AN APPLICATION OF RESPONSE SURFACE METHODOLOGY

TO ASSEMBLY LINE BALANCING

by

Mark R. Grabau

A thesis submitted to the Faculty and Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of Master of Science (Mathematical and Computer Sciences).

Golden, Colorado

Date 17 Feb 97

Signed:

Mark R. Grabau

Approved:

Dr. Ruth A. Maurer
Thesis Advisor

Golden, Colorado

Date 2/26/97

Dr. Graeme Fairweather
Professor and Head,
Department of
Mathematical and Computer Sciences

ABSTRACT

Coors Brewing Company's 16 ounce can production line was costing over $2 million annually in scrap production. Flow scrap is generated when a machine stops because it is working faster or slower than its adjacent machines. The production line had a flow problem. The machines were not working together in such a way to keep the flow of cans constant from the start to the finish of the line. This work demonstrates an analysis process that minimizes the generation of flow scrap in the context of the assembly line balancing problem. The process is 1) develop a verified and validated simulation model, 2) design an experiment to run using the simulation and to collect output data from the simulation; 3) using the experimental design to determine values of the independent variables and the output data as the dependent variable, develop a metamodel; and 4) optimize the metamodel using response surface methodology. Applying this process to the Coors problem generated an annual savings of $1.87 million in 1996 dollars.

TABLE OF CONTENTS

TABLE OF CONTENTS (CONT.)

TABLE OF CONTENTS (CONT.)

LIST OF FIGURES

LIST OF TABLES

LIST OF TABLES (CONT.)

ACKNOWLEDGEMENTS

First and foremost I would like to thank my beautiful wife, Gina, for her never-ending love and support throughout this entire project. I would like to thank Dr. Ruth Maurer for keeping me on track, steering me through the politics, and getting me this project. I would like to thank Mr. Dennis Ott for his help and guidance throughout the entire project. I would like to thank Professor Bill Astle for his meticulous review that made this thesis significantly better than the original draft. I would like to thank Dr. Bill Navidi for jumping right into the fire once he arrived on campus and helping with the practical statistical issues. I would like to thank my father-in-law, Dr. Mike Belmont, for reviewing this thesis to ensure its grammatical correctness and clarity. Finally, I would like to thank Dave Ciemnoczolowski for his help during the data collection phase of this project.

CHAPTER ONE

INTRODUCTION

1.1 General Importance

1.1.1 General Class of Problem

Many manufacturing systems today are automatic and products are created at an amazing rate. The assembly line allows these products to be produced by a small number of people. In many instances the human factor is nothing more than a maintenance device, fixing problems when they arise. Otherwise, the machines run by themselves. These machines do not have the physical limitations that humans do and, therefore, can operate at significantly higher speeds than a human.

The goal of any assembly line is to continuously generate throughput, by keeping the flow of materials constant from start to finish. Generally, the back of the assembly line pulls the front. More specifically, each machine pulls production from the machine immediately preceding itself. In some manufacturing situations, if a machine runs too fast or too slow, it will cause other machines to stop. When these machines stop, objects currently being manufactured may be scrapped because they fall off the assembly line. Budgets do not permit additional personnel to be hired solely to place these objects back on the assembly line. The more objects that finish the production process, the greater the throughput. So a restatement of the goal becomes, maximize throughput and meet production objectives by minimizing scrap.

The rapid rate at which the whole process is occurring, the interaction between machines, and different transition times between machines make it increasingly more difficult for a human being to make the correct decisions regarding how fast each machine should be working to continue the pulling process, while at the same time keeping scrap low and throughput at an acceptable level. Likewise, a casual observer cannot just observe the system and make these decisions. Many times the system must be intensely studied.

It is possible to design experiments to test a system by setting the machines at certain speeds and observing what happens. However, even this scientific approach to studying a manufacturing system may not be economically feasible. As mentioned above, a human being may not be able to comprehend and anticipate the reaction of the system to certain experimental conditions on the spot. Requiring a human to make these decisions has the potential of creating catastrophic problems just to observe one level of the experiment. In addition, this process can take an inordinate amount of time. These economic and time constraints preclude actual experiments being run on the system. A more economic and timely approach is needed.

1.1.2 General Solution

Simulation modeling of a system is both economical and timely. Once a simulation model is verified and validated, experiments can be run that do not have any direct impact on the system. The only costs are the time to develop the simulation and the computer resources to run the experiments.

Running the experiments correctly using the simulation should be cost-efficient. Even though the simulation model is built and experiments will have no impact on the actual system, there may be additional constraints on the amount of computer time available. Management may also need a timely response to their questions about the system under study. This may preclude testing every possible design point. Testing every possible design point may not be economically feasible either. If the simulation is large and complex, running it at different design points, let alone replications at each design point, may not be possible in the time allotted. Experimental design is a field in and of itself and much effort has been put into developing designs that are the best for certain situations. Even though an efficient design may reduce the number of runs, the amount of output to analyze may still be large.

Simulations have the potential of generating copious amounts of output. Once the experiments are run, a methodology is needed to analyze the results and synthesize them into a form that is presentable to the decision maker. As Arthur Geoffrion (1976) best put it, "The purpose of analysis is insight, not numbers." Metamodeling and response surface methodology are the analysis techniques that lead to the efficient use of simulation output for studying the assembly line optimization problem.

Metamodeling approximates a relationship between a dependent variable and one or more independent variables by using a mathematical function (Banks, Carson, and Nelson 1996, 514). Once a metamodel is developed it can replace another model, i.e. a simulation or linear program. It also allows the data to be combined in a manner that is manageable and provides the necessary insight to answer very specific questions about the system under study. Subsequently, once a system is metamodeled, there is no longer a need to run additional simulations to answer questions regarding the dependent variable given different levels of the independent variables. These types of "what if" questions can be answered with a calculator. Response surface methodology can be used to optimize the metamodel and to obtain the values of the system input parameters, machine speeds in this example, that minimize the amount of scrap that is generated because machines are starting and stopping.

1.2 Overview of Experimental Design

A goal of experimental design is to keep the number of experiments small while still providing a statistically significant analysis. Different designs achieve different results. A full-factorial model, for example, will test every factor at every level with every other factor at each of their levels. If there are 6 factors and 3 levels for each factor, then 3^6, or 729, trials must be made to enumerate all combinations of factor levels. If 5 replications are needed at each design point and each replication takes 9 minutes, then 32,805 minutes of computer time are needed. This is equivalent to running one Pentium 100 MHz computer non-stop for approximately 23 days. Obviously, a smaller, yet reasonably comprehensive design or a faster computer is needed.

Since response surface methodology will be used to optimize the system under study here, it would be ideal for a design to provide good approximation of the expected response while keeping the number of runs to a minimum. A Box-Behnken design has been developed for just this problem, and with 6 factors and 3 levels for each factor, only 54 runs are needed (Launsby and Schmidt 1989, 3-23). This is only 2,430 minutes of computer time on a Pentium 100 MHz processor or 1.7 days, which is a significant reduction from the full-factorial design. This design will estimate individual effects, 2-way interactions, and quadratic effects of the factors (Launsby and Schmidt 1989, 3-23).

1.3 Overview of Metamodeling

The goal is to take the very complex relationships between inputs and the outputs from the simulation and combine them into a simpler mathematical form called a metamodel (Banks, Carson, and Nelson 1996, 514). If Y is the response, or output, from the simulation and there are k inputs, or factors, represented by X's, then metamodeling attempts to define the following mathematical relationship between the Y and X's: $Y = f(X_1, X_2, \ldots, X_k, X_1X_2, X_1X_3, \ldots, X_{k-1}X_k, X_1^2, X_2^2, \ldots, X_k^2)$. This functional form is then defined through multiple regression analysis to be $Y = \beta_0 + \beta_{11}X_1 + \beta_{12}X_2 + \ldots + \beta_{1k}X_k +$

$\beta_{21}X_1X_2 + \beta_{22}X_1X_3 + \ldots + \beta_{2k}X_{k-1}X_k + \beta_{31}X_1^2 + \beta_{32}X_2^2 + \ldots + \beta_{3k}X_k^2 + \varepsilon$. The β's provide insight into which factors are the most important in determining Y if the experimental design is orthogonal, thereby reducing the possibility of any multicollinearity that would cause the estimates to be inefficient.

A statistical analysis of this equation is then performed to determine the significant inputs to the system. An additional benefit is that pictures of the system can be developed that are much more explanatory to a decision maker than an equation. However, if while presenting the results to a decision maker he or she asks, "what if we changed this machine speed to …," an answer can be estimated in seconds with a calculator or by pointing to a location on a chart.

In short, a metamodel is a mathematical equation representing a system. Given specific inputs, a response is estimated without actually modifying the system or rerunning the simulation.

1.4 Overview of Response Surface Methodology

Since the Box-Behnken design estimates linear, interaction, and quadratic effects of inputs, and metamodeling develops a mathematical equation to estimate the output given different inputs, response surface methodology can be used to optimize this mathematical function. Using necessary optimal conditions of a function from calculus, a partial derivative is taken of the output variable with respect to each input variable ($\partial Y/\partial X_i$) and set to zero.

Once the k partial derivatives are taken, since only linear, interaction, and quadratic effects are estimated, what will remain is a system of k linear equations with k unknowns. Such a system can be solved using techniques from linear algebra. The resulting vector is the solution to the system of equations. As long as each element of the vector falls within the design levels from the experiment, this vector will also represent the stationary point of the metamodel. Otherwise, the current solution falls outside the range of the current experiment.

1.5 Summary

Cost-effective methods for studying rapid manufacturing systems are needed in order to make improvements to current operations. Running actual experiments on the system under study is seldom economically feasible or timely. Simulation modeling is a means to accomplish this in both an economic and timely manner. Simulation outputs can then be used to shed insights into the current problem by being analyzed through other statistical methods. By incorporating efficient experimental designs, metamodeling, and response surface methodology, many questions about the system can be answered quickly, and the ideal operating conditions can be determined.

The entire system analysis is a series of steps: 1) develop a verified and validated simulation model using the method outlined in Banks, Carson, and Nelson 1996; 2) design an experiment to run using the simulation and to collect output data from the simulation; 3) using the experimental design to determine values of the independent variables and the output data as the dependent variable, develop a metamodel; and 4) optimize the metamodel using response surface methodology.

CHAPTER TWO

PROBLEM DESCRIPTION

2.1 General Problem Description

Coors Brewing Company is in a cooperative partnership with Valley Metal Container that produces 16 ounce and 12 ounce aluminum cans. The 16 ounce can production line produces 200 million cans for Coors products annually. In addition, they produce over 50 other labels for other beverage companies. The Coors Light ® label accounts for 140 million cans of the 200 million of Coors products. This is the easiest and cheapest can to produce, and it accounts for 56% of the annual production. For this reason, this label is of specific importance to Coors. Since it is the easiest and most economical can to produce, it should represent the lower bound on scrap generation and the upper bound on production throughput.

In 1995, the 16 ounce can production line lost over $2 million gross in production due to scrap. If a can falls down while on the production line, it becomes scrap. Manpower constraints do not allow the positioning of personnel throughout the line to set the fallen cans up. The majority of the cans fall over when the can-pack throughout the line is not tight. While in tightly packed groups, the cans lend each other support to keep them from falling over. This author's goal is to demonstrate how to make the stages of the production line work in concert in order to maintain a tight pack of cans throughout the line and thereby reduce the amount of annual scrap to at least an acceptable level.

2.2 Specifics

In order to manufacture a can, it must go through the following stages:

1) A cup is formed from a sheet of aluminum at one of four cuppers.

2) The cup is stretched to the correct length and the rough top edge is trimmed at one of fifteen body makers. The cup is now a can.

3) The can is washed and dried in the washer to remove the grease from the first two stages.

4) A label is applied at the printer.

5) The paint is dried and the bottom is coated.

6) The inside of the can is coated at one of nine coaters.

7) The can is cured in an oven.

8) The top of the can is prepared for a cover at the necker/flanger.

9) The can is checked for holes and flaws at the tester.

10) The can is placed on one layer of a fourteen layer pallet at the palletizer.

Each of these steps is occurring at a rate of 1400 to 1600 cans per minute. Scrap may be generated at any one of these ten steps. In order to focus on the controllable flow scrap generation, this author breaks scrap up into 3 distinct categories: flow, break, and random.

If the printer or the coaters stop because there are not enough cans behind them or because there are too many cans in front of them, *flow scrap* is generated. The printer clears its wheels. Thermal energy rising in the oven knocks over the unsupported cans that are standing on the outer edges of the group.

Break scrap is caused by a printer problem or the necker/flanger jamming. *Random scrap* is generated in the washer, in the oven, and at the printer. Cans fall over in the washer when an edge of a group of cans is exposed to the sprayers, and cans fall over in the oven when an edge of a group of cans is exposed to thermal energy. Even if the washer and the oven are full, the outside edges of the flow of cans are still exposed. Random scrap at the printer is generated when a can is not properly seated on a wheel.

The reduction of flow scrap is the goal of this work. If the machines work together in such a way that there are no gaps in the line from start to finish, then the absolute amount of flow scrap will be minimized. The speed at which the line operates and the inability of any human being to comprehend all of the possible variables at once preclude a common sense approach based on observation and a subjective decision. This problem must be studied extensively, yet economic constraints do not permit actual experiments to be performed on the line. A methodology is needed that allows the system to be studied in a non-invasive environment at no physical cost to the actual line.

CHAPTER THREE

LITERATURE REVIEW

3.1 Introduction

Assembly line balancing is a analytical problem that is common in industrial engineering, production economics, and operations research literature. In general, this class of problems falls into the NP hard category (Gagnon and Ghosh 1989, 637). One view of line balancing "involves selecting the appropriate combination of work tasks to be performed at each work station so that the work is performed in a feasible sequence and approximately equal amounts of time are required at each of the work stations" (Dilworth 1996, 255).

Assembly line balancing can be broken down into four different types of problems: single model deterministic (SMD), single model stochastic (SMS), multi/mixed model deterministic (MMD), and multi/mixed stochastic (MMS) (Gagnon and Ghosh 1989, 639). The goal of the SMD version is to optimize an efficiency criterion given specific task times; i.e., a conveyor moves products through an entire assembly line at a constant speed. The SMS version has the same goal, but task times may vary; i.e., a worker moves a product to the next station when he is finished. The MMD version allows for different products to be produced given specific task times, i.e. a metal forming shop where different products are being formed but each takes the same amount of time. Finally, the MMS version allows for different products, but, once again, the task times may vary, i.e., a metal forming shop where different products are being formed but where each takes a different amount of time. The problem of interest for this work is the MMD.

Several different solution techniques are available to solve different types of line balancing problems. Some of these include linear programming, integer programming, dynamic programming, genetic algorithms, and heuristics (Gagnon and Ghosh 1989). Heuristics, dynamic programming, and genetic algorithms are briefly discussed, followed by another proven methodology.

3.2 Heuristics

Several algorithms may be used to solve the line balancing problem. The advantage of heuristics is that many times they are presented as simple cookbook methods that can either be implemented by hand or are easily programmed on a computer. It is also this author's experience that heuristics converge to a generally acceptable solution significantly faster than implementing a linear or integer programming solution that normally takes several iterations. In a real-time situation, heuristics may be ideal for their speed and ease of use. Each algorithm and heuristic attempts to optimize a specific efficiency criterion. Two of the more popular heuristics for assembly line balancing were developed by Davis and Taylor (1974) and Gutjahr and Nemhauser (1964). Heuristics, however, do not necessarily provide the optimal solution.

In an effort to ensure implementation, K. Rosco Davis and Bernard W. Taylor introduced the approach to the line balancing problem they termed linear line balancing. The heuristic's goal is to minimize the effect on the production environment while achieving a desired production rate (Davis and Taylor 1974, 57). Davis and Taylor define a balanced line to be "a series of progressively related operations each having approximately equal production times, arranged so that work flows at a desired rate from one operation to the next" (Davis and Taylor 1974, 57). The heuristic can be implemented using a calculator. More importantly, if implemented on a computer, it may be used for real-time process control. However, the problem at hand does not have equal production times, and this heuristic does not attempt to control scrap generation.

Allan L. Gutjahr and George L. Nemhauser developed an algorithm to find the shortest route through any finite directed network and then formulated the line balancing problem to fit this solution technique. The goal of their algorithm is to minimize the sum of the delay times at each station (Gutjahr and Nemhauser 1964, 308). The formulation is difficult to understand and, therefore, not easy to implement. This algorithm also does not attempt to control scrap generation, and minimizing delays times at a station is not of interest to the problem at hand. Therefore, this approach is dismissed as a potential solution technique.

Heuristics can be easy to use, depending on the formulation of the problem. However, what is gained by simplicity and speed is made up for by the inability to guarantee an optimal solution. "Any heuristic can only guarantee results within 150% of the optimal solution" (Gagnon and Ghosh 1989, 641). This wide range is not acceptable. In addition, "exact algorithms become intractable with increasing problem size" (Gagnon and Ghosh 1989, 640). Given the size of the problem and the importance of scrap generation, these specific heuristics are dismissed as viable solution techniques for the current problem.

3.3 Dynamic Programming

Robert Caraway (1989) formulated the assembly line balancing problem as a dynamic program. The goal is to "minimize the required number of work stations on an assembly line for a given cycle time when the processing times are independent, normally distributed random variables" (Carraway 1989, 459). This goal is incompatible with the objective of this work. The capital investment required to alter the number of machines for the current problem is not an acceptable alternative. Because a) the number of work stations is fixed, b) the task times are deterministic, c) the task times are not normally distributed, d) scrap generation is lacking as an important criterion, and e) because of the complexity of the formulation, this technique is also dismissed.

3.4 Genetic Algorithms

Genetic algorithms are the currently fashionable solution technique. G. Levitin and J. Rubinovitz (1995) formulated a genetic algorithm to solve the assembly line balancing problem. These authors define the problem as "how to group the assembly activities, which have to be performed in an assembly task, into workstations, so that the total assembly time required at each workstation is approximately the same" (Levitin and Rubinovitz 1995, 343). They also define two metrics to determine how difficult a given line balancing problem is. A rectangular matrix P is developed where each element is a 1 if a work element immediately precedes another, 0 otherwise. The F-ratio, flexibility ratio, is defined as 2H/[Na*(Na-1)] where H is the number of zero cells above the diagonal and Na is the number of activities at all of the work stations. The higher the F-ratio the better because this means there are fewer precedence constraints for the assembly line. For the Coors problem the F ratio is 0.67 as seen in Figure 1.

		cupper	body makers	printer	coaters	necker/ flanger	tester
	cupper	0	1	0	0	0	0
	body makers		0	1	0	0	0
P =	printer			0	1	0	0
	coaters				0	1	0
	necker/flanger					0	1
	tester						0

F = (2*10)/(6*5)= 0.67

Figure 1: P Matrix for Coors Problem

Another metric is the West-ratio, which equals Na/Nst, where Nst is the number of stations. Good balances are difficult to achieve when Na = Nst because the number of feasible sequences is small. For this problem, the West-ratio is 1.

This genetic algorithm for assembly line balancing works best with a high West-ratio and a high F-ratio. Neither of these is true for the current problem, so this genetic algorithm is also dismissed.

3.5 Method Used

This line balancing problem is extremely complex, and, as was pointed out, scrap is the primary concern and has a large impact on the line's production. None of the above solution techniques accounts for any of these issues. Edward F. Watson and Alan S. Wood have successfully used a computer simulation to improve one of Whirlpool Corporation's assembly lines (Watson and Wood 1995).

Simulation has the advantage of answering many "what if' questions at once and can easily simulate changes to the system without affecting the physical system under observation. Watson and Wood specifically note that "computer simulation proved to be extremely valuable in enhancing the communication process, for helping identify and solve complex production problems, and for helping to establish high levels of confidence in the final system design" (Watson and Wood 1995, 53). They go on to say that "the simulation model helped identify problem areas related to system design and product flow" (Watson and Wood 1995, 53).

This author expands on the definition of a balanced line to fit the current problem as follows: a line is considered balanced when there is an uninterrupted flow of cans, no gaps or hour-glass effects, from the start of the production line to the end. This work implements a simulation model to generate a metamodel of the system. This metamodel is then used to compute the optimal machine speeds using response surface methodology to minimize the amount of flow scrap.

CHAPTER FOUR

STEPS OF THE SIMULATION STUDY

4.1 Introduction

"A simulation is the imitation of the operation of a real-world process of system over time" (Banks, Carson, and Nelson 1996, 3). This representation of the system under study is generally more economical to experiment with than actual experiments on the system itself. It is also significantly more timely. Hours of real time may be simulated in a few minutes. The economical and timely constraints make simulation the ideal methodology for the Coors line balancing problem.

Banks, Carson, and Nelson (1996, 13-18) outline the steps of a simulation study as follows:

1) Problem formulation: A statement of the problem is presented so that all of the interested parties are in agreement on the problem under study.

2) Setting of objectives and overall project plan: This step determines what the goals of the simulation study are and what questions the simulation must answer.

3) Model conceptualization: This is the art of simulation. The simulation is an abstraction of reality that must be specific enough to answer the relevant questions, yet general enough to be modeled.

4) Data collection: The only data that is reliable is the data that is personally collected. This is perhaps the most important part of the simulation study because the data drives the simulation model.

5) Model translation: Given the conceptual model and the collected data, a simulation platform must be chosen that can accomplish the simulation effort.

6) Verification: The simulation model is verified when it is confirmed that the simulation is performing as the modeler expects.

7) Validation: The simulation model is validated when it is confirmed that the verified model is a reasonable abstraction of reality.

8) Experimental design: Once it is determined that a useful model has been developed, experiments are performed using the model that do not have a direct impact on the system being studied.

9) Analysis: After the necessary runs are completed, the performance of the system is analyzed.

10) Documentation and reporting: Once the analysis is complete, the results are briefed to the decision maker for action.

11) Implementation: This depends on how well the other steps are performed and how well the report is sold. Without system acceptance, the entire project is a waste of time.

Each step as it pertains to the Coors problem will now be discussed in detail.

4.1.1 Problem Formulation

The current perception is that there is a flow scrap problem on the 16 ounce production line. The machines are not working together in a way that minimizes flow scrap and keeps a constant flow of cans from start to finish. The line is too complex to use heuristics or common sense, and actual experiments on the line are not economically feasible. What is needed is a simulation model of the production line. Once this is built, it will be used to run experiments on the line and determine the optimal settings of the machines to minimize flow scrap.

4.1.2 Setting of Objectives and Overall Project Plan

The primary goal of the simulation project is to develop a realistic simulation of the 16 ounce production line. The model needs to be general enough to be easily extended to other labels, yet specific enough to answer the following questions: 1) what are the optimal settings of the machines to reduce flow scrap and 2) where is the scrap being generated?

4.1.3 Model Conceptualization

Figure 2 contains a schematic of the 16 ounce production line.

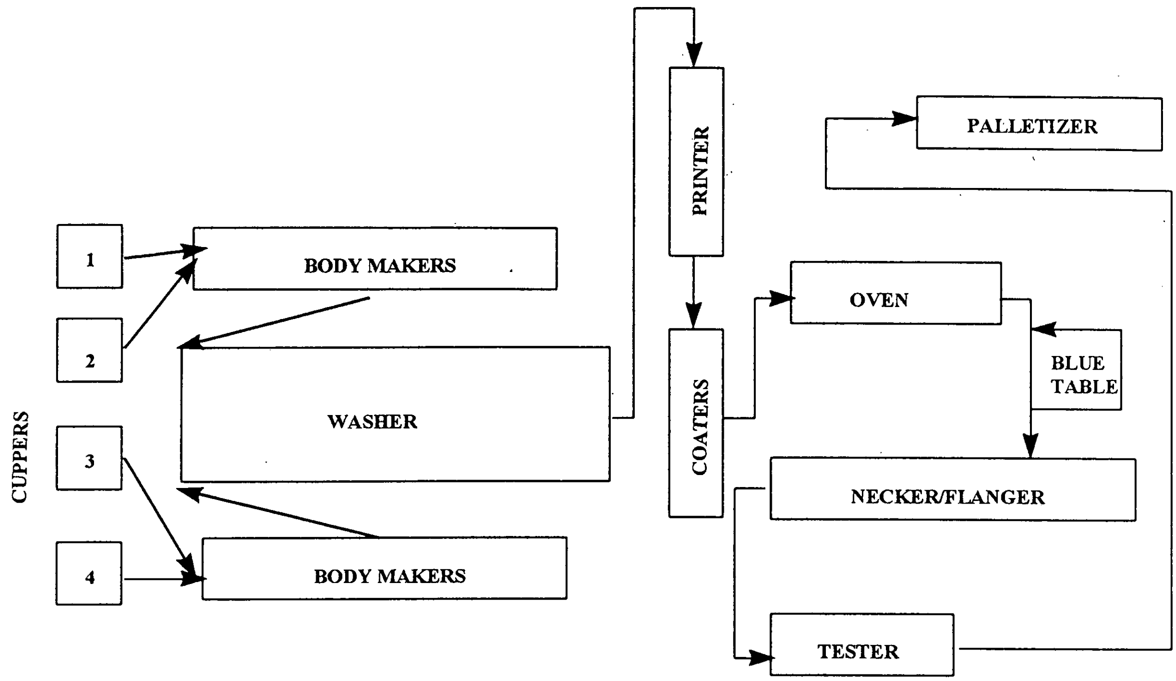

Figure 2: Coors 16 Ounce Production Line

At the beginning of the line, the cuppers create cups from a sheet of aluminum. There are currently 4 cuppers, and each produces 4 cups with each motion. The cuppers must also check the body maker chutes to ensure there are not too many cups being sent to the body maker area. The cups travel to the body makers via an elevator conveyor and continue to cycle above the body makers until there is room in a body maker chute.

The body maker area is composed of 15 body makers, each capable of servicing 125 cans per minute. The output of the body maker area solely depends on how many body makers are operating. If the body maker chutes fill up, the cuppers will shut off. Any cups that were created and sent to the body makers before the cuppers shut off cycle on a conveyor above the body makers until there is room in a chute. Once the cup is formed into a can and trimmed, it must travel to the washer via an elevator conveyor. The can then travels through the washer to the printer. If the washer gets too full, the body makers shut off.

While traveling through the washer, many cans are knocked over by spray from the washing device. If a can is knocked over, it is removed from the line by falling into a bucket under the washer or by falling into a box at the end of the washer. This is random scrap. At the end of the washer, the cans travel up and around a C curve to the printer air table.

The printer runs at a speed set by the operator. It is here that the specific label is applied to the can. The printer checks its air table to be sure that it is at a high enough level. If it is not, then the printer shuts off. If the printer air table gets too high, then the washer and C curve stops. If the coater air table gets too high, then the printer stops. The printer also stops when it breaks. Any time the printer stops, cans are scrapped. Finally, some cans are not properly seated on the printer wheel. When this happens the can becomes random scrap. Once a can is processed through the printer, it travels via mechanized conveyor and air table to the coater area.

The coater area is composed of 9 coaters, each operating at 195 cans per minute. The coaters seldom break and, therefore, generate an insignificant amount of break scrap. The coater area is either all on or all off. Once the can is processed through the coater area, it travels through the oven. If the coater air table gets too low, then the coater area will shut off. If the necker/flanger air table gets too full, then the coater area will shut off.

If the coaters are shut off for more than 20 seconds, then a gap is created in the oven. If a gap is created in the oven, then thermal energy knocks cans over. Once a can falls down, it becomes scrap. Cans on the outside edge of the flow of cans through the oven are also knocked down due to thermal energy, regardless of whether or not the coaters have been down for more than 20 seconds. This is random scrap. If there are cans in the oven and the necker/flanger air table is full, then once the cans are at the end of the oven, they cycle on a mechanized conveyor (blue table) until there is room for them in the necker/flanger air table.

The necker/flanger runs at a speed set by the operator. The necker/flanger will continue to run unless its air table is too low or until the tester air table is too high. The necker/flanger can start and stop without creating flow scrap. However, if a can with flaws from the trimming at the body makers makes it to the necker/flanger, the necker/flanger will jam and crush several cans. These cans are break scrap. Once the jam is cleared by the operator, the necker/flanger resumes processing cans.

The tester also runs at a speed set by the operator. It will also continue to run unless its air table gets too low or the palletizer air table gets too high. The number of cans the tester rejects is negligible when compared to other types of scrap.

After leaving the tester, the can travels down a long air table to the palletizer. It is in this air table that the cans are grouped into layers. If enough layers are formed in the air table and enough layers are immediately behind the palletizer, then the palletizer will move a layer onto the pallet. For Coors' products, each pallet has 14 layers. Once a pallet is complete, it is moved by a forklift to the warehouse. The palletizer stops when a can is jammed in one of its chutes, when the air table is not full enough, or when it breaks.

4.1.4 Data Collection

The simulation is only an empty shell without data to drive it. Based on the conceptualization, the following data are needed: travel times on the air tables and mechanized conveyors, scrap generation rates, time between machine breaks, and machine down times. BestFit (Pallisade 1995) is a software package used to determine which probability density function statistically best fits the collected data. SLAM II is the simulation language that is used for this work. A second requirement for the density function chosen is that is must be supported by SLAM II (Pritsker 1996).

BestFit performs Anderson-Darling, Chi-Squared, and Kolmogorov-Smirnov tests at $\alpha = 0.05$, after deriving estimates of the distribution specific parameters from the input data using maximum likelihood estimation, to determine if the data fits the hypothesized density function. The hypothesized density function for the amount of oven flow scrap is a gamma distribution with $\alpha = 3.04$ and $\beta = 9.60$. The alternative hypothesis is that it is not a gamma distribution with these parameters. Figures 3 to 5 show an example of the

BestFit analysis for the amount of oven flow scrap. Test statistics with a 0 subscript are the calculated values. Test statistics with a c subscript are the critical values.

H_0: gamma(3.04, 9.60)
H_1: not gamma(3.04, 9.60)
χ^2_c: 16.92
Rejection Region: $\chi^2_0 \geq \chi^2_c$ $\chi^2_0 = 3.93 < 16.92$
Conclusion: Fail to reject H_0, that the data is gamma(3.04, 9.60).

Figure 3: Chi-Squared Test

H_0: gamma(3.04, 9.60)
H_1: not gamma(3.04, 9.60)
KS_c: 1.36
Rejection Region: $KS_0 \geq KS_c$ $KS_0 = 0.09 < 1.36$
Conclusion: Fail to reject H_0, that the data is gamma(3.04, 9.60).

Figure 4: Kolmogorov-Smirnov Test

H_0: gamma(3.04, 9.60)
H_1: not gamma(3.04, 9.60)
AD_c: 2.49
Rejection Region: $AD_0 \geq AD_c$ $AD_0 = 0.32 < 2.49$
Conclusion: Fail to reject H_0, that the data is gamma(3.04, 9.60).

Figure 5: Anderson-Darling Test

Fitting the other random number distributions follows a similar procedure. Table 1 shows the other distributions.

Activity	Distribution
1) body maker to washer	1) beta(1.43, 3.51)*35.06 + 18.97
2) washer	2) lognormal(428.9, 24.9) + 10.9
3) printer to coater air table	3) lognormal(48.5, 7.0) + 10.9
4) coater through the oven	4) lognormal(154.6, 8.0) + 13.45
5) interarrival printer break	5) gamma(571.1, 1.2)
6) printer downtime	6) lognormal(121.5, 111.1)
7) interarrival necker/flanger break	7) exponential(697.4)
8) necker/flanger downtime	8) lognormal(33.0, 21.4)
9) necker break scrap	9) lognormal(42.2, 83.9)
10) interarrival palletizer break	10) exponential(162.6)
11) palletizer downtime	11) lognormal(15.5, 4.5)
12) interarrival box scrap	12) beta(0.65, 1.5)*106.5 + 5.96
13) interarrival bucket scrap	13) exponential(10.2)
14) interarrival oven random scrap	14) lognormal(15.5, 17.1)
15) interarrival printer random scrap	15) exponential(15.8)

Table 1: Probability Distributions

4.1.5 Model Translation

An abstract representation of each stage of the production line is needed in order to make this complex problem manageable. Each stage of the can line is a separate area where distinct operations and queuing principles are applied. SLAM II is the simulation language used; it is assumed that the reader has a basic familiarity with SLAM II. The conceptual model is now translated into the SLAM II simulation language. A schematic of the 16 ounce can production line is in Figure 2.

Cups are represented by entities. These same entities represent the cans when the cups are processed into cans. In order to keep the simulation time at a reasonable length, 1 entity represents 16 cups or cans. This simplification does not cause a loss of fidelity since several cups or cans normally travel together. All times are in seconds. Since entities represent more than one can, process times are scaled accordingly. For example, if the cuppers are operating at 2000 cans per minute, then the time to create one entity is

$$\frac{60 \text{ seconds}}{1 \text{ minute}} \cdot \frac{16 \text{ cans}}{\text{entity}} \cdot \frac{1 \text{ minute}}{2000 \text{ cans}} = 0.48 \text{ seconds / entity}.$$

Each machine on the line is modeled as a resource. The cuppers, body makers, and coaters are aggregated into one resource. The speed of the resource is the speed of one machine times the number of machines running. The machine's air table or chute act as queues. When an entity arrives at an air table, it waits in a first-in-first-out queue until the resource is available. Resources can also be preempted, simulating a broken machine.

A machine breaking is modeled as a separate entity arriving at a resource which moves to the front of the queue. The break entity then uses the resource until it is fixed. Resources can also balk and block.

If the body maker resource queue is full, then the entity balks to the track above the body makers where it cycles until the body maker queue has room. If the necker/flanger queue is full, then the entity balks to the blue table where it cycles until there is room in the necker/flanger queue.

If the printer queue, coater queue, tester queue, or palletizer queue gets too full, then it will block the preceding activity. This causes entities to stop moving through the system and to wait until the blocking queue has enough room to proceed.

Entities enter the system at a create node, where they are sent to the cupper resource to await processing. When the cupper resource is available, an attribute for that entity is set that determines at what time the entity was created. This attribute is then used at the palletizer to determine how long is takes an entity to travel through the system on average. If the body maker queue and the track above the body maker are not too full, the entity proceeds to the body maker resource. Travel time to the body maker queue is deterministic because of the extreme difficulty in observing enough travel times to fit a random number distribution.

When the line starts up after a weekend, some cups are left in the system at the body makers. This is simulated by filling the body maker queue at time 0.00 in the simulation. Since the body makers are aggregated into one body maker resource, the entities travel to the centroid of the body maker area. Once at the body maker resource, the entity waits in a queue until the resource is available. Once the body maker resource is available, the entity is processed, and if it is determined that it will not be random scrap in the washer, then it travels to the washer and through it. Interarrival times for random scrap are determined using a random number distribution, and travel times to and through the washer are based on random number distributions from collected data. Disjointed networks represent the control logic on the line. A diagnostic for the body maker checks to be sure that the washer is not too full. If it is, an entity is sent to the body maker resource and placed at the front of the queue. This entity then holds the resource until the washer activity is at an acceptable level. After the entity travels through the washer, it proceeds to the printer.

Like the body makers, the printer also has cans waiting in its queue when the line starts after a weekend. This is simulated by filling the printer queue to this level at time 0.00 in the simulation. Once an entity leaves the washer it must travel up and around a C curve and down an air table to the printer queue. If the printer queue is too full, then it blocks the C curve activity. Once the C curve is too full, it blocks the washer, eventually shutting down the body maker and cupper. Travel time on the C curve is deterministic. The air tables are engineered to move cans at 200 feet per minute. This constant is used to determine travel time for all air tables. The travel time from the C curve to the printer queue is the time it would take an entity traveling through an empty air table minus the time it would take an entity to travel from the current level of the queue to the printer resource. A disjointed network representing the control logic for the printer area checks to be sure that the printer queue is not too low and that the coater queue is not too high. If either of these conditions exist, then an entity is sent to the front of the printer queue to hold the resource until the queue lengths are at acceptable levels. Any time the printer stops for these reasons, one entity is removed as flow scrap. Once at the printer resource, the entity is processed. If it is determined that this entity will not become part of printer break, random, or flow scrap, the entity then travels to the coater area. Interarrival times of random scrap are determined by a random number distribution. The interarrival times for a break, along with the time to fix the printer, are also determined using a random number distribution. Travel time to the coater air table is based on a random number distribution from collected data.

Once an entity leaves the printer it must travel up to the coater queue. If the coater queue is too full, then it blocks the activity from the printer to the coater. Once this blocked activity is too full, it blocks the printer, eventually shutting down the C curve, washer, body maker, and cupper, if the delay is long enough. The travel time in the air table to the coater queue is the time it would take an entity traveling through an empty air table minus the time it would take an entity to travel from the current level of the queue to the coater resource. A disjointed network, representing the control logic for the coater, checks to be sure that the coater queue is not too low and that the necker/flanger queue is not too high. If either of these conditions exists, then an entity is sent to the front of the coater queue to hold the resource until the queue lengths are at acceptable levels. Once at the coater resource, the entity is processed. If it is determined that this entity will not become part of oven random or flow scrap, the entity then travels to the necker/flanger area. Random scrap at the printer has an interarrival time that is determined by a random number distribution. The amount of flow scrap is also determined by a random number distribution if the coater stops for more than 20 seconds. Travel time through the oven to the necker/flanger queue is based on a random number distribution from collected data.

Once an entity leaves the coater it must travel to the necker/flanger queue. If the necker/flanger queue is too full, then the entity balks to the blue table where it cycles until there is room in the queue. The coater diagnostic checks the level of the necker/flanger queue. If this queue is too high and the coater queue fills up because the coater stops, it blocks the printer, eventually shutting down the C curve, washer, body maker, and cupper, if the delay is long enough. The travel time in the air table to the necker/flanger queue is the time it would take an entity traveling through an empty air table minus the time it would take an entity to travel from the current level of the queue to the necker/flanger resource. A disjointed network representing the control logic for the necker/flanger checks to be sure that the necker/flanger queue is not too low and that the tester queue is not too high. If either of these conditions exists, then an entity is sent to the front of the necker/flanger queue to hold the resource until the queue lengths are at acceptable levels. Once at the necker/flanger resource, the entity is processed. The interarrival time of a necker/flanger break is determined by a random number distribution. The amount of break scrap is also determined by a random number distribution. If it is determined that this entity will not become part of necker/flanger break scrap, the entity travels to the tester area based on a deterministic travel time.

Once an entity leaves the necker/flanger it must travel up to the tester queue. If the tester queue is too full, then it blocks the activity from the necker/flanger to the tester. Once this blocked activity is too full, it blocks the necker/flanger, coater, printer, C curve, washer, body maker, and cupper, if the delay is long enough. The travel time in the air table to the tester queue is the time it would take an entity traveling through an empty air table minus the time it would take an entity to travel from the current level of the queue to the tester resource. A disjointed network representing the control logic for the tester checks to be sure that the tester queue is not too low and that the palletizer queue is not too high. If either of these conditions exist, then an entity is sent to the front of the tester queue to hold the resource until the queue lengths are at acceptable levels. Once at the tester resource the entity is processed. The entity then travels to the palletizer through an air table. The travel time from the tester to the air table is deterministic.

The palletizer is divided into two parts. The first part groups entities into pallet layers in the air table. The second part places the layers onto a pallet. Separate control diagnostics exist for each part of the palletizer. The first diagnostic ensures the level of the first queue is at an acceptable level and that the second queue is not too full before releasing a layer. If the first palletizer queue gets too full, the rest of the line will systematically shut down because of the blocking resources. The second diagnostic ensures the queue for the second part is full enough to warrant moving another layer onto the pallet.

The travel time in the air table to the first palletizer queue is the time it would take an entity traveling through an empty air table minus the time it would take an entity to travel from the current level of the queue to the first palletizer resource. Once the level in the first queue is at an acceptable level, a layer is released down a mechanized conveyor. The travel time on the conveyor to the second palletizer queue is the time it would take a layer traveling on an empty conveyor minus the time it would take a layer to travel from the current level of the queue to the second palletizer resource. Once the second queue is at an acceptable level, a layer is placed on a pallet. The time to move a layer onto a pallet is deterministic for the each of the fourteen layers. A tag is placed on the tenth layer, so this is simulated by a longer time. Once the pallet has all fourteen layers, the second palletizer resource is held until the pallet clears the palletizer and a new pallet is moved into position.

Several statistics are calculated throughout the simulation. The amount of scrap at each location is accumulated. At the palletizer, the time for an entity to move through the system is calculated. The interarrival time of pallets and the number of pallets are also accumulated. Since a label change is not an important part of the study, the simulation is allowed to warm up for an hour of simulation time, and then these statistics are cleared. Statistics are then kept for another eight hours of simulation time.

Possible proprietary considerations preclude displaying the actual SLAM II networks.

4.1.6 Verification

SLAM II makes verification easy. The TRACE option prints the entire event list allowing the entity to be tracked through the system. Because of the size of the simulation and the number of entities being processed each second, the networks were verified one at a time. Each condition that causes a state change was tested. The queue levels that the diagnostics check were also reduced to smaller levels so the TRACE report could be kept to a reasonable size.

This process led to changes in the body maker diagnostic. The body maker queue is managed by the cupper checking to make sure it is not too full and that the track above the body maker is not too full. So having the body maker double check this was not necessary and was shutting off the body makers unnecessarily. The body maker only needs to check the level of the washer activity. This change was made and the networks reverified.

The palletizer network was also changed in the verification. Originally, the layers being formed were too large, so an incorrect number of cans was eventually placed on a pallet. This was changed to ensure that the correct number of cans was placed on a pallet.

Copies of the TRACE reports are not provided due to their size and complexity. (Each is approximately ten to twenty pages long.) These reports may be obtained from the author upon request.

4.1.7 Validation

Data were collected based on the number of pallets over a four hour period. Validation is based on the number of pallets and uses the methodology in Banks, Carson, and Nelson (1996). Table 2 shows the data for the number of pallets per hour validation. Z is the number of pallets observed; W is the number of pallets using the simulation; and d is the difference between Z and W. $\bar{d}$ is the average difference. S_d^2 is the sample variance of the differences.

Z	W	d
12	12.75	-0.75
14	15	-1
14	14	0
13	12.75	0.25
		$\bar{d}$=-0.375
		S_d^2=0.3541

Table 2: Pallets Per Hour Validation

The null hypothesis is that the mean of the difference is zero. The alternative hypothesis is that this mean does not equal zero. This is a two-tailed test. The test statistic is from a student-t distribution with level of significance α and n - 1 degrees of freedom. The null hypothesis is rejected if the absolute value of the test statistic is less than or equal to the critical value. The hypothesis test is located in Figure 6.

H_0: $\mu_d = 0$
H_1: $\mu_d \neq 0$
t_c: 3.182
Rejection Region: $|t_0| \geq t_{0.025,3}$ $t_0 = \bar{d}/(S_d/\sqrt{K}) = 1.26 < 3.182$
Conclusion: Fail to reject H_0, that there is not a difference between the simulation and the actual system.

Figure 6: Pallets Per Hour Validation

4.1.8 Experimental Design

The first part of the design of the experiment is to determine how many replications are necessary for statistical significance. This test is done using the number of pallets per hour. The goal is to be able to determine the number of pallets per hour with an error, ε, of plus or minus one pallet. S_0 is an estimate of the population standard deviation. The number of runs required is then calculated using the following formula: $R \geq \left(\frac{z_{\alpha/2} S_0}{\varepsilon}\right)^2$, where $z_{\alpha/2}$ is the tabulated z statistic (Banks, Carson, and Nelson 1996, 449). For this test, α is set at 0.05; therefore, $z = 1.96$. The simulation was run for 4 hours and the standard deviation of the average pallets per hour calculated. $R \geq \left(\frac{1.96 \cdot 1.127}{1}\right)^2 = 4.882$.

Therefore, after rounding up, five replications will be needed for each run of the simulation.

At the beginning of each week, the production line begins empty. The first shift must fill the line with cans, and then the remainder of the week there are cans throughout the production line. This start-up condition must also be simulated. Statistics during this part of the simulation may negatively bias the final results since the line takes time to "warm up" and begin operating consistently. This state of the system after this warm up period is also called steady state, the point in the simulation when the current state is not dependent on time (Banks, Carson, and Nelson 1996, 257).

Once the simulation reaches steady state, the statistics need to be reset to zero so unbiased statistics may be calculated. A machine breaking is a significant random shock to the system and may never allow the simulation to approach steady state; therefore, the simulation was run for a nine hour period where the machines were not allowed to break. Filling the line during the first hour is similar to a start-up after a weekend or after a label change. As seen in Figure 7, the system levels off at fifteen pallets per hour after the first hour. The simulation was then designed to clear the statistics after a one hour warm-up period. Then statistics are calculated for one eight hour shift.

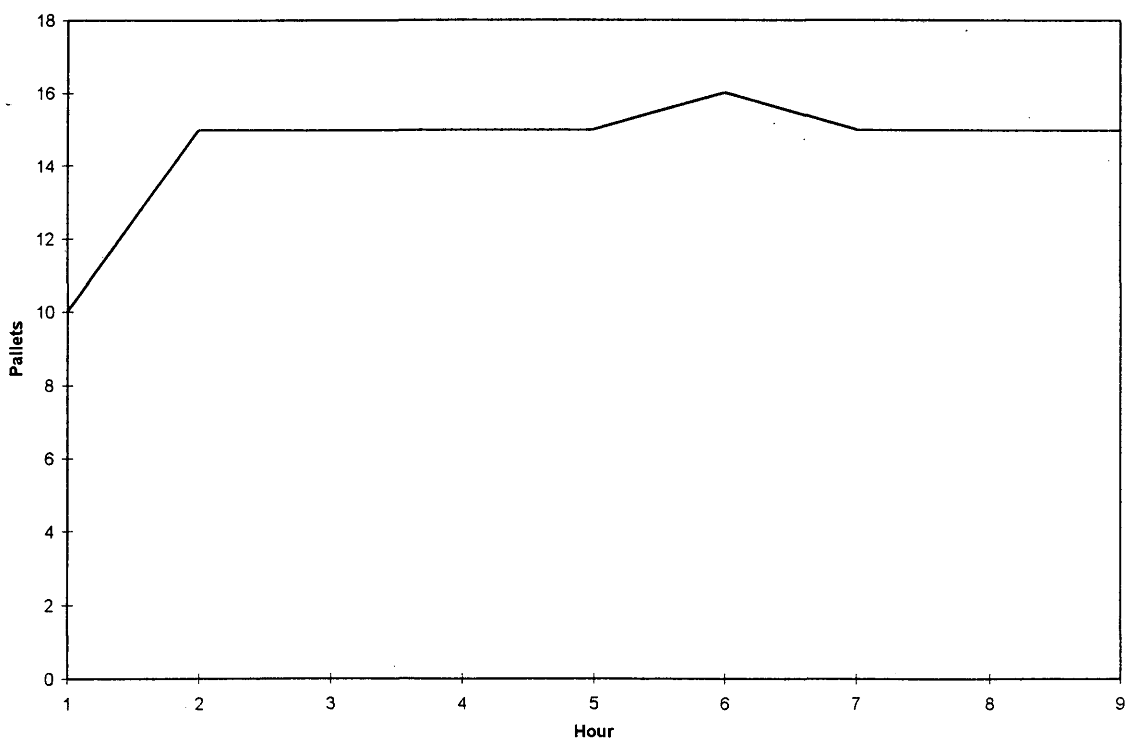

Figure 7: Steady State Determination

Theoretically, the machine speed should be able to increase to a certain point where the system is overloaded and flow scrap gets worse instead of better. This effect can be modeled as a quadratic effect; therefore, the experiment will need to be run at three levels for each machine in order to estimate this quadratic effect. Only running the experiment at two levels would only fit a line to two points. In addition, adjacent machines interact with each other by checking the level of the queue in front of them; therefore, the design must also account for interaction effects. Table 3 shows the levels of the experiment with the corresponding machine speeds. There are also only 48 hours of available computer time on a Pentium 100 MHz processor; therefore, the design will need to keep the number of runs low in order to meet this constraint.

Levels	**cuppers**	**body makers**	**printer**	**coaters**	**necker**	**tester**
-1	1600	1375	1400	1365	1600	1600
0	1800	1500	1500	1560	1700	1700
+1	2000	1625	1600	1755	1800	1800

Table 3: Experimental Design Levels

With six factors and three levels, a full factorial design requires $3^6 = 729$. Each run has five replications and takes forty-five minutes on a Pentium 100 MHz processor. This is equivalent to 546 hours and 45 minutes of computer time, or 23 days straight. Another type of a design that could be used is the central composite design (Launsby and Schmidt 1989, 3-23). This would require 90 runs. This is equivalent to 67 hours and 30 minutes or 3 days straight on the same computer. Time does not permit this number of runs, so an alternative design must be considered.

The ultimate goal of the simulation is to estimate a metamodel and optimize the response surface. A design for this type of analysis is a Box-Behnken design (Launsby and Schmidt 1989, 3-23). This type of design for six factors with three levels and five center point runs requires 54 runs. This is equivalent to 40 hours and 30 minutes, or less than 2 days on the same computer. This is definitely feasible and also models the needed linear, quadratic, and interaction effects. The Box-Behnken design also allows valid statistical analysis with as few runs as possible. Drawbacks of the Box-Behnken design are a) corner points are not tested and b) there are enough runs to estimate all interactions and quadratic effects whether the analyst wants to or not (Launsby and Schmidt 1989, 3-18). A comparison of the three designs is in Table 4.

Design Type	Number of Runs	Computer Hours	Feasible ?
Full Factorial	729	546.75	No
Central Composite	90	67.5	No
Box-Behnken	54	40.5	Yes

Table 4: Experimental Design Comparison

The goal of the study is to minimize flow scrap. The response for the experiment is the average number of cans per hour of flow scrap. The entire design matrix along with the responses for the first 27 runs are in Table 5, and the last 27 runs are in Table 6. The first column is the number corresponding to the design point. The second through the seventh columns are the factors (the machines) and the eighth column is the response or output from the experiment, flow scrap.

Obs	Cupper	BodyMaker	Printer	Coater	Necker	Tester	Scrap
1	1	1	0	1	0	0	146.0
2	0	0	1	1	0	-1	243.2
3	-1	1	0	1	0	0	144.8
4	1	0	0	-1	-1	0	222.8
5	0	-1	0	0	-1	1	183.2
6	1	0	0	1	1	0	220.4
7	0	-1	-1	0	1	0	137.2
8	1	-1	0	-1	0	0	224.8
9	0	-1	-1	0	-1	0	140.8
10	0	0	0	0	0	0	140.8
11	0	1	1	0	1	0	154.8
12	0	-1	0	0	1	1	168.0
13	0	0	-1	1	0	-1	138.4
14	0	-1	0	0	-1	-1	184.0
15	1	0	1	0	0	1	200.8
16	-1	1	0	-1	0	0	219.2
17	0	0	-1	1	0	1	138.4
18	0	1	0	0	1	-1	141.2
19	0	1	0	0	1	1	140.4
20	0	0	-1	-1	0	1	133.2
21	0	1	-1	0	-1	0	138.0
22	0	1	0	0	-1	-1	151.2
23	1	0	1	0	0	-1	202.4
24	-1	0	0	-1	1	0	216.8
25	-1	-1	0	-1	0	0	220.4
26	-1	-1	0	1	0	0	194.8
27	0	0	0	0	0	0	140.8

Table 5: First 27 Runs of Box-Behnken Design

Obs	Cupper	BodyMaker	Printer	Coater	Necker	Tester	Scrap
28	1	0	0	1	-1	0	150.0
29	0	-1	0	0	1	-1	169.6
30	0	1	1	0	-1	0	157.6
31	1	-1	0	1	0	0	172.8
32	0	0	1	-1	0	-1	252.8
33	-1	0	1	0	0	1	154.8
34	1	0	-1	0	0	-1	138.8
35	0	0	0	0	0	0	140.8
36	-1	0	1	0	0	-1	163.6
37	0	0	0	0	0	0	140.8
38	0	0	1	1	0	1	246.4
39	1	0	0	1	1	0	140.0
40	0	-1	1	0	1	0	936.0
41	-1	0	0	-1	-1	0	220.4
42	-1	0	-1	0	0	1	135.6
43	0	-1	1	0	-1	0	869.2
44	0	1	0	0	-1	1	146.8
45	0	1	-1	0	1	0	136.0
46	0	0	0	0	0	0	140.8
47	-1	0	0	1	-1	0	147.6
48	-1	0	0	1	1	0	142.0
49	1	0	-1	0	0	1	140.0
50	-1	0	-1	0	0	-1	136.4
51	1	1	0	-1	0	0	218.4
52	0	0	1	-1	0	1	252.8
53	0	0	0	0	0	0	140.8
54	0	0	-1	-1	0	-1	133.2

Table 6: Last 27 Runs of Box-Behnken Design

4.1.9 Analysis

In order to determine if the metamodel and response surface optimization are beneficial, a baseline must be established. Table 7 shows the simulation results for observed settings of the machines during the data collection phase along with the average flow scrap and pallets per hour.

cuppers	1890 cans/minute
body makers	12 running at 125 cans/minute
printer	1500 cans/minute
coaters	7 running at 195 cans/minute
necker/flanger	1630 cans/minute
tester	1800 cans/minute
flow scrap/hour	219 cans
pallets/hour	13.1

Table 7: Baseline Simulation Settings and Output

The next step is to use the experimental data from Tables 5 and 6 to determine the metamodel and calculate the response surface using the following model:

$$\begin{aligned}\text{Flow Scrap} = {} & \beta_0 + \beta_1\text{Cupper} + \beta_2\text{BodyMaker} + \beta_3\text{Printer} + \beta_4\text{Coater} + \beta_5\text{Necker} \\ & + \beta_6\text{Tester} + \beta_7\text{Cupper*Cupper} + \beta_8\text{BodyMaker*Cupper} + \beta_9\text{BodyMaker*BodyMaker} \\ & + \beta_{10}\text{Printer*Cupper} + \beta_{11}\text{Printer*BodyMaker} + \beta_{12}\text{Printer*Printer} + \beta_{13}\text{Coater*Cupper} \\ & + \beta_{14}\text{Coater*BodyMaker} + \beta_{15}\text{Coater*Printer} + \beta_{16}\text{Coater*Coater} + \beta_{17}\text{Necker*Cupper} \\ & + \beta_{18}\text{Necker*BodyMaker} + \beta_{19}\text{Necker*Printer} + \beta_{20}\text{Necker*Coater} + \beta_{21}\text{Necker*Necker} \\ & + \beta_{22}\text{Tester*Cupper} + \beta_{23}\text{Tester*BodyMaker} + \beta_{24}\text{Tester*Printer} + \beta_{25}\text{Tester*Coater} \\ & + \beta_{26}\text{Tester*Necker} + \beta_{27}\text{Tester*Tester} + \varepsilon. \end{aligned} \tag{4.1}$$

This model includes all linear effects, quadratic effects, and interaction effects. Statistical Analysis Software (SAS) is used to calculate the parameters in the model, specifically, the procedure to estimate a response surface using ordinary least squares, SAS PROC RSREG. The results are displayed in Tables 8 and 9.

Parameter	Estimate	Error	T	Prob > |T|
Intercept	140.800	44.470	3.166	0.0039
Cupper	4.447	22.838	0.195	0.8471
BodyMaker	-71.100	22.235	-3.198	0.0036
Printer	91.183	22.235	4.101	0.0004
Coater	-19.894	23.107	-0.861	0.3971
Necker	0.697	22.838	0.031	0.9759
Tester	-0.600	22.235	-0.027	0.9787
Cupper*Cupper	-14.869	34.142	-0.435	0.6668
BodyMaker*Cupper	2.250	38.512	0.058	0.9539
BodyMaker*BodyMaker	48.337	34.669	1.394	0.1750
Printer*Cupper	9.750	38.512	0.253	0.8021
Printer*BodyMaker	-186.100	38.512	-4.832	0.0001
Printer*Printer	103.134	34.009	3.033	0.0054
Coater*Cupper	1.459	28.820	0.051	0.9600
Coater*BodyMaker	-8.650	38.512	-0.225	0.8240
Coater*Printer	-3.300	38.512	-0.086	0.9324
Coater*Coater	18.382	34.142	0.538	0.5949
Necker*Cupper	2.842	41.568	0.068	0.9460
Necker*BodyMaker	-3.425	27.232	-0.126	0.9009
Necker*Printer	8.700	38.512	0.226	0.8230
Necker*Coater	5.468	42.885	0.128	0.8995
Necker*Necker	41.429	35.059	1.182	0.2480
Tester*Cupper	1.150	38.512	0.030	0.9764
Tester*BodyMaker	-0.350	38.512	-0.009	0.9928
Tester*Printer	-0.475	27.232	-0.017	0.9862
Tester*Coater	0.400	38.512	0.010	0.9918
Tester*Necker	0.350	38.512	0.009	0.9928
Tester*Tester	-70.016	34.009	-2.059	0.0497

Table 8: Metamodel 4.1

Regression	Degrees of Freedom	Type I Sum of Squares	R-Square	F-Ratio	Prob > F
Linear	6	331289	0.2981	4.653	0.0024
Quadratic	6	191631	0.1725	2.692	0.0362
Crossproduct	15	279740	0.2518	1.572	0.1513
Total Regress	27	802659	0.7224	2.505	0.0109

Residual	Degrees of Freedom	Sum of Squares	Mean Square	F-Ratio	Prob > F
Lack of Fit	20	305268	15263	28.335	0.0002
Pure Error	6	3232	538.68		
Total Error	26	308500	11865		

Table 9: ANOVA Table for Metamodel 4.1

The F statistic for this model is 28.335 with a p-value of 0.0002, indicating the model has explanatory significance at $\alpha = 0.05$. The R^2 is 0.72; therefore, metamodel 4.1 explains 72% of the variation in flow scrap. Looking at the Prob > |T| column, a value less than $\alpha = 0.05$ indicates a significant factor in determining the amount of flow scrap. Table 8 shows that the Printer and BodyMaker speeds are significant factors in determining the amount of flow scrap and should be included in a reduced model of the system. The quadratic Tester term is also significant; however, the Tester is not significant linearly or while interacting with other machines. In summary, this model does a reasonable job of predicting flow scrap and may be used as a proxy for the simulation to answer the "what if" questions about the impact of different machine speeds on flow scrap.

Using all of the coefficients in metamodel 4.1, the next step is to find the optimal combination of machine speeds that minimizes this metamodel by using response surface methodology. This is done by taking partial derivatives of metamodel 4.1 with respect to every variable and setting the result equal to zero. This system of equations may then be solved to determine the optimal point of the metamodel. Table 10 shows the stationary point calculated from metamodel 4.1.

Cupper	0.0617
BodyMaker	0.0804
Printer	-0.3639
Coater	0.5255
Necker	-0.0037
Tester	-0.0013

Table 10: Stationary Point for Metamodel 4.1

Eigenvalues and eigenvectors are used in a multidimensional model to determine if the stationary point is a maximum, minimum, or a saddle point. Table 11 shows how to interpret the eigenvalues to determine what stationary point has been reached.

Eigenvalue Sign	**Stationary Point**
1) All positive →	minimum
2) All negative →	maximum
3) Some negative, some positive →	saddle point

Table 11: Eigenvalue Interpretation (Launsby and Schmidt 1989, 5-15)

The eigenvalues and eigenvectors for metamodel 4.1 calculated using SAS PROC RSREG are in Table 12.

	Eigenvectors					
Eigenvalues	**Cupper**	**BodyMaker**	**Printer**	**Coater**	**Necker**	**Tester**
172.96	0.02	-0.60	0.80	0.01	0.03	0.00
41.63	0.03	0.03	-0.02	0.11	0.99	0.00
18.58	0.00	-0.09	-0.07	0.99	-0.11	0.00
-13.24	0.91	0.33	0.23	0.04	-0.03	0.01
-23.51	-0.41	0.72	0.55	0.10	-0.01	-0.01
-70.02	-0.01	0.00	0.00	0.00	0.00	1.00

Table 12: Eigenvalues and Eigenvectors for Metamodel 4.1

Since there are both positive and negative eigenvalues, using Table11, this stationary point is a saddle point. Additional experimentation is needed in order to locate the local optimum. Looking at Table 12, the highest positive eigenvalue is also the largest absolute eigenvalue, 172.96. This means that the valley orientation of the surface is more pronounced than the hill orientation. The largest eigenvector across from this eigenvalue is 0.80 under the Printer, meaning that the printer has a significant influence on this valley. Additionally, the Printer and the BodyMaker are very significant in metamodel 4.1. Therefore, additional experimentation using these two inputs while keeping the others constant may yield the minimum. SAS PROC RSREG performs a gradient search along the path of steepest descent to determine at which values to fix the other variables and where to center the two variables of interest. The results of this search are in Table 13.

Radius	Scrap	Cupper	BodyMaker	Printer	Coater	Necker	Tester
0.0	140.80	0.00	0.00	0.00	0.00	0.00	0.00
0.1	130.72	0.00	0.06	-0.08	0.02	0.00	0.00
0.2	123.82	-0.01	0.12	-0.14	0.07	0.00	0.00
0.3	119.36	-0.03	0.17	-0.17	0.11	0.00	0.14
0.4	114.37	-0.03	0.17	-0.17	0.12	0.00	0.30
0.5	107.99	-0.03	0.17	-0.17	0.12	0.00	0.42
0.6	100.23	-0.04	0.17	-0.17	0.12	0.00	0.54
0.7	91.06	-0.04	0.17	-0.17	0.12	0.00	0.65
0.8	80.50	-0.04	0.18	-0.17	0.12	0.00	0.75
0.9	68.53	-0.04	0.18	-0.17	0.12	0.00	0.86
1.0	55.17	-0.04	0.18	-0.16	0.12	0.00	0.96

Table 13: Gradient Search in Direction of Steepest Descent

The minimum amount of scrap is in the last row of Table 13, 55.17. Reading across on this row indicates where to perform more experiments. Since metamodel 4.1 shows the BodyMaker and Printer as the most significant machines in determining flow scrap, more experiments need to be run changing these speeds while holding the others constant. Table 13 shows that the Cupper (-0.04), Coater (0.12), and Necker (0.00) should be set at a middle value, while Tester (0.96) should be set at a high value. Finally, additional testing should be done with the BodyMaker (0.18) and Printer (-0.16) around their middle values. Since the problem has been reduced to only two factor with three levels, a full factorial model is feasible. This design matrix along with the response is in Table 14.

Obs	BodyMaker	Printer	Scrap
1	-1	-1	137.6
2	1	-1	136.4
3	-1	1	920.4
4	1	1	163.2
5	0	0	141.2
6	0	0	141.2
7	0	0	141.2
8	0	0	141.2
9	0	0	141.2
10	0	1	210.0
11	0	-1	136.4
12	1	0	140.8
13	-1	0	169.6

Table 14: Full Factorial Design

The third observation was thrown out because it was considered an outlier. This experimental trial produced an output much larger than the others, indicating that the minimum is not in that region. The next step is to use the experimental data to determine the metamodel and calculate the response using the following model:

$$\text{Flow Scrap} = \beta_0 + \beta_1\text{BodyMaker} + \beta_2\text{Printer} + \beta_3\text{BodyMaker*BodyMaker}$$

$$+ \beta_4\text{Printer*BodyMaker} + \beta_5\text{Printer*Printer} + \varepsilon. \tag{4.2}$$

This model includes all linear effects, quadratic effects, and interaction effects for these two factors. Using SAS PROC RSREG, the parameters are calculated. The results are displayed in Tables 15 and 16.

Parameter	Estimate	Error	T	Prob > |T|
Intercept	143.195	2.685	53.326	0.0000
BodyMaker	-20.093	3.480	-5.774	0.0012
Printer	36.093	3.480	10.372	0.0000
BodyMaker*BodyMaker	7.019	4.180	1.679	0.1441
Printer*BodyMaker	-22.340	4.710	-4.743	0.0032
Printer*Printer	25.019	4.180	5.986	0.0010

Table 15: Metamodel 4.2

Regression	Degrees of Freedom	Type I Sum of Squares	R-Square	F-Ratio	Prob > F
Linear	2	2958.803	0.5714	36.537	0.0004
Quadratic	2	1041.385	0.2021	12.860	0.0068
Crossproduct	1	910.791	0.1767	22.494	0.0032
Total Regress	5	4910.979	0.9529	24.258	0.0007

Residual	Degrees of Freedom	Sum of Squares	Mean Square	F-Ratio	Prob > F
Lack of Fit	2	242.941	121.471	6.68E13	0.0000
Pure Error	4	7.276E-12	1.819E-12		
Total Error	6	242.941	40.490		

Table 16: ANOVA Table for Metamodel 4.2

The F statistic for this model is 6.68×10^{13} with a p-value of 0.0000, indicating the model has explanatory significance at $\alpha = 0.05$. The R^2 is 0.95; therefore, metamodel 4.2 explains 95% of the variation in flow scrap. Looking at the Prob > |T| column, a value less than $\alpha = 0.05$ indicates a significant factor in determining the amount of flow scrap. This shows that all of the parameters are significant except for the quadratic BodyMaker term. These extremely good results override the small sample size and degrees of freedom. This metamodel may be used as a proxy for the simulation to answer the "what if" questions about different machine speeds, provided the Cupper, Coater, and Necker are at a middle value, and the Tester is set at a high value. Metamodel 4.2 is also significantly less complicated than 4.1 in regards to the number of terms. The next step is to find the optimal combination of machine speeds that minimizes this metamodel by using response surface methodology. Table 17 shows the stationary point calculated from metamodel 4.2 using all of the coefficients.

BodyMaker	0.9790
Printer	-0.2842

Table 17: Stationary Point for Metamodel 4.2

The eigenvalues and eigenvectors for metamodel 4.2 calculated using SAS PROC RSREG are in Table 18.

	Eigenvectors	
Eigenvalues	**BodyMaker**	**Printer**
30.36	-0.43	0.90
1.67	0.90	0.43

Table 18: Eigenvalues and Eigenvectors for Metamodel 4.2

Since both eigenvalues are positive, this is a minimum. The optimal machine settings are in Table 19. The optimal value for the BodyMaker, 0.9790, is considered to be 1.00 since the BodyMaker range is not continuous.

	Optimal Setting	Actual Speed
cuppers	0.00	1800 cans per minute
body makers	1.00	1625 cans per minute
printer	-0.28	1472 cans per minute
coaters	0.00	1560 cans per minute
necker/flanger	0.00	1700 cans per minute
tester	1.00	1800 cans per minute

Table 19: Flow Scrap Minimizing Settings

Using these settings, metamodel 4.1 predicts 82 cans of flow scrap, metamodel 4.2 predicts 128 cans of flow scrap, and the simulation estimates an average of 140 cans of flow scrap per hour while increasing to 13.4 pallets per hour. This is a 36% reduction in scrap while increasing the line's throughput, and shows the improvement in predictability from metamodel 4.1 to 4.2.

4.1.10 Documentation and Reporting

A formal report and briefing were given to the plant manager at Coors, who accepted the results enthusiastically. The credibility gained by actually working the line and taking data, along with an intuitive understanding of the line, were extremely beneficial. The results were understandable and believable. The plant manager took the results for action and is ready for another study to be done on another label. He is also interested in extending the study to the 12 ounce production lines as well.

4.1.11 Implementation

Several months after completing the study, Coors verified an annual savings of $1.87 million in 1996 dollars. Coors also reinforced its willingness to allow another study of another label.

4.2 Synopsis

The most difficult portions of this simulation study were the data collection and the model translation steps. They were also the most time consuming. Over 200 hours were spent on data collection, and over 100 hours were spent on model translation, accounting for over 80% of the total project hours. The efficient experimental design was crucial to keeping the computer time to a minimum while producing the desired result - a balanced line.

Experimental design, metamodeling, and response surface methodology make an extremely powerful analysis toolbox. Solving a problem from this point of view significantly reduces the complexity found in this manufacturing system.

CHAPTER FOUR

CONCLUSION AND AREAS FOR FURTHER RESEARCH

5.1 Conclusions

If the resources are available, metamodeling of a production line through simulation is extremely insightful. Not only does it point out the key inputs of the line, but it also provides a simple algebraic equation for the response under study. The benefit of the metamodel is that anyone with a calculator can determine the level of the response given certain inputs. No knowledge of simulation, simulation languages, or computers is required. The metamodel is also significantly faster than running the simulation, a few seconds versus forty five minutes on a Pentium 100 MHz processor.

5.2 Areas for Further Research

Currently, the model assumes instantaneous speeds. For example, if the printer stops because it breaks, when it starts again it instantaneously jumps to the speed set at the beginning of the simulation. More realistically, the printer should ramp up to the speed set at the beginning of the simulation. Also, instead of shutting off instantly when the coater queue begins getting full, the printer ramps down until the coater queue gets full. This keeps the flow of cans moving while allowing the system to potentially fix itself before a machine shuts down.

Coors Light ® is the easiest and cheapest label to produce. This allows the machines to be set at high speeds. Other more decorated cans are not as easily produced and are significantly more expensive. The machines also do not run as fast for these labels, because otherwise quality will suffer. The fact that the machines run in a range slower than this study means that the inputs are outside of the current metamodel, and using the current metamodel is invalid. Therefore, the scrap generation rates should be observed at these slower speeds and new random number distributions fit. The simulation experiment should then be rerun and the metamodel recalculated.

Finally, an experiment may be run with two printers instead of one. This would allow the printers to run at slower speeds, maintaining a high level of quality, while increasing the throughput at the printer area. This may prove to be significant because the printer is definitely a bottleneck for the line. It is fed by up to fifteen body makers and feeds up to nine coaters.

REFERENCES CITED

Banks, Jerry, John S. Carson III, and Barry L. Nelson. 1996. Discrete-event system simulation. Upper Saddle River: Prentice Hall.

Carraway, Robert L. 1989. A dynamic programming approach to stochastic assembly line balancing. Management Science. 35 (4): 459-471.

Davis, K Roscoe and Bernard W. Taylor. 1974. Want to reduce WIP inventory? Try linear line-balancing. Production and Inventory Management. 15 (2): 57-65.

Dilworth, James B. 1996. Operations Management. New York: McGraw-Hill

Gagnon, Roger J. and Soumen Ghosh. 1989. A comprehensive literature review and analysis of the design, balancing and scheduling of assembly systems. International Journal of Production Research. 27 (4): 637-670.

Geoffrion, Arthur M. 1976. The purpose of mathematical programming is insight, not numbers. Interfaces. 7 (1): 81-92.

Gutjahr, Allan L. and George L. Nemhauser. 1964. An algorithm for the line balancing problem. Management Science. 11 (2): 308-315.

Launsby, Robert G. and Stephen R. Schmidt. 1989. Understanding industrial designed experiments. Colorado Springs: Academy.

Levitin G. and J. Rubinovitz. 1995. Genetic algorithm for assembly line balancing. International Journal of Production Economics. 41: 343-354.

Palisade. 1995. Bestfit user's guide. Newfield: Palisade.

Pritsker, A. Alan B. 1995. Introduction to simulation and SLAM II. New York: Wiley.

REFERENCES CITED (CONT.)

Watson, Edward F. and Alan S. Wood. 1995. Mixed-model production system design using simulation methodology. Production and Inventory Management Journal. 36 (4): 53-58.

SELECTED BIBLIOGRAPHY

Goldratt, Eliyahu M. 1992. The goal. Great Barrington: North River.

Law, Averill M. and W. David Kelton. 1991. Simulation modeling and analysis. New York: McGraw-Hill.

Long, Julia B. 1993. Metamodeling utilizing d-optimal designs as applied to large scale simulations. Ph.D. dissertation. Colorado School of Mines.

Montgomery, Douglas C. 1984. Design and analysis of experiments. New York: Wiley.

SAS Institute, Inc. 1987. SAS user's guide; statistics, version 6. Cary: SAS Institute.