin
        Wait for abs (Row-1)*VerticalSpeed + abs (Column-ColumnMax)*HorizontalSpeed
                // Move SRMachine to drop point
        Make SRMach IDLE
        Set RowStart = 1
        Set ColumnStart = ColumnMax
        Set LocState(Rack, Row, Column) = 0
End
```

–

18.3.3 Identify Root Causes and Assess Initial Alternatives

Table 18-1 gives the design for the AS/RS system simulation experiment. The final assembly process consumes all subassemblies stored in the rack each day. Thus, a terminating experiment with the simulated time interval equal to the time each day that the subassemblies arrive to the AS/RS system is appropriate. The dynamics of how the final assembly process consumes the subassemblies remaining in the storage racks after all subassemblies have arrived will not affect the choice of configurations. Thus, this part of the system need not be included in the experiment.

Table 18-1: Simulation Experiment Design for the AS/RS System

Element of the Experiment	**Values for This Experiment**
Type of Experiment	Terminating
Model Parameters and Their Values	Rack configuration -- (10X18 and 9X25)
Performance Measures	1. Number of bins in non-IDLE states 2. Time subassemblies wait for a bin 3. Time subassemblies and final process requests wait for the SR machine.
Random Number Streams	1. Type of subassembly to store 2. Time between arrivals of subassemblies to the AS/RS system 3. Type of subassembly requested by final assembly 4. Time between arrivals of requests from final assembly
Initial Conditions	The bins empty and the SR machine idle
Number of Replicates	20
Simulation End Time	One eight hour day (time in seconds)

The initial conditions are the daily start-up conditions for the system: all bins empty and the SR machine idle. Twenty replicates will comprise the experiment. There are four random number streams, one each to determine the type of subassembly delivery to the AS/RS and requested by the final assembly process as well as one each for the time between arrivals of subassemblies and requests from the final assembly process.

The model parameter is the rack configuration with the two alternatives proposed by management tested. Performance measures have to do with the utilization of bins, subassembly waiting time for an empty bin, and waiting time for the SR machine to move subassemblies.

Results of this experiment are shown in Table 18-2. The average percent of bins occupied is 16% less for the 9 X 25 rack configuration with an approximate 95% confidence interval of 15% to 18% for the true percent difference. The 9 X 25 rack is 25% bigger than the 10 X 18 rack. Thus, some use is made of the extra bin space. This is reflected in the fact that there is no waiting for an empty bin when the larger rack is used. However, the average waiting time for the smaller rack is only 2.5 seconds with an approximate 95% confidence interval of 0.7 to 4.3 seconds for the true mean waiting time. The average waiting time for the SR machine increases when the larger rack size is used, though the average difference is only 2.4 seconds.

Table 18-2: Results of the AS/RS System Simulation Experiment

	Percentage of Full Bins			Average Time Waiting for an Empty Bin (Seconds)			Average Time Waiting for the SR Machine (Seconds)		
Replicate	**10 X 18**	**9 X 25**	**Diff**	**10 X 18**	**9 X 25**	**Diff**	**10 X 18**	**9 X 25**	**Diff**
1	70%	92%	22%	0	8.3	8.3	9.4	6.8	2.6
2	72%	92%	20%	0	8.8	8.8	9.4	6.7	2.7
3	71%	92%	20%	0	8.2	8.2	9.5	6.7	2.8
4	67%	93%	27%	0	8.7	8.7	9.6	6.7	2.9
5	65%	92%	27%	0	8.5	8.5	9.3	6.7	2.6
6	72%	92%	19%	0	8.2	8.2	9.4	6.7	2.7
7	68%	94%	26%	0	8.8	8.8	9.5	6.7	2.7
8	72%	93%	21%	0	8.8	8.8	9.5	6.7	2.9
9	73%	91%	18%	0	8.9	8.9	9.5	6.7	2.9
10	70%	93%	23%	0	8.5	8.5	9.5	6.7	2.8
11	68%	92%	25%	0	8.8	8.8	9.6	6.6	3.0
12	69%	92%	24%	0	8.4	8.4	9.5	6.8	2.7
13	77%	91%	14%	0	8.2	8.2	9.4	6.7	2.7
14	68%	92%	24%	0	8.3	8.3	9.5	6.7	2.8
15	63%	94%	31%	0	9.0	9.0	9.6	6.7	2.9
16	70%	92%	22%	0	8.2	8.2	9.6	6.7	2.9
17	68%	93%	25%	0	8.9	8.9	9.5	6.7	2.9
18	74%	93%	18%	0	8.5	8.5	9.3	6.7	2.6
19	71%	92%	22%	0	8.7	8.7	9.4	6.7	2.8
20	65%	92%	27%	0	8.3	8.3	9.6	6.7	2.9
Average	70%	92%	23%	0	8.5	8.5	9.5	6.7	2.8
Std. Dev.	3%	1%	3.9%	0	0.28	0.28	0.088	0.043	0.10
99% C.I. Lower Bound	67%	92%	20%	0	8.4	8.4	9.4	6.7	2.7
99% C.I. Upper Bound	72%	93%	25%	0	8.7	8.7	9.5	6.7	2.8

18.3.4 Review and Extend Previous Work

The smaller rack configuration seems preferable since it would be less costly and require less floor space as long as system performance is not significantly improved by using the larger rack. While the larger rack does eliminate waiting for an empty bin, it increases waiting time for the SR machine. However, no waiting time is very long for either configuration. The bin utilization is relatively high for both rack sizes (92% versus 70%).

18.3.5 Implement the Selected Solution and Evaluate

The AS/RS system with the 10 X 18 configuration of two racks of bins will be implemented. Subassembly waiting at the pick point will be monitored and sufficient buffer space provided as needed.

18.4 Summary

This case study shows how system operational algorithms can be included in models. The use of modeler defined resource states is included. Inventories and other resources are shared between processes in the model. The simulation experiment compares alternative system configurations.

Problems

1. Based on the process steps in the simulation model, tell why the bin states: occupied with a subassembly of type one, two, three, or four that is committed to the final assembly process are necessary.

2. Validate the function SearchOne by searching from rack location row 2 column 2 as shown in Figure 18-2 in the right and up direction. List the first ten values of RowIndex and ColIndex computed in SearchOne, including the infeasible bin location values that cause the loops in SearchOne to end.

3. In the Arrival process model, why is it not necessary to check if the function SearchRack located a bin in the IDLE state?

4. Tell why the quantity: Number of final assembly process requests waiting for a subassembly is not an effective performance measure for the simulation experiment in this chapter.

5. What impact would running the simulation experiment until all subassemblies had been moved to the final assembly process have on the validity of the performance measure estimates?

6. Explain why the average waiting time for the SR machine increases when the larger rack size is used especially considering that there is no waiting for an empty bin.

7. Would you expect the utilization of the SR machine to increase or decrease when the larger rack size is used? Justify your answer.

8. Use Little's Law to estimate the average number of subassemblies waiting at the pick point. How much buffer space would you use at the pick point?

9. Compute the expected time to store a carrier in a bin after the SR machine is obtained.

10. Visit a manufacturing facility and observe the automated material handling equipment that is in use.

11. Make a list of the automated material handling equipment you have observed in the service systems you encounter regularly.

12. How much improvement is there in the AS/RS system if the speed of the SR machine increases by 100%.

13. How much improvement is there in the AS/RS system if the time between requests from the second manufacturing process is uniformly distributed between 10 and 30 seconds?

14. Perform additional simulation experiments to find the smallest difference between the starting time of the storage process (currently 6:00 A.M.) and the retrieval process (currently 8:00 A.M) for which the system can effectively operate.

15. The current rack configurations are about one story high. Suppose a two story high configuration was preferred, specifically 18 bins high and 10 bins wide. Compare system performance using this configuration to the 10 bins high and 18 bins wide configuration.

16. Embellish the model in this chapter with acceleration and deacceleration of the SR machine. Assume the acceleration (deacceleration) distance is one bin in either direction and the average time to traverse this bin is twice that of other bins.

Case Problem

The benefits of AS/RS technology have been effectively realized in libraries. The amount of floor space required for books and periodicals has been reduced by ten-fold or more. The number of librarians required was reduced as well. Reshelving errors were eliminated. The location of each item while in the library is known with certainty. Despite these benefits, it is estimated that a few (less than 12) mini-load AS/RS systems have been installed in libraries.

This case problem involves determining the saturation point for a mini-load AS/RS system installed in a particular library. This is done be creating a graph of the cycle time for retrieving a book or periodical versus the arrival rate for such requests. The arrival rate resulting in the longest acceptable retrieval time is the saturation point. The smallest arrival rate of interest is 10 requests per hour. Assume that the arrival rate for retrievals is the same as the arrival rate for returns.

The mini-load AS/RS system installed in one particular library has a capacity of 250,000 books and periodicals. There is a single aisle with identical racks on each side. The system is installed inside a secured vault for safety and security reasons.

Books and periodicals are stored in carriers that are 4 feet deep and 2 feet wide. Each carrier row is one of three heights: 10, 12, or 15 inches. Each item is stored in the shallowest carrier in which it can stand. Thus, vertical space is used most efficiently. Assume that the number of books and periodicals of each height is the same.

There are 36 carrier rows on each side of the single aisle. The height of the first row is 10 inches, the second 12 inches, the third 15 inches, the fourth 10 inches and so forth. There are 60 carriers in each row.

The S/R machine travels at a high rate of speed: 12.6 feet/second horizontally and 4.3 feet/second vertically. Assume that the S/R machine must travel either horizontally or vertical but not diagonally.

The process of retrieving a book or periodical is the following. A patron makes a request using the electronic library catalog system. The AS/RS fills one request at a time. The location of the

item is completely random. The S/R machine moves from its idle location to the required carrier, extracts the carrier in 3 seconds, and places the carrier in the pick and delivery station. A librarian must remove the desired item from the carrier and record its status in the information system. This takes 7 seconds. The S/R machine remains idle at the pick and delivery station.

Next the librarian determines whether any item that needs to be returned to storage is of the same size as the carrier. If so, the item's new carrier location is recorded in the information system and the item placed in the carrier. Both steps combined take 7 seconds.

Assume the library is open 16 hours per day, 7 days per week.

Embellishment: The AS/RS system tests the carrier for weight restrictions. One in 100 tests fail. In this case, the librarian must remove the item as well as the newly entered location from the information system in 7 seconds. In either case, the S/R machine replaces the carrier and returns empty to its idle location.

Embellishment: Find the saturation point when the following procedure is used. The S/R machine does not replace a carrier that is at a pick and delivery station until the next retrieval request is made. At that time, a carrier is first stored and then the next carrier retrieved.

Embellishment: Limit the number of carriers stored at the pickup/dropoff station to a total of three. When the fourth carrier arrives, it is immediate returned to the same storage location by the AS/RS machine.

Case Problem Issues:

1. How should carriers be modeled?

2. How should the location of the carrier containing the book or periodical requested be determined?

3. How should S/R machine travel time be computed?

4. Specify the process for book and periodical returns.

5. What are good initial conditions for this simulation experiment?

6. What performance measures, other than cycle time, would be of interest?

7. What is the expected utilization of the SR machine?

8. How should verification and validation evidence be obtained?

AutoMod Summary and Tutorial for the Chapter 6 Case Study

A.1. Introduction

AutoMod modeling constructs and experimental specifications generally needed for modeling arrivals, operations, and detractors such as rework, downtime, and setup / batching are presented. Example models illustrating routing and inventory dynamics are given as part of the application studies. A tutorial gives step-by-step instructions for building and simulating the model associated with the single workstation case study in Chapter 6.

A.2. AutoMod Modeling Elements

The application studies use primarily AutoMod modeling elements defined in Table A-1.

Table A-1: AutoMod Modeling Elements

Modeling Element	Definition
Process	The steps used to model entity processing at a workstation as well as upon arrival or departure
Loads	Entities
Attributes	Entity attributes
Resources	Resources
Resource Cycles	The pattern of state changes of a resource due to the breakdown and repair cycle
Counters	Resource-like variables used to model inventories
Queues	Buffers or waiting areas
Order Lists	A list of loads. Loads remain on the list until ordered to leave.
Variables	State variables used throughout a model such as parameters of a processing time or characteristics of a resource
Tables	The collection mechanism for performance measure observations not automatically maintained by AutoMod
Random Streams	Pseudo-random number streams

In AutoMod, loads (entities in the text) flow through one or more processes. A process is described by a set of statements. AutoMod has many statements. Table A-2 describes some of the commonly used statements. A complete definition of each statement is provided in the AutoMod help system along with examples.

The user needs to be aware of one quirk in AutoMod, whick expects models to have a visual component. Thus, entities must always be where they can be displayed graphically. For right now, this place is in a queue. Thus, while an entity is being processed by a resource, it must be in a queue. Thus, a single queue preceding a resource will contain the loads in the buffer as well as the loads in processing that is all the loads at the workstation. Alternatively, the user can employ one queue to represent the buffer where entities wait for a resource and a second queue to represent where an entity is graphically while it is being processed by the resource. The former approach will be used in this tutorial.

Table A-2: Commonly Used Statements

Statement	Definition
begin	Start of a process or of a block of statements
end	End of a process or of a block of statements
set	Assign a value to a variable or attribute as well as changing the state or number of units of a resource or the value of a counter
send to	Send an entity to the start of another process
tabulate	Record the value of a performance measure (observed type)
clone	Create copies of an entity and send the copies to a process
move into	Enter a queue
wait for	Time delay for a process step
wait until <condition>	Delay until the condition (logical expression) becomes true
get	Acquire one or more units of a resource that are in the idle state. Same as: wait until <resource> is idle; make <resource> busy
free	Free one or more units of a resource placing them in the idle state. Same as: make <resource> idle
increment	Add to the value of a variable or attribute as well as increasing the number of units of a resource or the value of a counter
decrement	Subtract from the value of a variable or attribute as well as decreasing the number of units of a resource or the value of a counter
wait to be ordered	Enter an order list
order	Send one or more loads on an order list to a process

while <condition> do
begin
end
	While loop.

A-3. Tutorial – Model Building

This section shows how to build the single workstation model as specified in the chapter 6 case problem in AutoMod step-by-step.

1. Start AutoMod as you would any windows program.
2. Choose FILE from the menu bar and then NEW. Specify the location you want for the model files in the directory structure.
3. Design the model.
   a. Decide what processes are necessary. In this case, use three processes: one for entity arrival, one for entity departure, and one for the workstation.
   b. Decide what attributes are necessary. In this case, arrival time is sufficient.
4. Define the arrival process. By convention, process names begin with P_. Choose PROCESS from the process system menu and then NEW. Give the name of the process (P_Arrive is good) and enter a title as documentation.
5. Select EDIT arriving procedure and the text editor appears. The statements for P_Arrive can be entered.
   a. Enter **begin** on the first line and **end** on the second line to delimit the procedure. Insert a comment line after the first line to describe the procedure. Comments start with //. Comments may be placed on the same line as statements.
   b. The procedure P_Arrive must accomplish two things. The first thing is assigning the value of the time between arrivals load attribute to the arrival time: **set A_ArriveTime = ac**, where ac is the current simulation time (absolute clock).
   c. The second thing is to send the arriving entity to the process for the workstation: **send to P_WSA**.
   d. Terminate the edit using FILE then SAVE and FILE then EXIT. Notice that AutoMod will object that the load attribute (A_ArriveTime) as well as the workstation process (P_WSA) have not as yet been defined. The strategy that

we are using is to define them at this point. In the error box for A_ArriveTime, choose define and load attribute. In the attribute definition box, enter the name and a title for documentation as well as the type as **real**. In the error box for P_WSA, choose define and process and then simply hit return to take all of the defaults.

e. In the Edit a Process window, select OK.

6. Next choose PROCESS from the process system menu and edit P_WSA in the same way that P_Arrive was created. The procedure must accomplish the following.
   a. Enter the buffer of the workstation: move into **Q_WS**
   b. Acquire the workstation resource: **get R_WS**
   c. Perform processing: **wait for RS_WS uniform 7.5, 1.5 min**
   d. Free the workstation resource: **free R_WS**
   e. Send the load to the process for departing entities: **send to P_Depart**
7. Next choose FILE then SAVE and FILE then EXIT. Note that one queue, one resource, one random number stream, and a process must be defined. Define a queue by specifying its name, a title, and capacity. The capacity of Q_WSA is INFINITE.
   a. Define a resource by specifying its name, a title, and default capacity (number of units), in this case one.
   b. Define a random number stream by specifying its name: RS_WS.
8. P_Depart must accomplish the following.
   a. Observe entity time in the system: **tabulate (ac – A_ArriveTime) in T_LeadTime**
   b. Destroy the entity: **send to die**.
9. Choose File then SAVE and FILE then EXIT.
   a. A table is defined by specifying its name and a title.
10. Define the load type for parts. From the process system menu, select Loads and then select New for a new load type. Name the load L_Part.
    a. Next select New Creation to specify the arrival process for loads.
    b. Specify the time between arrivals as exponentially distributed with a mean of 10 minutes.
    c. Specify the first arrival at time 0: Constant 0 in the First One at field.
    d. Specify the first process as P_Arrive.
11. Define the load type for initial parts at the workstation at the start of the simulation. From the process system menu, select Loads and then select New for a new load type. Name the load L_InitPart.
    a. Next select New Creation to specify the arrival process for loads.
    b. Specify the number of creations to be 3.
    c. Specify the time between arrivals as a constant 0 so all the parts arrive at time 0
    d. Specify the first arrival at time 0: Constant 0 in the First One at field.
    e. Specify the first process as P_Arrive.
    f. Modify P_Depart so that data is not collected on the parts initially in the system, where **type** is a built-in load attribute: **if type = L_Part tabulate (ac – A_ArriveTime) in T_LeadTime**
12. Specify the length of the run as 168 hours. Select Run Control and new. Specify the snap (replicate) length as 168 hours.
13. Save the model.
14. Export the model: File/Export
15. Use the zip utility to create a zip file containing the exported (archive) version of the model: Programs/AutoMod/Utilities/Model Zip and select the model archive.

Note: The exported version of the model is a condensed version of the model suitable for sending by email. This is the version of the model that should be submitted.

A-4. Tutorial – Model Execution

The model can be run as follows.

1. Select RUN and then RUN MODEL.
2. The model will be compiled and a new window opened.
3. In the new window, select CONTROL and CONTINUE to run the simulation.
4. To make the model run faster, turn off animation: CNTL-G.
5. At the end of the run (or during the run), examine the reports for Processes, Queues, Resources, and Tables using VIEW and then REPORTS.
6. Use the information in the reports to obtain verification evidence.

A-5. Tutorial – Modeling Extension

Next close the execution window and return to the model. Save the model under a new name so that the modifications to follow are kept distinct from the original model.

The first modification is to model setup and batching at the workstation using the logic described in chapter 6. First determine the batch size using the computations in chapter 6. The enter setup and batching into the model as follows:

1. Modify P_Arrive to create a batch. Whenever the total number of arrivals to P_Arrive (**P_Arrive total**) is a multiple of the batch size, a batch is created. Thus, when a load arrives, test whether or not this condition if met. The expression: **P_Arrive total % V_Batchsize** will be zero when a P_Arrive total is a multiple of the batch size. Recall that % is the remainder operator.
   a. If it is NOT met: **wait to be ordered on OL_BatchList // hold load on batch list**
   b. If it is met: **send to P_WSA**
2. Save and exit. Define the order list OL_BatchList by giving its name and description.
3. Modify P_WS to process a batch. Between get R_WS and free R_WS, add the following
   a. Wait for the setup time: **wait for 45 min**
   b. Use a while <condition> do loop to model processing each item in the batch individually
      i. **set V_LoopIndex = 0**
      ii. **while V_LoopIndex < V_BatchSize do**
      iii. **begin**
      iv. **wait for RS_WS uniform 7.5, 1.5 min**
      v. **increment V_LoopIndex by 1**
      vi. **end**
4. After free R_WS, send each individual load to P_Depart:
   a. order (V_BatchSize-1) loads from OL_BatchList to P_Depart
5. Save the model.

The second change is to add rework of a part to the model. This requires a little thought since loads in P_WS represent batches not parts. Here is one way this can be accomplished. Incrementing V_LoopIndex means that the part successfully completed. Thus, incrementing V_LoopIndex with the probability of completing a successful part would model part rework.

If RS_Rework uniform 0.5, 0.5 > 0.05 then increment V_LoopIndex by 1
// 0.05 is the probability that a part needs rework

The third change in the model involves a downtime repair cycle. Your tasks are as follows:

1. Create a new resource cycle and name it C_Bdown. Select Resources and then New for resource cycles. Select OK, edit to create the resource cycle.
2. Select MTTF/MTTR and fill in the required information.
3. Edit the resource WS to attach the resource cycle.
4. Save the model.

Follow the directions in IV above to make sure the model works by obtaining verification evidence.

A-6. Tutorial – Conducting Experiments with AutoStat

AutoStat is the component of the AutoMod simulation environment that is used to conduct simulation experiments. AutoStat is used after the model is built as well as verified and validated using the graphical execution component.

Start AutoStat from the build component menu: RUN, Run AutoStat. The AutoStat setup wizard will ask several questions. Answers can be modified later by selecting Properties from the menu bar. In answer the setup wizard questions, use the following information.

1. The model is random.
2. Answer no to the second question.
3. The model does not require warm-up.
4. The snap length is 168 hours.
5. It is fine to have the method of common random numbers as the default method.

Next conduct a simulation experiment as follows:

1. Define a new analysis of type single scenario.
2. In the pop-up box, give the analysis a name, specify 20 replications. Next select: OK do these runs.
3. Next from the main AutoStat window, select new responses to extract from the simulation runs the performance measure statistics of interest. In this case, select the mean lead time. This is done by choosing Table as the AutoMod entity and mean as the statistic of interest. A name should be specified as well. This step can be repeated for all performance measures of interest, such as utilization and maximum lead time.
4. View the performance measure values by selecting Analyses from the main AutoMod window and then the Run Results item under the name of the analysis of interest.
5. Copy the results to an Excel spreadsheet from the window where the run results are displayed. Select Edit/Copy Entire Table. In Excel, select Edit / Paste Special / Unicode Text.

One through five above should be done for each model, the original workstation model and the one with detractors

6. Analyze the simulation results using Excel. Create three columns: Replicate number (1-20), Lead Time for Original, Lead Time with detractors. Use the Excel function Transpose to place the simulation results in the proper column. Compute the difference in cycle time replicate by replicate in a fourth column. Compute summary statistics and t confidence intervals as appropriate. Use the Excel function TINV to return the appropriate critical values from the Student's t distribution with n-1 degrees of freedom.

A-7. Initialization of State Variables

Initialization of state variables, that is setting the value of a counter or a resource capacity (number of units of the resource) before the simulation begins, is important in some models. This is accomplished using the model initialization function, which AutoMod automatically executes before a model is simulated. There is at most one model initialization function per model.

A model initialization function is created as follows:

1. Select Source Files from the Process System panel.
2. Select New
3. For name, use logic.m
4. Select edit to open the editor.

The following example illustrates how to use the model initialization function. Assume the variables have been defined and given initial values in their definitions.

```
begin model initialization function

// Set the value of counter to target inventory value
// Note the current attribute of the counter must be referenced
        set C_Inventory current = V_TargetInventory

// Set the capacity of a resource (number of units) to the number of machines at a station
        set R_Station capacity = V_MachinesAtStation

        return true        //AutoMod requirement
end
```

A-8. Creating a Trace File in Comma Separated Value (.csv) Format

Consider the model of a single workstation with no detractors as described in section III above. Suppose a trace of all state changes: from idle to busy as well as from busy to idle is desired. This trace is to be written to a user defined comma separated value (.csv) file that can be opened in Excel. In the file, columns are delimited by commas. Every time Excel sees a comma, the following information is placed in the next column to the right. As well, such files can be opened in editors, like Notepad, in which the contents of the file including the commas can be seen.

The following example shows how to open .csv file in the model initialization function and write the column headers to the file.

```
begin model initialization function

// open the trace file; note that the variable V_TraceFile is of type file ptr (pointer)
// by Automod convention, the file will reside in the \arc directory for the model
        open "StateTrace.csv" for writing save result as V_TraceFile

// write the header to the trace file
        print “Clock, New State” to V_TraceFile

        return true        //AutoMod requirement
end
```

Column values can be written in a similar way whenever desired. For example, the print statement to write the state change to busy to the trace file is as follows:

print ac, ", Busy" to V_TraceFile

A-9. Choose between Two Resources

Suppose an operation can be performed by either of two resources, R_MachineA or R_MachineB. The first resource with one unit in the idle state will be used. If both are available R_MachineA will be use. The following process fragment shows how to accomplish this. Note that A_Machine is load attribute of type resource ptr (resource name).

```
wait until R_MachineA remaining > 0 or R_MachineB remaining > 0  // wait for a machine
if R_MachineA remaining > 0 then
begin
        set A_Machine    = R_MachineA          // Machine A is available
end
else
begin
        set A_Machine    = R_MachineB          // Only Machine B is available
end

get A_Machine  // get selected machine
wait for 15 min  // perform operation
free A_Machine // free selected machine
```

Distribution Function Fitting in JMP: Tutorial

B.1 Introduction

JMP is a general purpose data analysis software tool that includes fitting distribution functions to data. This tutorial leads the reader through a data fitting exercise for version 9 of JMP. Steps of the tutorial are shown in *italics*.

B.2. Procedures for Fitting Data to Distributions

Start up JMP in the usual way for a Windows program.

Select View / JMP Starter

Within JMP Starter, *Select New Data Table.*

Within New Data Table, *Select File / Open to load the file with the data to be fit. The file is a .txt file.* The data in the file will appear in a spreadsheet- like table.

Next select Basic from the category column.

Next select Distribution. Click in the box to the right of: Y, columns. Then double click on column 0. Then select OK.

A box appears containing statistical summaries of the data set. Examine these carefully.

Next see how well the data fits a normal distribution. *Click the arrow next to the column label 0. Select Continuous Fit then normal distribution.* Look at the normal distribution superimposed on the histogram.

Next test the fit. *Click the arrow next to Fitted Normal. Select Goodness of Fit.* Note that the fit to a distribution is not adequate.

Let go back and re-examine the data values. Assume that a zero value represents a no ship condition and that we are interest in the distribution of the volume shipped given that shipments were made. Let's eliminate the zero values and refit the distribution. *Select the first six rows in the data table by selecting the row numbers 1 through 6. Select the arrow next Rows and then Exclude / Unexclude.*

Repeat the above process for fitting a distribution function to the data.

In addition, repeat all of the above for the gamma distribution. Which fits better in your opinion, the normal or the gamma?

Bibliography

Askin, R. G. & Standridge, C. R. (1993). *Modeling and analysis of manufacturing systems*. John Wiley and Sons, New York.

Askin R.G. & Estrada, S., (1999). Investigation of cellular manufacturing practices. *Handbook of cellular manufacturing systems*, S. Irani, ed., John Wiley & Sons, Inc., New York.

Askin, R. G. & Goldberg, J. B. (2002). *Design and analysis of lean production systems,* John Wiley & Sons, New York.

Balci, O. (1994). Validation, verification, and testing techniques throughout the life cycle of a simulation study. *Annals of Operations Research*, 53:121-173.

Balci, O. (1996). Principles of simulation model validation, verification, and testing. *International Journal in Computer Simulation.*

Banks, J., Carson II, J. S., Nelson, B. L. & Nicol, D.M. (2009). Discrete-event system simulation, 5th ed**.** Prentice Hall. Englewood Cliffs, NJ.

Bozer, Y. A., & White, J. A. (1984). Travel-time models for AS/RS. *IIE Transactions*, 16(4), 329-338.

Bozer, Y. A., & White, J. A. (1996). A generalized design and performance analysis model for end-of-aisle order-picking systems. IIE Transactions, 28(4), 271-280.

Buzacott, J. & Hanifin, L. (1978). Models of automatic transfer lines with inventory banks: a review and comparison. *AIIE Transactions*, 10, 197-207.

Carson II, J. S. (2002). Model verification and validation. *Proceedings of the 2002 Winter Simulation Conference***.** Paper presented at the 2002 Winter Simulation Conference: San Diego, California. Retrieved June 12, 2010 from http://www.informs-sim.org/wsc02papers/008.pdf

Conway, R., Maxwell, W., McClain, J. O., & Thomas, L. J. (1988). The role of work-in-process inventory in serial production lines. *Operations Research*, 35(2), 291-305.

Devore, J. L. (2008). *Probability and statistics for engineering and the sciences, 7th ed.,* Duxbury Press, Belmont, CA.

Duinkerken, M. B., Ottjes, J. A., & Lodewijks, G. (2006). Comparison of routing strategies for AGV systems using simulation. Proceedings of the 2006 Winter Simulation Conference. Paper presented at the 2006 Winter Simulation Conference: Monterey, California. Retrieved September 12, 2011 from http://www.informs-sim.org/wsc06papers/193.pdf

Elsayed, E. A. & Unal, O. I. (1989). Order batching algorithms and travel-time estimation for automated storage / retrieval systems. *International Journal of Production Research*, 27(7), 1097-1114.

Ekren, B. Y. & Heragu, S. S. (2009). Simulation based regression analysis for rack configuration of autonomous vehicle storage and retrieval system. Proceedings of the 2009 Winter Simulation Conference, Paper presented at the 2009 Winter Simulation Conference: Austin, TX. Retrieved October 16, 2011 from http://www.informs-sim.org/wsc09papers/232.pdf.

Flanigan-Wagner, M. A. & Wilson, J. R. (1995). Graphical interaction simulation input modeling with bivarate bezier distributions. *ACM Transactions on Modeling & Computer Simulation*, 5(3), 163-189.

Flanigan-Wagner, M. A. & Wilson, J. R. (1996). Using univariate bezier distributions to model simulation input processes. *IIE Transactions*, 28(9), 699-712.

Ferrin, D. M., Miller M. J., & Muthler D. (2005). Lean sigma & simulation, so what's the correlation? *Proceedings of the 2005 Winter Simulation Conference*. Paper presented at the 2005 Winter Simulation Conference: Orlando, Florida. Retrieved June 5, 2010 from http://www.informs-sim.org/wsc05papers/249.pdf.

Gorman, M. F., Hoff, J. & Kinion, R. (2009). Tales from the front: case studies indicate the potential pitfalls of misapplication of lean improvement programs. *Interfaces* 39 (6).

Grimard, C., Marvel, J. H. & Standridge, C. R. (2005). Validation of the re-design of a manufacturing work cell using simulation. *Proceedings of the 2005 winter simulation conference.* Paper presented at the 2005 Winter Simulation Conference: Orlando, Flordia. Retrieved September 9, 2010 from http://www.informs-sim.org/wsc05papers/170.pdf.

Han, M.-H., McGinnis, L. F., Shieh, J. S. & White, J. A. (1987). On sequencing retrievals in an automated storage / retrieval system, *IIE Transactions*, 19(1), 56-66.

Hopp, W. J. & Spearman, M. L. (2007). Factory physics: foundations of manufacturing management, 3rd edition, McGraw-Hill/Irwin, New York.

Hyden, P., Roeder, T., & Schruben, L. (2001). Resource graphs for modeling large-scale, highly congested systems. *Proceedings of the 2001 winter simulation conference***.** Paper presented at the 2001 Winter Simulation Conference: Arlington, Virginia. Retrieved September 9, 2010 from http://www.informs-sim.org/wsc01papers/068.PDF.

Irani, S. A., Subramanian, S., & Allam, Y. S. (1999). Introduction to cellular manufacturing systems. *Handbook of cellular manufacturing systems*, S. Irani, ed., John Wiley & Sons, Inc., New York.

Jing, G. G., Kelton, W. D., Arantes, J. C., & Houshmand, A. A. (1998). Modeling a controlled conveyor network with merging configuration. Proceedings of the 1998 Winter Simulation Conference, Paper presented at the 1998 Winter Simulation Conference: Washingon, DC. Retrieved September 27, 2011 from http://www.informs-sim.org/wsc98papers/142.pdf.

Jones – Lang- Lasalle. (2008). Lean practices in the supply chain. Retrieved May 8, 2011 from http://www.joneslanglasalle.com/Documents/JLL-LeanPracticesInSupplyChain.pdf

Keller, G. (2001). *Applied statistics with Microsoft Excel*, Duxbury Press, Belmont, CA.

Koo, L. Y., Chen, Y., Adhitya, A., Srinivasan, R., & Karimi, I. A. (2006). Evaluating refinery supply chain policies and investment decisions through simulation-optimization. Proceedings of the 2006 Winter Simulation Conference. Paper presented at the 2006 Winter Simulation Conference: Monterey, California. Retrieved September 12, 2011 from http://www.informs-sim.org/wsc06papers/181.pdf

Law, A. M. (2007). Simulation modeling & analysis, 4th edition, McGraw-Hill, New York.

Law, A. M. & McComas, M. G. (2001). How the EXPERFIT distribution fitting software can make your simulation models more valid. *Proceedings of the 2001 Winter Simulation Conference.* Paper presented at the 2001 Winter Simulation Conference: Arlington, VA.

Law, A. M. & McComas, M. G. (1996). EXPERFIT: total support for simulation input modeling. *Proceedings of the 1996 Winter Simulation Conference.* Paper presented at the 1996 Winter Simulation Conference: Coronado, California.

Learnsigma. (2007). What is lean manufacturing? Retrieved February 13, 2009 from http://learnsigma.wordpress.com/2007/11/10/what-is-lean-manufacturing/.

Lehmer, D. H. (1951). Mathematical methods in large-scale computing units. *Annals of the Computing Laboratory of Harvard University*, 26,141-146.

Liu, R, Kumar, A., & Stenger, A. J. (2006). Simulation results for supply chain configurations based on information sharing. Proceedings of the 2006 Winter Simulation Conference. Paper presented at the 2006 Winter Simulation Conference: Monterey, California. Retrieved September 12, 2011 from http://www.informs-sim.org/wsc06papers/076.pdf

Marvel, J., & Standridge, C. (2009). A Simulation-enhanced lean design process. *Journal of Industrial Engineering & Management*, 2(1), pp 90-113. Retrieved June 5, 2010 from http://www.jiem.org/index.php/jiem/article/viewFile/61/18.

Marvel, J.H., Schaub, M.A., & Weckman, G.R. (2008). Assessing the availability & allocation of production capacity in a fabrication facility through simulation modeling: a case study. *International Journal of Industrial Engineering*, 15(2), 166-175. Abstract retrieved June 5, 2010 from http://ijietap.utep.edu/ojs/index.php/ijie/article/view/117.

Material Handling Industrial Association. (2011). Automated Guided Vehicle Systems Industry Group. Retrieved October 2, 2011 from http://www.mhia.org/industrygroups/agvs.

Miller, G., Pawloski, J. & Standridge, C. (2010). A Case Study of Lean, Sustainable Manufacturing. *Journal of Industrial Engineering & Management*, 3(1), 11-32. Retrieved September 27, 2011 from http://www.jiem.org/index.php/jiem/article/viewFile/156/50.

Mittal, S. & Wang, H.-P. (1992). Simulation of JIT production to determine number of kanbans. *International journal of advanced manufacturing technology*, 7, 292-308.

Nazzal, D. & McGinnis, L. F. (2006). An analytical model of vehicle-based automated material handling systems in semiconductor fabs. Proceedings of the 2006 Winter Simulation Conference. Paper presented at the 2006 Winter Simulation Conference: Monterey, California. Retrieved October 2, 2011 from http://www.informs-sim.org/wsc06papers/239.pdf

Pritsker, A. A. B. (1989). Why simulation works. Proceedings of the 1989 Winter Simulation Conference, Paper presented at the 1989 Winter Simulation Conference: Washington, D. C.

Pritsker, A. A. B. (1977). Modeling & analysis using Q-GERT networks. Halsted Press, New York.

Pritsker, A.A.B. (1967). Application of multichannel queueing results to the analysis of conveyor systems. *Industrial Engineering*, 17, 14-21.

Rosenblatt, M. J., Roll, Y. & Zyser, V. (1993). A combined optimization and simulation approach for designing automated storage / retrieval systems, *IIE Transactions*, 25(1), 40-50.

Rother, M. & Harris, R. (2001). *Creating continuous flow: an action guide for managers, engineers, and production associates*. The Lean Enterprise Institute, Brookline, MA.

Rubrich, L. & Watson, M. (1998). Manufacturing cells: One piece flow and the key to employee empowerment and ownership. In *Implementing World Class Manufacturing*. WCM Associates, Fort Wayne Indiana.

Sargent, R. G. (2009). Verification & validation of simulation models. Proceedings of the 2009 Winter Simulation Conference. Paper presented at the 2009 Winter Simulation Conference: Austin, Texas. Retrieved June 5, 2010 from http://www.informs-sim.org/wsc09papers/014.pdf.

Sargent, R. G. (2012). Verification and validation of simulation models. *Journal of Simulation* (2012), 1–13. doi:10.1057/jos.2012.20.

Schmeiser, B. W. (1980). Random deviate generation: a survey. Proceedings of the 1980 Winter Simulation Conference. Paper presented at the 1980 Winter Simulation Conference:Orlando, Florida.

Schonberger, R. J. (2011). Taking the measure lean: efficiency and effectiveness, parts I and II. *Interfaces* 41(2).

Schruben, L. W. (1983). Simulation modeling with event graphs. *Communications of the ACM,* 26(11), 957-965.

Schruben, L. W. (1995). Graphical simulation modeling and analysis: sigma for windows. Duxbury Press, Pacific Grove, CA.

Sekine, K. (1992). *One-piece flow: cell design for transforming the production process*. Productivity Press, Portland, Oregon.

Shannon, R E. (1975). System simulation: the art & science, Prentice-Hall, Englewood Cliffs, N.J.

Shapiro, J. F. (2007). *Modeling the supply chain 2nd edition,* Cengage Learning, Florence, KY.

Standridge, C. R., & Marvel, J. H. (2006). Why lean needs simulation. Proceedings of the 2006 Winter Simulation Conference. Paper presented at the 2006 Winter Simulation Conference: Monterey, California. Retrieved June 5, 2010 from http://www.informs-sim.org/wsc06papers/244.pdf

Standridge, C. R. & Heltne, D. R. (2000). An MSE-based simulation capability for strategic & tactical logistics. Proceedings of the 2000 Winter Simulation Conference. Paper presented at the 2000 Winter Simulation Conference: Orlando, Florida. Retrieved June 5, 2010 from http://www.informs-sim.org/wsc00papers/147.PDF.

Standridge, C. R. (1998). Manufacturing system simulation. In concurrent design of products, manufacturing processes, & systems, H-P Wang, ed. Gordon & Breach Science Publishers, London.

Taj S., Cochran, D. S., Duda, J. W. & Linck, J.(1998). Simulation and production planning for manufacturing cells. Proceedings of the 1998 Winter Simulation Conference, Paper presented at the 1998 Winter Simulation Conference: Washingon, DC. Retrieved September 9, 2010 from http://www.informs-sim.org/wsc98papers/131.PDF.

Vardeman, S. B. & Jobe, J. M. (2001). *Basic engineering data collection and analysis*. Duxbury / Thompson Learning.

Vasudevan, K., Lote, R., Williams, E, & Ulgen, O. (2009). High speed bottle manufacturing lines: case studies and simulation software selection techniques. Proceedings of the 2009 Winter

Simulation Conference, Paper presented at the 2009 Winter Simulation Conference: Austin, TX. Retrieved September 27, 2011 from http://www.informs-sim.org/wsc09papers/030.pdf.

Warber, M. & Standridge, C. R. (2002) Material handling expansion for a package routing hub. Proceedings of the 2002 International Mechanical Engineering Congress and Exposition. Paper presented at the 2002 International Mechanical Engineering Congress and Exposition: New Orleans, LA.

Wilson, J. R. & Pritsker, A. A. B. (1978). A procedure for evaluating startup policies in simulation experiments, *Simulation*, 31, 79-89.