1-1-2007

Throughput analysis and improvement of paint shop in automobile industry.

Guangming Qiu
University of Windsor

THROUGHPUT ANALYSIS AND IMPROVEMENT OF PAINT SHOP IN AUTOMOBILE INDUSTRY

By

Guangming Qiu

A Major Paper
Submitted to the Faculty of Graduate Studies and Research
Through Industrial Engineering
In Partial Fulfillment of the Requirements for
the Degree of Master of Applied Science
at the University of Windsor

Windsor, Ontario, Canada
2007

Your file Votre référence
ISBN: 978-0-494-42319-6
Our file Notre référence
ISBN: 978-0-494-42319-6

ABSTRACT

It is natural that if a thing happens two times under similar conditions, it is likely to get the same result. Based on this assumption, the simulation model of a paint shop is built by software AutoMod to generate an artificial history to draw inferences about the operating characteristics of the real system. This model considers many factors that could affect the throughput, such as mixed products, optimal buffer location, varying part arriving rate, mean time between failures, mean time to repair, and maintenance policy. Using simulation as modelling techniques is a general approach to the complex situation where analytical methods have limitation.

DEDICATION

This major paper is dedicated to my family: Hong Qiu, Jie Sun, Yanqiu Gao, Shu Qiu

ACKNOWLEDGEMENTS

I feel fortunate to have had a lot of people who helped me completing the program of Master of Applied Science. I would like to take the opportunity to express my sincere gratitude to Dr. Walid Abdul-Kader, my academic advisor, for his encouragement, patience, invaluable advice through the course of writing this major paper.

My gratitude also goes to Dr. R. Lashkari, who acted as my committee member, for the time he spent reading and examining this major paper. His valuable suggestions helped significantly improve the quality of my work.

Sincere gratitude is also expressed to the faculty and staff of the Department of Industrial and Manufacturing Systems Engineering for providing me with such an excellent research experience.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

Buffer – is used to store loads on semi-finished product

Mean time between failures (MTBF) – this is a quantity of time from which an object in working condition will fail

Mean time to repair (MTTR) – this is a quantity of time required to repair an object which is currently offline

Steady-state – is the limit of a response variable of a simulation model if the simulation model were run without termination

Transient state – the period after which steady-state is reached

Corrective maintenance – occurs when a system accidently fails and is usually driven by the failure of a component or system

Preventive maintenance – repair and maintenance of the facilities every certain period

White body storage (WBS) – is used to store white body before a paint shop

Painted body storage (PBS) – is used to store painted body after a paint shop

System configurations – are ways to deploying facilities

Cycle time – the number of time units per product produced on the line

CHAPTER I: INTRODUCTION

This paper studies a paint shop in the automobile industry. There are five main shops whose work is required to produce a car: the stamping shop, the body shop, the paint shop, the trim and chassis shop and the power train shop. The flow of work from stage to stage is illustrated in Figure 1.1 (with WBS standing for "white body storage" and PBS for "painted body storage").

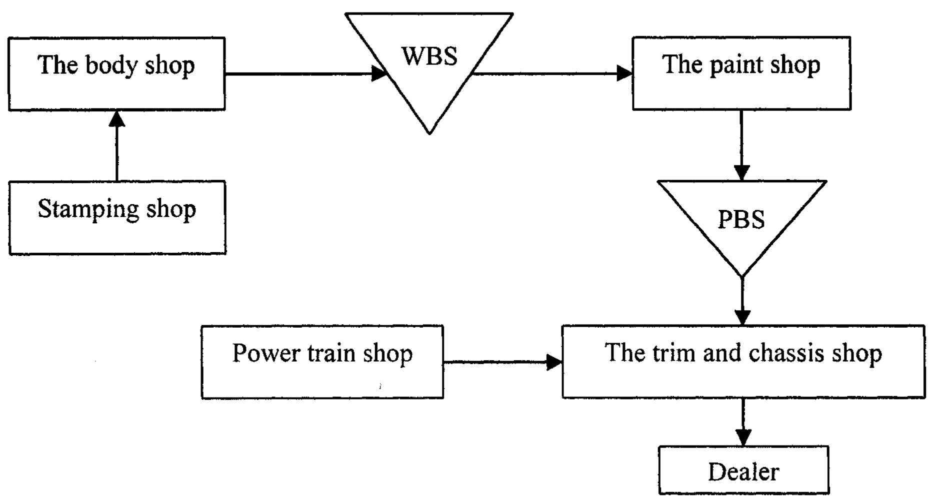


Figure 1.1: Automobile Manufacturing System

Conway et al. (1988) presented six rules for the optimal buffering of balanced lines with moderate variability. One of these rules is the bowl phenomenon, which states that buffers should be allocated evenly among all sites—if possible—with any remaining buffers allocated symmetrically around the center of the line. Furthermore, Powell and Pyke (1996) stated that while a bottleneck station tends to draw buffers toward it, the optimal allocation depends on the location and severity of the bottleneck, as well as the number of buffers available. Therefore, if we consider the entire automobile manufacturing system as a production line, the paint shop—located in the middle of said line—is likely to act as a bottleneck.

The paint shop performs a series of processing steps to prepare and then coat the car body surfaces according to the quality requirements. Usually, a sedan demands higher painting quality than a truck and, therefore, requires more processes. Defects are common in a paint shop, and painted cars with minor defects are taken off the regular production lines and moved to the repair area. Painted cars with major defects are directed to the rework loops to be processed again. Once the repair is finished, they are placed back on the regular line as soon as an opening occurs.

Multiple product types with a variety of surface areas are often processed in the same painting line. A car with a larger surface area requires more processing time than one car with a smaller surface area—facts that prompt varying processing times. Color changing is another common occurrence in the paint shop, i.e. mixed products. When changing color, a specific set up time is required for some workstations. For example, the plant must cleanse the painting apparatus of one paint color before switching to a new color.

In order to save on the investment in the painting facility and increase painting quality at the same time, jobs can enter the painting booths multiple times. For example, products with major defects are directed back to particular painting booths to be reprocessed, otherwise another painting booth must be built to perform the repair function. Therefore, Li (2004) stated that paint shops tend to be system bottlenecks in many automotive assembly plants due to the complexity of their tasks. This paper selects a paint shop—starting from WBS and ending at PBS—as a case study.

This is the first time that the two kinds of system configurations are thoroughly compared in terms of throughput, and a new equation for predicting optimal buffer location is introduced in Chapter II and validated in our case study.

The remainder of this major paper is divided into five more chapters. Chapter II presents analytical solutions to the throughput calculation, as well as their strong points and limitations. Techniques to improve throughput, such as suitable system configuration, optimal buffer location and an efficient maintenance policy are also offered.

Chapter III focuses on the paint shop, which consists of a material handling system, various work booths and resources such as workers and/or robots. The material handling system includes various power and free conveyors and carriers (fixtures or skids) to perform the transportation, buffer and grouping functions. One of the main problems encountered by designers is the difficulty of gaining insight into how the number of carriers impacts the throughput. In addition, random failures and the repair time of the paint booths also affect the system throughput.

Chapter IV—Simulation Modeling—introduces the primary purpose of the simulation modeling techniques, the two base models, the different factors to be tested and the simulation tools.

Chapter V sets up several experiments to analyze the effect of different factors on system performance. A modeller could sense how the number of carriers affects the throughput based on the experimental results. Buffer location is also an important design factor. The impact of maintenance on the throughput is addressed as well. Ultimately, the final throughput is presented when all the suitable techniques are applied to the manufacturing system.

Finally, the conclusion and recommendations for future research are provided in chapter VI.

CHAPTER II: LITERATURE REVIEW

In an effort to gain insight into the throughput estimation for the whole manufacturing system, several typical modules are presented—including how to analyze single workstation throughput as well as two workstations in a series, separated by a buffer. Theoretically, the throughput of a complex manufacturing system could be calculated by decomposing the complex manufacturing system into many simple templates. Several techniques for improving throughput are presented later, some of which are applied to our case study.

2.1 Throughput estimation of the single workstation

For the single workstation, Sawyer (1970) gave a well-known relation—see equation 2.1—for predicting the throughput:

$$Q = \frac{T}{C} \tag{2.1}$$

where:

- C = cycle time (the number of time units per product produced on the line)
- T = useful processing time
- Q = throughput

However, further analysis reveals that this equation is not as simple as it initially appears. How does the designer calculate useful processing time? For example, given a one year period, the designer can reduce the time for holidays, for breaks and for repairing or setting up the facility, but how can they estimate the time for starting the production line and the time for stopping it—during which the production rate of the manufacturing system is apparently slowing down? In reality, the designer cannot ignore this issue because the factory manager encounters it whenever there is shift change. This problem is revisited in Chapter III, with our research trying to provide a solution according to the paint shop's special rules.

2.2 Throughput estimation of two workstations in series separated by a buffer

Alden (2002) developed a formula estimating the performance of two workstations in series with downtime and unequal speeds, as illustrated in Figure 2.1. In this configuration, jobs flow from Station 1 to Station 2, and are to be processed at each station. To derive equation 2.2, which calculates the throughput in a steady state, Alden (2002) treats the movement of jobs through the line as a fluid flow.

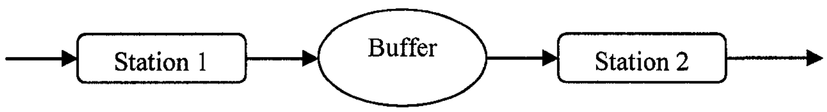


Figure 2.1: Two workstations in series separated by a buffer

The throughput equation developed for this case is given by

$$\rho = S_2 \times P(w,w) + \hat{S}_2 \times P(d,w) \qquad (2.2)$$

where:

- ρ = the average number of jobs produced per time unit
- S_2 = the speed of station 2, i.e. jobs produced per time unit
- $P(w, w)$ = the fraction of time that both stations are processing
- $P(d, w)$ = the fraction of time that station 2 is processing, while station 1 is down
- $$\hat{S}_2 = S_2 \times \frac{MTBF_2}{MTBF_2 + MTTR_2} \qquad (2.2a)$$

Based on the research conducted by Alden (2002),

$$P(w,w) = \frac{1}{(\lambda_1 + \lambda_2)\left[1/(\lambda_1 + \lambda_2) + B\theta_2 P_0/\hat{S}_1 + B\theta_2 P_0/\hat{S}_2 + \theta_2 P_0/\theta_1\mu_2 + P_0/\mu_2 \right]} \qquad (2.3)$$

and

$$P(d,w) = \frac{\hat{S}_2}{(B\theta_2 P_0)\left[1/(\lambda_1 + \lambda_2) + B\theta_2 P_0/\hat{S}_1 + B\theta_2 P_0/\hat{S}_2 + \theta_2 P_0/\theta_1\mu_2 + P_0/\mu_2 \right]} \qquad (2.4)$$

where,

- B = the buffer size
- μ_1 = 1/$MTTR_1$ (mean repair rate of workstation 1)

- λ_1 = 1/MTBF$_1$ (mean failure rate of workstation 1)
- μ_2 = 1/MTTR$_2$ (mean repair rate of workstation 2)
- λ_2 = 1/MTBF$_2$ (mean failure rate of workstation 2)
- $\theta_1 = \lambda_1/(\lambda_1 + \lambda_2)$ (for simplifying the above equation)
- $\theta_2 = \lambda_2/(\lambda_1 + \lambda_2)$ (for simplifying the above equation)
- $P_0 = \theta_1/(1 + \theta_1 B)$ (the probability that the buffer is empty)
- $\hat{S}_1 = S_1 \times \dfrac{MTBF_1}{MTBF_1 + MTTR_1}$

Alden's paper also reveals that Equations 2.3 and 2.4 are based on many assumptions—that "time between failures" and "time to repair" are exponentially distributed, that the first station is never starved and the second station is never blocked, and that jobs flow through the system with zero transit time. With so many assumptions (some of them impossible in a real-world situation), it is inconvenient to give an analytical solution, even if the manufacturing system is slightly modified. To remedy this problem, another method is wanted as the manufacturing system becomes more complex.

2.3 Increasing line's throughput by system configurations

Webbink and Hu (2005) stated that the system configuration could be divided into five categories: (a) serial configuration, (b) parallel configuration, (c) simple hybrid configuration, (d) complex hybrid configuration and (e) complex-hybrid configuration—as illustrated in Figure 2.2, where the little rectangle represents a workstation.

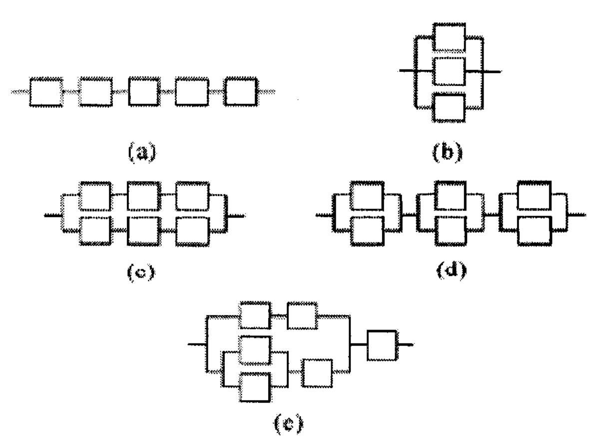


Figure 2.2: System configurations

These configurations, individually, have their own advantages and disadvantages. For example, the serial configuration is simple and does not require too many material handling facilities and the parallel configurations are more reliable. When one workstation breaks down, the other—possessing the same function—can keep the production line running. However, parallel configurations demand a complex material handling system and therefore waste too much time in the process of transportation, which does not add value to the product. Therefore, a factory manager is faced with a trade-off when selecting a suitable system configuration.

The studied case in this paper selects the simple hybrid configuration shown in Figure 2.2 (c) as its strategy for deploying equipment because several kinds of products are mixed together. Serial configuration requires a particular set-up time when changing product types, leading to a reduction in the over-all throughput. Furthermore, because most manufacturing systems now adopt a pull system instead of a push system and the market demand is always varying, grouping the same products as a batch is sometimes not practical. In this situation, a simple hybrid configuration could perform the grouping function by the material handling system.

In our case, a "cross-line transportation car" is the material handling equipment used to convey the first group of products to the upper serial line and the second group of products to the lower serial line (i.e. "grouping"). Both the upper and lower serial lines perform the same sequence of processes.

In addition, from a reliability standpoint, a simple hybrid configuration is better than a serial configuration. For example, if one of the workstations in the upper serial line breaks down, the factory manager can continue to run the production line as a whole by using the lower serial line. Failures in both the upper and lower serial production lines, however, causes the production to stop completely.

Chapter V studies and compares the serial configuration and the simple hybrid configuration when the types of product increase from 1 to 8.

2.4 Contribution of properly locating buffer

Powell and Pyke (1996) studied the problem of buffering serial lines with moderate variability and a single bottleneck—a single station with a larger mean processing time than all other stations. Their analysis reveals that a bottleneck station draws buffers towards itself, but the optimal allocation depends on the location and severity of the bottleneck, as well as the number of buffers available.

In our case, a simple hybrid configuration is adopted as our deploying strategy. A restricted factory space prompted us to allow two buffers—each with a capacity of 1—in each line of the simple hybrid configuration. Therefore, the designer does not worry about the buffer capacity, only the buffer location. This makes the buffer analysis much easier.

The mean processing time for each workstation in our case is almost the same as the cycle time. The main difference between the various workstations is their mean time between failures, and mean time to repair. The spraying guns in the paint booth are very sensitive to the density of paint material, and frequently become blocked. Therefore, the painting booths have a much lower mean time between failures.

However, the conclusions, drawn by Powell and Pyke (1996), could be expanded by the assumption of replacing a larger mean processing time with a slower speed (see Equation 2.2 a) in order to conceptually predict a bottleneck in a serial production line. In addition, Equation 2.5 expresses the workstation's parameter for mixed products if the set up time is ignored.

$$S = \frac{1}{n_1 t_1 + n_2 t_2 + \cdots n_n t_n} \times \frac{MTBF}{MTBF + MTTR} \tag{2.5}$$

Where,

- n_n = the number of nth product
- t_n = the time for producing one of nth product

Equation 2.5 is used to conceptually predict a serial line's bottleneck in the simple hybrid configuration, and is validated by the simulation model. For more information, please refer to Experiment IV in Chapter V.

2.5 Increasing throughput with efficient maintenance policy

Maintenance activities are performed to enhance or restore efficiency and alleviate the risk of losing throughput. Zhang (2005) presented that maintenance activities can be categorized as corrective maintenance (CM) or preventive maintenance (PM). CM occurs when a system experiences a random failure—usually driven by the failure of a component or system. PM occurs when a system can still run and be performed in either a time-based or a condition-based manner.

The challenge the designer faces is not knowing which maintenance policy is more efficient, because efficiency depends on the reliability of the components and the system, or the failure rate curve. In addition, preventive maintenance can only reduce a facility's failure rate to a certain level, it cannot effectively prevent accident failures. This makes selecting a suitable maintenance policy more complex, because the levels depend on the frequency that preventive maintenance is performed. Chapter V compares data provided by the maintenance crew, corrective maintenance and time-based preventive maintenance in terms of throughput.

CHAPTER III: PAINT SHOP REVIEW

This chapter introduces the paint shop in our case study, presenting an overview of the paint-process as well as the layout and operation of the shop.

3.1 Process overview

The white bodies are transported from the automotive body shop to the paint shop, where they undergo pre-paint treatment. —a process that involves thorough washing and phosphate, which is used to cause a chemical crystallization on the white body surface that provides improved paint adhesion and anti-corrosion protection.

The white bodies dry, and are directed through the process of base-coat application—a layer of material applied to the vehicle surface that causes the top-coat to readily adhere to the surface of the vehicle. The base-coat material requires excellent adhesion to both the automotive white body—generally made of lightweight steel—and the top-coat painting material.

The truck bodies are then transported to a bake oven where the base-coating is cured and dried, prior to being directed to the sanding booth where the surface of the base-coat is made coarser to improve its adhesive ability. Forced cooling is provided by the cooling-booth between the bake-oven and the sanding-booth to lower the temperature of the truck bodies to around 45 °C.

When the top-coating process is finished—one similar to the base-coating process—the painted bodies undergo inspection and repair in terms of paint quality and correctable defects.

3.2 Layout and facility

As previously stated, the facilities are finally deployed according to a simple hybrid configuration—illustrated in the conceptual layout of Figure 3.1—where the squares represent various facilities and the stars signify the candidate buffer locations. Grouping function is performed by the cross-line transportation cars, which separate groups of products into a variety of serial production lines. Only two Capacity 1 buffers are allowed in each parallel production line, due to space limitation and investment-efficiency on the paint shop floor. The size and shape of the truck bodies—approximately ten meters long and two meters high whether they are stacked or stored on the same level—the buffer occupies too much space in the paint shop. Chapter V addresses the optimal buffer location.

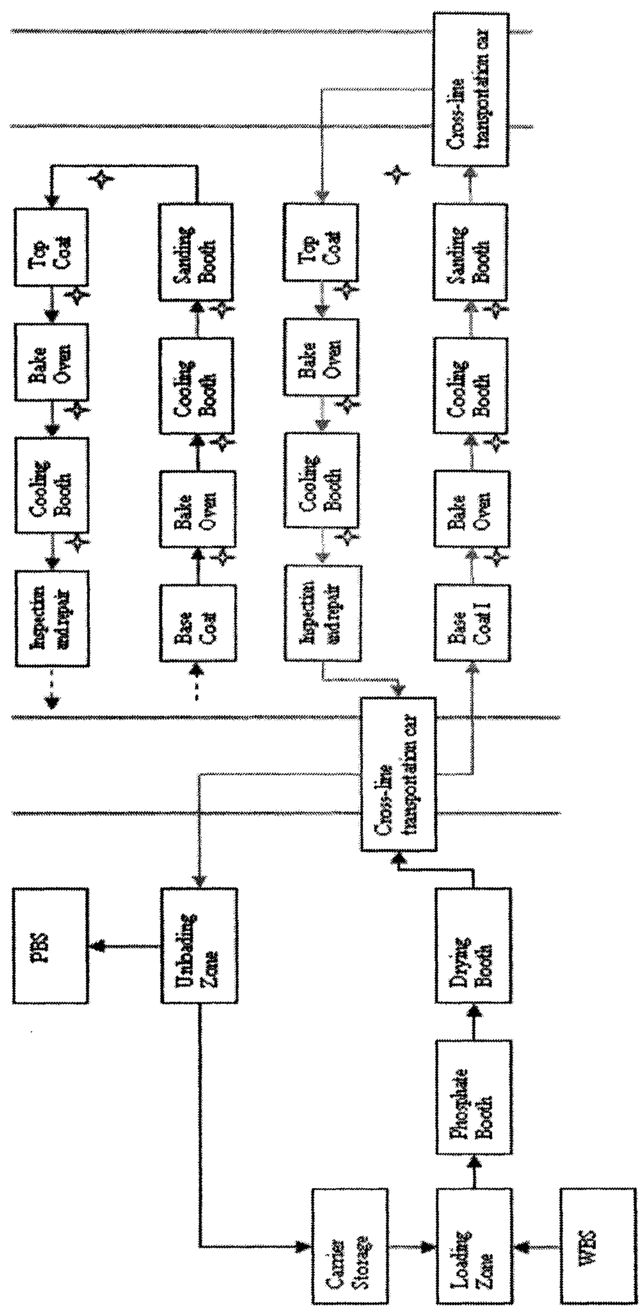


Figure 3.1: Layout of paint shop

Other workstations—except the phosphate and drying booths—are doubled because the production rates of the phosphate and drying booths are much faster than the rates of the other workstations. In addition, the phosphate and dry-up processes are similar even for different products—unlike other workstations, such as base-coat and top-coat, which must change the performance of their tasks with the changing product types. The cross-line transportation car is the material handling equipment used to perform the grouping function. According to PLC code, it can separate different products and send them to one of the base-coat booths, based on the "first come first serve" principle.

White truck bodies are only placed on carriers in the loading zone before they are conveyed to the next process if two conditions have been met. The first is that there are white truck bodies in the WBS. The second is that carriers are available in the carrier storage. Once the entire painting process is complete, the truck bodies are transported to the unloading zone where they separate with the carriers. The workers then put the carriers in the carrier storage, and the painted bodies in PBS.

3.3 Paint shop operation

The paint shop runs eight hours a day, five days a week. During its daily operation, production must be scheduled so that end-shift changes can be made with a minimum of interruption in production. Scheduling the end-shift is difficult because the conveyors running parts through bake ovens cannot be stopped for long, or they risk inflicting damage resulting from the high temperature. With this in mind, the manager empties all the bake ovens before shift, leading to a temporary slow down for the throughput of the whole system. The end-shift operation routine is illustrated in Figure 3.2, and this special rule makes the calculation of useful processing time in Equation 2.1 very complex. Chapter V includes the shifting impact in the simulation model to accurately estimate the performance of the whole system.

To prevent paint build-up on spray gun tips, each paint-delivery hose and spray gun are purged with solvent between each change in color. Despite these efforts, the spray guns are often blocked as a result of an inefficient purge—prompting the base-coat and top-coat booths to suffer a high fraction of down time. Consequently, they test CM and PM to minimize the impact caused by the facility's various fractions of down time.

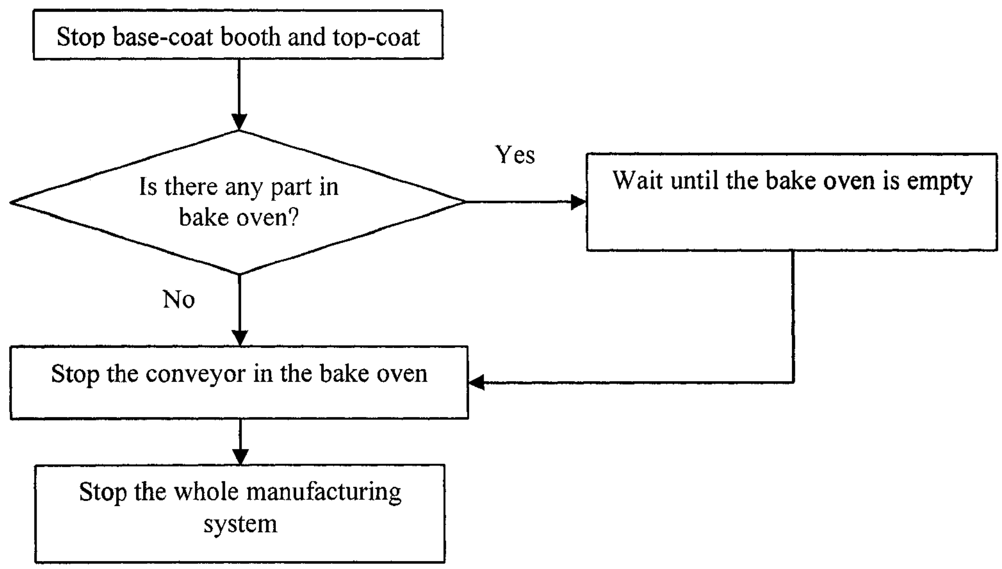


Figure 3.2: End-shift operation routine

CHAPTER IV: SIMULATION MODELING

The first step in designing the simulation model was to identify its primary purpose. In our case, the primary purpose of simulating the manufacturing system was to estimate the throughput and test different factors that might increase throughput before the final layout is applied to the shop floor. Two base models were built to compare serial system configuration and simple hybrid system configuration. After selecting a suitable system configuration, different factors with potential effects on the throughput were tested. These factors are the number of carriers, buffer locations and maintenance policies.

4.1 Serial system configuration

Base Model I was built to estimate the throughput of a serial system configuration, and then compared with the simple hybrid system configuration estimated by Base Model II. The flow chart of Base Model I is illustrated as Figure 4.1.

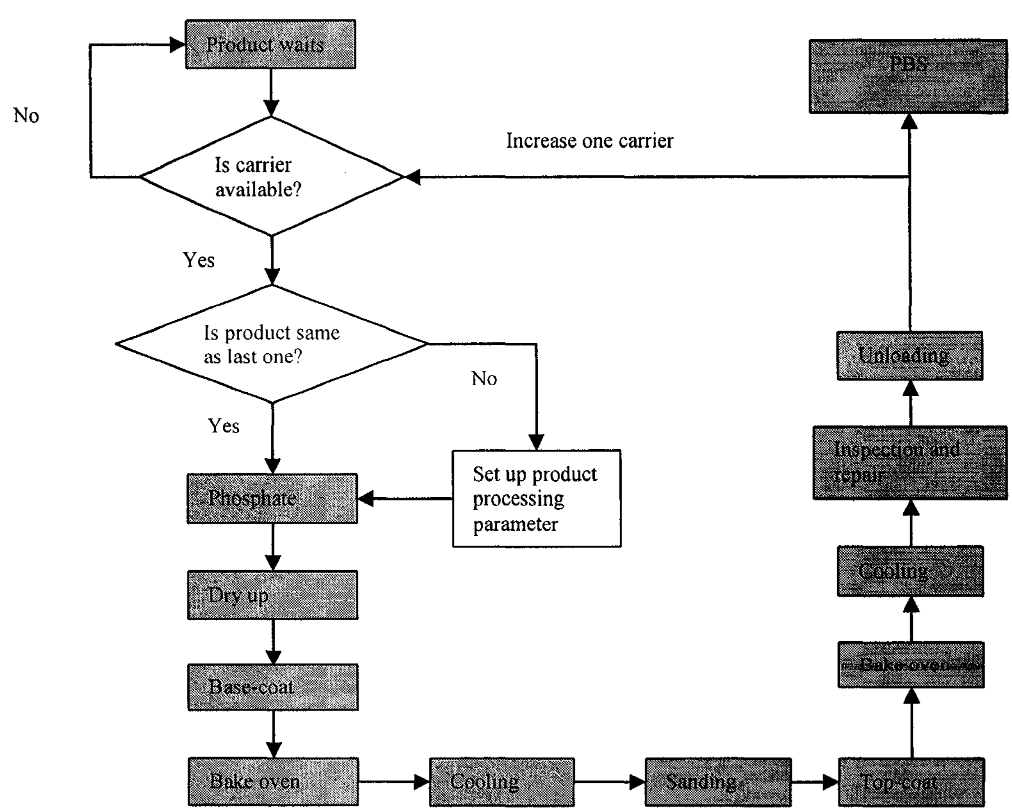


Figure 4.1: Flow chart of Base Model I, serial system configuration

4.2 Simple hybrid system configuration

Base Model II is built to estimate the throughput of a simple hybrid system configuration and then compared with the serial system configuration estimated by Base Model I. The flow chart of Base Model II is illustrated in Figure 4.2.

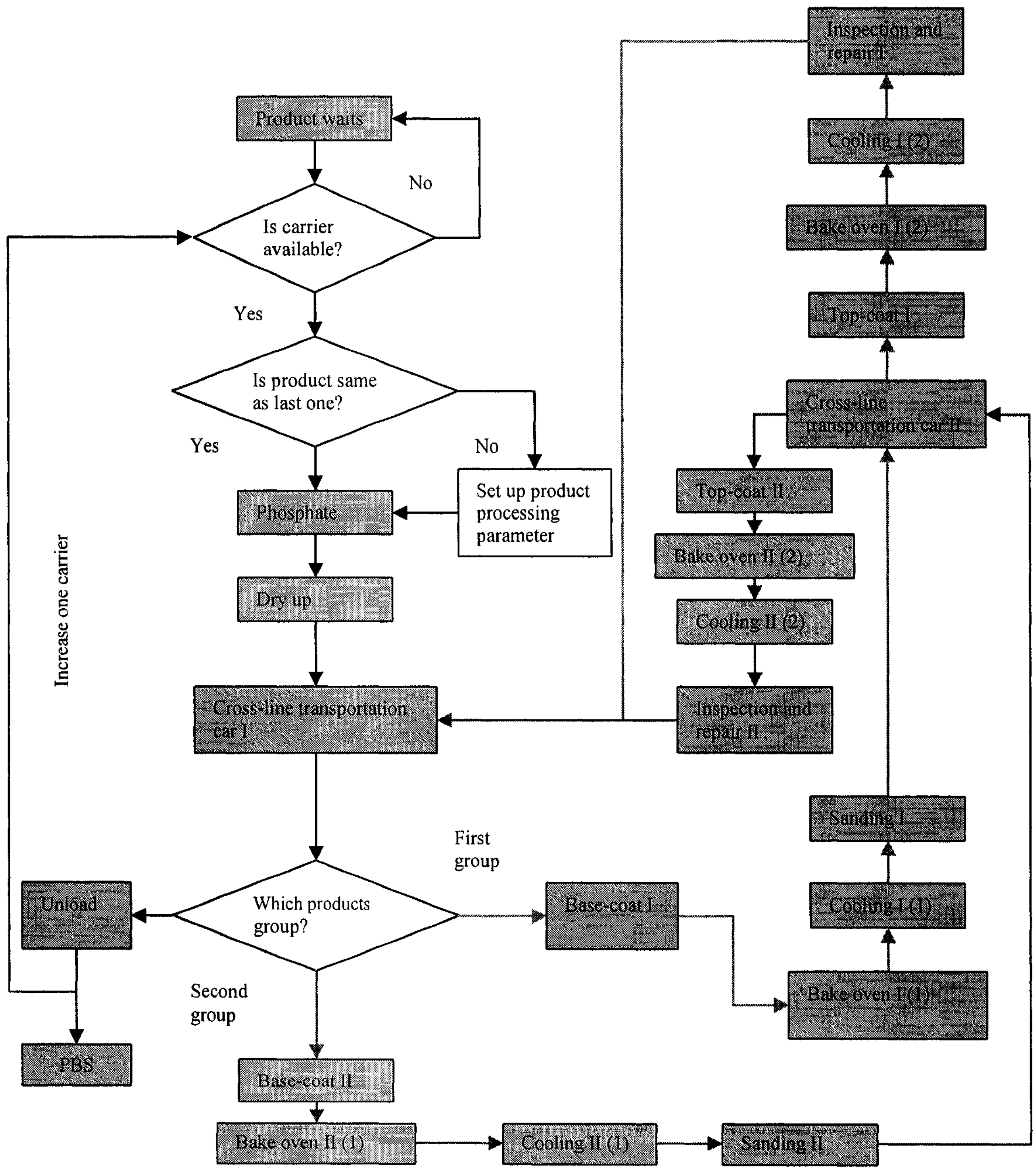


Figure 4.2: Flow chart of Base Model II, simple hybrid system configuration

4.3 Factors to be tested

After a suitable system configuration is selected, various factors such as the quantity of carriers, buffer locations and maintenance policies—all of which have a potential influence on the final throughput—are tested by the simulation of Base Models I and II.

4.4 Simulation tools and notation in simulation code

The simulation models were built using AutoMod—a commercial simulation package that combines CAD-like drawing tools with a powerful engineering-oriented language to model control logic and material flow. Unlike most other simulation languages, AutoMod's strong graphical interface precisely captures the physical constraints of distance, size and space—producing exceptionally accurate 3-dimensional details.

To help readers understand the simulation code, a summary of the notation is presented below:

- V1_... = variable used for product 1
- V2_... = variable used for product 2
- V3_... = variable used for product 3
- V4_... = variable used for product 4
- V5_... = variable used for product 5
- V6_... = variable used for product 6
- V7_... = variable used for product 7
- V8_... = variable used for product 8
- V_... = variable used for all products
- A_... = load attribute
- R_... = resource
- P_... = process
- L_... = load
- Conv = power and free conveyor system
- Pm = automatic guided vehicle system
- Sta_... = workstations
- Lbl_... = labels
- Ol_... = order list

CHAPTER V: SIMULATION EXPERIMENTS AND RESULTS ANALYSIS

Chapter V is divided into six sections. The first section defines the warm-up period using a leading base model, then applies the warm-up period and the steady-state length to the subsequent simulation models. The second section compares the simple hybrid system configuration with the serial system configuration in terms of throughput when the number of product types varies. The third section determines the optimal number of carriers. The fourth section uses a conceptual simulation model to validate Equation 2.5, which predicts the optimal buffer location. The result is then applied to arrange the facilities on the shop floor. The fifth section addresses factory management—with the maintenance policies considered in this experimental design. The last section predicts the throughput per month when the end-shift change is considered.

5.1 Determining warm-up period

Most of the simulation models are started empty and idle. Almost every time, these conditions differ from the steady-state condition. Therefore, the simulation model takes some time to reach steady-state. During this time period, the model is said to be in transient-state, i.e. warming up.

Mahajan and Ingalls (2004) categorized the methods for dividing the warm-up period into (1) graphical, (2) statistical, (3) heuristics and (4) initialization bias methods. In this paper, the graphical method is employed to define the warm up period using a leading base model. All the following experiments are analyzed with data recorded from the simulation steady-state behavior.

- **Introduction of Experiment I**

The leading base model is designed to define the warm-up period. Eight kinds of products are used to test this leading base model.

- **Input data for Experiment I**

Table 5.1 presents the input data for Experiment I. Column 1 reports the name of different workstations. In the table, all the cycle times are the uniform distribution. And the first value is the mean of the distribution, while the second value is the standard deviation. By using uniform distribution, the maximum and minimum processing times are limited that accords to the reality.

	Cycle time in minutes								
Product	1	2	3	4	5	6	7	8	Set-up time in minutes
Phosphate	5, 1	5.2, 1	5.5, 1	5.6, 1	6.0, 1	6.2, 1	6.4, 1	6.6, 1	7
Dry up	4.8, 1	5, 1	5.2, 1	5.4, 1	5.5, 1	5.5, 1	6, 1	6.4, 1	8
Basecoat I and II	10.4, 3	10.6, 3	11.2, 3	12.4, 3	12.6, 3	12.8, 3	13, 3	13.2, 3	6
Bake oven I (1 and 2) and II (1and2)	10.2, 2	10.4, 2	11, 2	12, 2	12.3, 2	12.6, 2	12.8, 2	13, 2	6
Cooling I (1 and 2) and II (1and2)	10.2, 1	10.4, 1	10.9, 1	11.8, 1	12.4, 1	12.7, 1	12.7, 1	12.9, 1	7
Sanding I and II	10, 1	10.2, 1	11, 1	11.8, 1	12.3, 1	12.5, 1	12.8, 1	12.8, 1	8
Topcoat I and II	10.3, 3	10.5, 3	11.3, 3	12.3, 3	12.5, 3	12.8, 3	13.1, 3	13.4, 3	8
Inspection and repair I and II	10.1, 1	10.3, 1	11.1, 1	12.1, 1	12.2, 1	12.6, 1	13, 1	13.2, 1	6
Unloading	5, 1	5.1, 1	5.4, 1	5.3, 1	5.8, 1	6, 1	6, 1	6.6, 1	N/A
Product mix-ratio	1/N	1/N	1/N	1/N	1/N	1/N	1/N	1/N	N:1~8

Table 5.1: Input data for Experiment I

- **Simulation results and analysis of Experiment I**

Figure 5.1 is based on the data from Experiment I. The X-axis in Figure 5.1 represents the time in hours, while the Y-axis tells us the throughput per hour. Figure 5.1 suggests that the manufacturing system enters the stable-state after 20 hours.

For the sake of insurance, the warm-up period is enlarged to 50 hours, and the simulation model is run for 1000 hours after the first 50 hours. During these 1000 hours, the simulation data are collected and analyzed to compare serial system configuration with simple hybrid system configuration and determine the optimal number of carriers, predict the optimal buffer locations and choose a suitable operation policy.

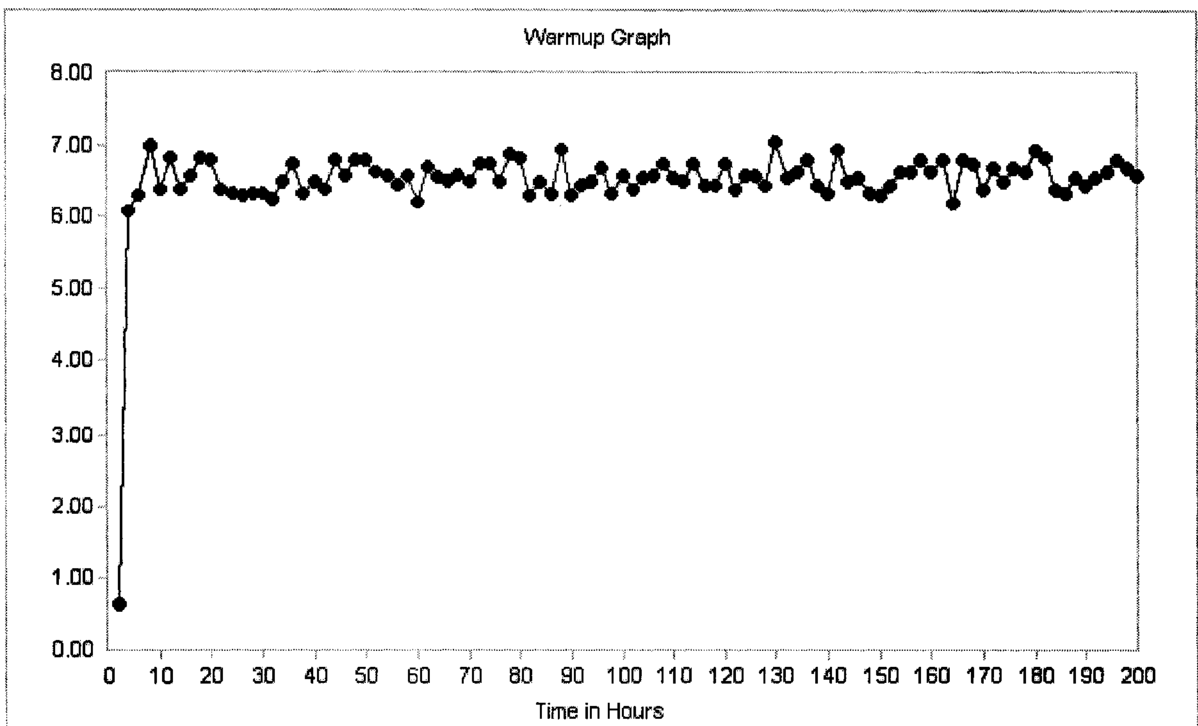


Figure 5.1: Warm-up period

5.2 Serial system configuration versus simple hybrid system configuration

The first step in designing a manufacturing production line is to choose a strategy for deploying the facilities. The truck paint shop has two common system configurations. The first is a serial system configuration, as shown in Figure 2.2 (a). The second is a simple hybrid system configuration, as illustrated in Figure 2.2 (c).

Which system configuration the designer finally chooses depends on a variety of restrictions. For example, if the truck paint shop floor is long and narrow, a serial system configuration is a better choice because the final shape of a serial system configuration on the shop floor usually appears as a slot that maximizes the floor's utilization.

Another important restriction is the investment. More capital expenditure means a highly-automated manufacturing system where a central control room could be set up to supervise and direct the material flow. This would allow the cross-line transportation cars to distinguish different products and convey them to the appropriate serial production lines according to PLC code, i.e. grouping.

There are two major benefits of grouping. The first is that it basically balances the different production lines according to the products' cycle times. For example, there are four types of products produced in two identical production lines. The cycle times of the four types of products are 4.2, 6.3, 8.5 and 10.6 minutes, separately. The material handling system can group the products that have 4.2 and 10.6 minute cycle times together in one production line, and the other two products into another production line—resulting in two balanced production lines with near 14.8-minute cycles. The other benefit is that it saves set-up time, as mentioned in Section 2.2.

- **Introduction of Experiment II**

The product goes through the same kind of process whether it's in the serial or simple hybrid system configurations. However, except for the phosphate, dry-up and unloading workstations, the number of other workstations in the simple hybrid system configuration are doubled when compared with the serial system configuration—as illustrated in Figures 4.1 and 4.2. As a result, the production rate of the workstations in the simple hybrid system configuration should be twice as slow as the production rate in the serial system configuration. Apparently the equipment with a fast production rate is generally more complex and expensive than the equipment with a slow production rate. Consequently, the set-up times for the workstation in the serial system configuration are also doubled, when compared with the simple hybrid system configurations—as illustrated in Tables 5.2 and 5.4.

This paper selects a suitable system configuration based on throughput. Two separate simulation models are built to predict the throughputs of the serial and the simple hybrid system configurations, according to a certain number of product types. At this level, the modeller does not need to consider the impact that the quantity of carriers, buffer location, random failures and maintenance policies have on the throughput—a fact that dramatically simplifies the simulation models.

- **Input data for the serial system configuration**

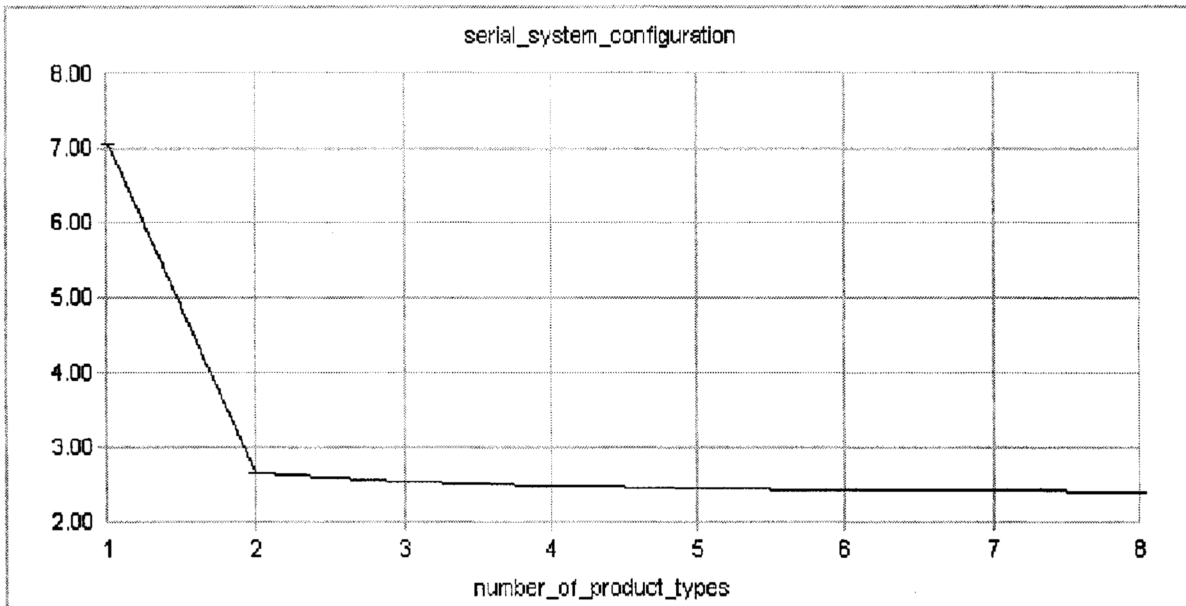
Table 5.2 represents the cycle times of different products in different workstations in minutes. All the cycle times are uniform distribution. For example, the cycle time of Product 4 at the phosphate workstation is an uniform distribution with a mean of 5.6 and a standard deviation of 1.

	Cycle time in minutes								
Product	1	2	3	4	5	6	7	8	Set-up time in minutes
Phosphate	5, 1	5.2, 1	5.5, 1	5.6, 1	6.0, 1	6.2, 1	6.4, 1	6.6, 1	7
Dry up	4.8, 1	5, 1	5.2, 1	5.4, 1	5.5, 1	5.5, 1	6, 1	6.4, 1	8
Basecoat	5.2, 3	5.3, 3	5.6, 3	6.2, 3	6.3, 3	6.4, 3	6.5, 3	6.6, 3	12
Bake oven 1 and 2	5.1, 2	5.2, 2	5.5, 2	6, 2	6.15, 2	6.3, 2	6.4, 2	6.5, 2	12
Cooling 1 and 2	5.1, 1	5.2, 1	5.45, 1	5.9, 1	6.2, 1	6.35, 1	6.35, 1	6.45, 1	14
Sanding	5, 1	5.1, 1	5.5, 1	5.9, 1	6.15, 1	6.25, 1	6.4, 1	6.4, 1	16
Topcoat	5.15, 3	5.25, 3	5.65, 3	6.15, 3	6.25, 3	6.4, 3	6.55, 3	6.7, 3	16
Inspection and repair	5.05, 1	5.15, 1	5.55, 1	6.05, 1	6.1, 1	6.3, 1	6.5, 1	6.6, 1	12
Unloading	5, 1	5.1, 1	5.4, 1	5.3, 1	5.8, 1	6, 1	6, 1	6.6, 1	N/A
Product mix-ratio	1/N	1/N	1/N	1/N	1/N	1/N	1/N	1/N	N:1~8

Table 5.2: Input data for serial system configuration

- **Simulation results and analysis of the serial system configuration**

Figure 5.2 is drawn according to the simulation result explicitly in an effort to analyze it. For the experimental data details, refer to Appendix D, where eighty runs are performed. For each number of product types, ten replications are performed to get the statistical average throughput per hour. There are eight types of product, in total, to be tested in the manufacturing system. Therefore, eighty runs are performed. In Figure 5.2, the X-axis represents the number of product types, while the Y-axis indicates the throughput per hour.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
number of product types	1	2	3	4	5	6	7	8	0	0	0	0	0	0	0	0	0	0	C
throughput_per_hour	7.07	2.67	2.54	2.49	2.45	2.43	2.42	2.40											

Figure 5.2: The effect of varying the number of product types on throughput in serial system configuration

It is reasonable to state that the throughput decreases as the product type increases, due to the time spent on set-up. When there is only one type of product, no set-up is needed and the percentage of time spent on set-up is zero, as illustrated in Table 5.3. As the number of product types increases to two, workstations require set-up whenever there is a product change. Therefore, the percentage of time spent on set-up increases dramatically from zero to a particular number. Table 5.3 also reveals that the percentage of time spent on set-up only gently increases when the types of products increases from two to three, three to four, four to five, five to six, six to seven and seven to eight. Take the phosphate workstation as an example. When the number of product types changes from one to two, the percentage of time spent on set-up increases 15.5-0=15.5. When the number of product types changes from two to three, the percentage of time spent on set-up increases 19.9-15.5=4.4. Following this routine provides the numbers 1.9, 1.2, 0.9, 0.1, and 0.8. As 15.5 is much bigger than 1.9, 1.2, 0.9, 0.1 and 0.8, the throughput dramatically reduces from 7.07 to 2.67 when the number of product types changes from one to two, and drops down slightly as the number of product types changes further.

	Percentage of time spent on set-up							
The number of product types	1	2	3	4	5	6	7	8
Phosphate	0	15.5	19.9	21.8	23	23.9	24	24.8
Dry up	0	17.7	22.5	24.8	26	27.1	27.8	28
Base-coat	0	26.6	33.9	37.2	39.2	40.5	41.4	42.1
Bake oven 1and2	0	26.6	33.9	37.2	39.2	40.5	41.5	42.1
Cooling 1and2	0	30.9	39.5	43.2	45.8	47.5	48.2	49.3
Sanding	0	35.5	45.1	49.3	52.2	54.2	55.2	56.3
Top-coat	0	35.5	45.1	49.3	52.2	54.2	55.2	56.3
Inspection and repair	0	26.6	34	37.2	39.2	40.5	41.4	42.1

Table 5.3: Workstations' percentage of time spent on set-up when the number of product types increases

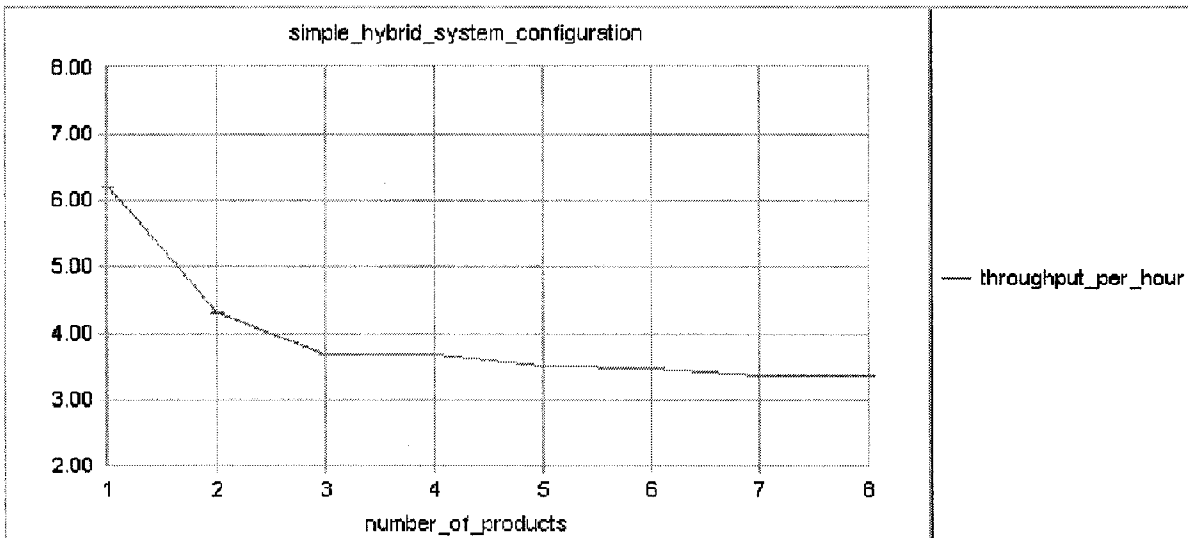
- **Input data for the simple hybrid system configuration**

The input data are the same as the data in figure 5.1. The product mix-ratios are the same for all kinds of product.

- **Simulation results and analysis of the simple hybrid system configuration**

Figure 5.3 is drawn based on the simple hybrid system configuration. For experimental data details, refer to Appendix E. The curved shape in Figure 5.3 is similar to the curved shape in Figure 5.2—with the throughput decreasing as the product type increases.

One might ask why the throughput drops down dramatically for the simple hybrid system configuration when the number of product types changes from one to two? We know that no set-up is needed when two types of products are produced by the simple hybrid system configuration.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
number of product types	1	2	3	4	5	6	7	8	0	0	0	0	0	0	0	0	0	0	0
— throughput_per_hour	6.20	4.31	3.68	3.69	3.50	3.50	3.38	3.38											

Figure 5.3: The effect of varying the number of product types on throughput in simple hybrid system configuration

Figure 5.4 presents an answer to this question, with the rectangles representing workstations and the circles signifying loads. When there is only one type of product, the products are directed to the upper and lower serial lines one-by-one. When there are two types of products, the first type of product is conveyed to the upper serial line and the second type of product to the lower serial line to save set-up time, i.e. grouping products. In addition, the loads arrive randomly, so the same kind of product is sometimes grouped together, leading to the increasing probability that either the upper or lower serial lines will become blocked when the number of product types changes from one to two. Therefore, the throughput decreases from 6.20 to 4.31 per hour, even though no set-up time is needed for either situation.

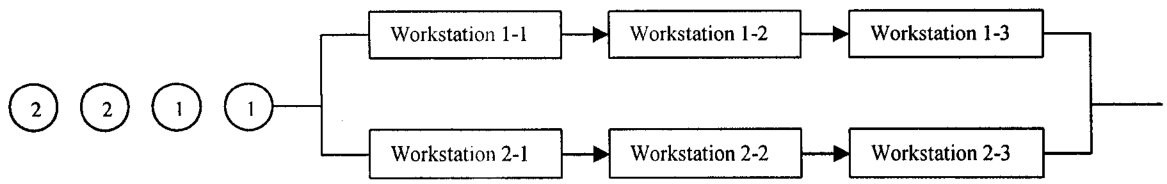


Figure 5.4: Throughput illustration

The above analysis is also verified by the simulation data in Table 5.4, which shows the probability of the product being blocked in the dry-up workstation (before Workstations 1-1 and 2-1) when the processing is over. As the number of product types changes from one to two, the probability of the product becoming blocked increases from 0.39 to 0.55, leading to a decrease in throughput.

The number of product types	1	2
Probability of product being blocked	0.39	0.55
Standard deviation	0.03	0.02
Minimum	0.34	0.53
Maximum	0.43	0.60
Median	0.39	0.54
Number of runs	10	10

Table 5.4: Probability of product becoming blocked

- **Comparison of the serial and the simple hybrid system configurations**

Figures 5.2 and 5.3 tell us that the throughput of the serial system configuration for one type of product is 7.07 per hour, while the throughput of the simple hybrid system configuration for the same product is 6.20 per hour. Therefore, the serial system configuration performs better than the simple hybrid system configuration for one type of product. The simple hybrid system configuration needs a complex material handling system (cross-line transportation cars), which leads to spending more time on work-in-process (WIP) than the serial system configuration. The simulation data from Table 5.5 also shows that the product spends more time on WIP in the simple hybrid system configuration than in the serial system configuration (43.8>38.7).

	The simple hybrid system configuration	The serial system configuration
Average	43.8 minutes	38.7 minutes
Standard deviation	0.2 minutes	0.1 minutes
Minimum	43.6 minutes	38.5 minutes
Maximum	44.1 minutes	38.9 minutes
Median	43.9 minutes	38.8 minutes
Number of runs	10	10

Table 5.5: WIP of the simple hybrid and the serial system configurations

As the product type increases from one to two, however, a significant difference occurs between the two kinds of system configurations. The throughput of the simple hybrid system configuration is around 4.31 per hour—much higher than the throughput of the serial system configuration, which is approximately 2.67 per hour. When the number of product types increases further, the modeller finds that the curve of the simple hybrid system configuration is always above the curve of the serial system configuration.

Analyzing both types of system configurations allows one to choose a suitable strategy for deploying facilities in terms of throughput. In the paint shop, different products are often produced together, and comparing the two sets of data reveals to the designer that the simple hybrid system configuration is preferred over the serial system configuration because the number of product types often exceeds one.

5.3 Determining optimal number of carriers

Carriers (a part of the material handling system introduced later) used in the truck paint shop are also conveyed to some workstations located in the body shop and the trim and chassis shop. Here, carriers are used to connect different material handling systems in the body shop, the paint shop and the trim and chassis shop.

Graehl (1992) stated that simulation studies of large material handling systems include only a portion of the entire system. One reason for this is that large systems tend to require equally large and time-consuming simulation efforts. As a result, the modeller defines the scope of study to include everything from the white body storage to the painted body storage, and assumes that the carriers are then freed and returned to the carrier storage once the load arrives at the painted body storage. Another assumption is that one kind of carrier is designed to fit all kinds of products. In this way, the designer can gain insight into how many carriers the truck paint shop needs.

It is important to determine what is a suitable number of carriers. On the one hand, if the number of carriers is not enough it becomes a bottleneck in the whole manufacturing system—leading to reduced throughput in the truck paint shop. On the other hand, because the carriers are bulky and require a lot of space for storage, it is impossible to put many carriers in the carrier storage—a waste of money and space on the shop floor.

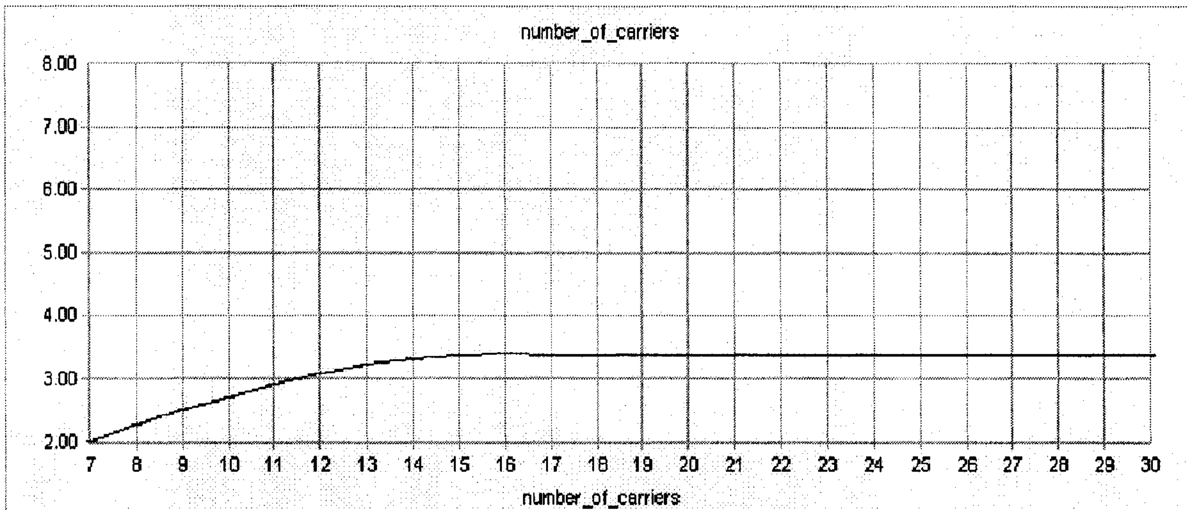
- **Introduction of Experiment III**

Carriers are used to transport truck bodies from one workstation in the paint shop to another. Each carrier in this paint shop is able to carry one truck body at a time. The carriers wait in storage until a white truck body arrives at the white body storage. In the loading zone, the white truck body is put on top of one carrier, which then goes through all the necessary processes with the truck body. When said processes are finished, the carrier separates from the truck body and is conveyed to carrier storage.

Similarly, at this level in the simulation model, the designer excludes optimal buffer location, shift-changing and maintenance policies as factors with the potential to affect the throughput in the truck paint shop. The simulation model of the simple hybrid system is furthered by considering the number of carriers as a factor that affects the throughput—based on the results from Experiment II. Eighty kinds of product are produced in this case—prompting the input data to be the same as that in Table 5.1.

- **Simulation results and analysis of Experiment III**

Figure 5.5 is obtained by running the furthered model. For more information on the experimental data, refer to Appendix F. The X-axis in Figure 5.5 represents the number of carriers, while the Y-axis indicates the throughput per hour. The modeller increases the number of carriers from seven to thirty.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
the number of carriers	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
throughput_per_hour	2.03	2.28	2.51	2.72	2.93	3.09	3.23	3.32	3.36	3.39	3.38	3.38	3.38	3.38	3.38	3.38	3.38	3.38	3.38	3.38	3.38	3.38	3.38	3.38

Figure 5.5: The effect of varying the number of carriers on throughput in simple hybrid system configuration

Figure 5.5 reveals how the throughput dramatically increases with the number of carriers at the beginning, because there are not enough number of carriers in the whole manufacturing system. Therefore, a small change in the quantity of carriers prompts a large difference in terms of throughput. When the number reaches forteen carriers, the throughput gently rises with the number of carriers until it hits seventeen—at which point the simulation model enters a stable state and the throughput stops increasing with the number of carriers. It comes down to a trade-off for the designer. If the efficient utilization of the shop floor is more important than the little difference in terms of throughput when the number of carriers is forteen to seventeen, then forteen carriers is preferable to seventeen carriers. Otherwise, seventeen carriers is preferable. Finally, the designer decides to put seventeen carriers in the carrier storage.

In addition, the modeller notices that the throughput at seventeen carriers in Figure 5.5 is 3.38— the same as 3.38, the throughput in Figure 5.3 for eight types of product. The experimental data in Figure 5.3 is obtained by assuming that there are infinite carriers available to fix the truck body, therefore the carriers cannot be the bottleneck of the whole manufacturing system. Furthermore, when the quantity of carriers reaches seventeen, they no longer act as the system bottleneck anymore—according to Figure 5.5. Consequently, the two throughputs are the same.

5.4 Determining optimal buffer location

This section is designed to validate Equation 2.5, which states that the workstation with a slower parameter draws the buffer towards it (where the parameter is expressed by Equation 2.5). The obtained result is applied in our case study to predict the optimal buffer location in the simple hybrid system configuration.

- **Introduction of Experiment IV**

Experiment IV considers cycle times, mean time between failures, mean time to repair and the product's mix-ratio as potential factors to define a buffer location. A simulation model consisting of eight workstations and two buffers is set up to obtain analysis data. Four kinds of products are produced in this serial production line.

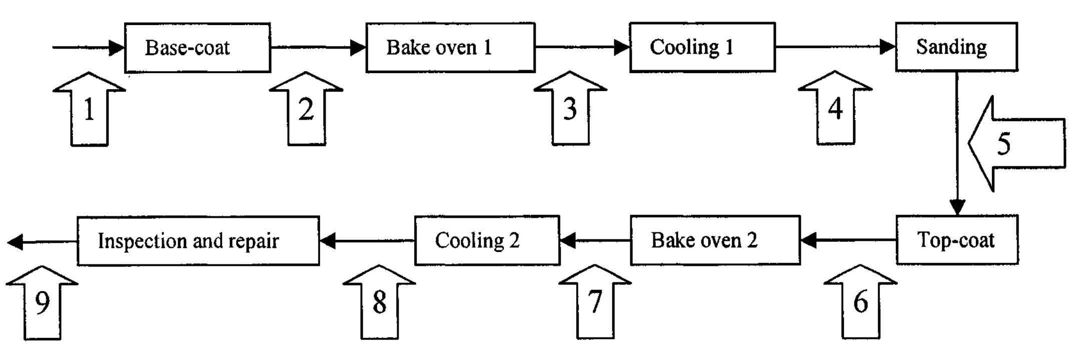


Figure 5.6: Buffer location

Figure 5.6 illustrates the configuration, with the rectangles representing workstations and the arrows indicating the candidate buffer locations. In our case, two identical buffers are allowed in the production line. Therefore, there are thirty-six possible combinations—as illustrated in Table 5.7.

	Possible combinations							
Buffer I at location 1	(1,2)	(1,3)	(1,4)	(1,5)	(1,6)	(1,7)	(1,8)	(1,9)
Buffer I at location 2	(2,3)	(2,4)	(2,5)	(2,6)	(2,7)	(2,8)	(2,9)	
Buffer I at location 3	(3,4)	(3,5)	(3,6)	(3,7)	(3,8)	(3,9)		
Buffer I at location 4	(4,5)	(4,6)	(4,7)	(4,8)	(4,9)			
Buffer I at location 5	(5,6)	(5,7)	(5,8)	(5,9)				
Buffer I at location 6	(6,7)	(6,8)	(6,9)					
Buffer I at location 7	(7,8)	(7,9)						
Buffer I at location 8	(8,9)							

Table 5.6: Possible combination of two buffers in seven locations

- **Input data of Experiment IV**

There are two sets of input data. The first set is about the workstations and is shown as follows: Time between failures for base-coat and top-coat booths is assumed to be uniform distribution with a mean of 40 minutes, and a standard deviation of 5 minutes. Time between failures for other facilities is assumed to be uniform distribution with a mean of 8 hours, and a standard deviation of 1 hour. Time to repair is assumed to be the same for all facilities—uniform distribution with a mean of 12 minutes and a standard deviation of 2 minutes.

	Product's mix-ratio	Cycle times in minutes					
		Base-coat	Bake oven 1 & 2	Cooling 1 & 2	Sanding	Top-coat	Inspection and repair
Product 1	30%	10.4, 3	10.2, 2	10.2, 1	10.0, 1	10.3, 3	10.1, 1
Product 2	30%	10.6, 3	10.4, 2	10.4, 1	10.2, 1	10.5, 3	10.3, 1
Product 3	20%	13, 3	12.8, 2	12.7, 1	12.8, 1	13.1, 3	13, 1
Product 4	20%	13.2, 3	13, 2	12.9, 1	12.8, 1	13.4, 3	13.2, 1

Table 5.7: Product's mix-ration and cycle times

The second set of data is about a product's mix-ratio and the cycle times of different products in different workstations. Table 5.8 specifies the detailed information. All the cycle times are uniform distribution. The first value is the mean, while the second value is the standard deviation.

- **Simulation results and analysis of Experiment IV**

Based on the above input data, Figure 5.7 is obtained by running the simulation model. Appendix G illustrates the detailed experimental data. The X-axis represents the throughput per hour, and the Y-axis indicates the different buffer-combinations. For example, (5, 6) means: one buffer is put at Location 5, and the other at Location 6. Figure 5.7 shows the modeller that the buffer combination of (2, 5) is better than all the other buffer combinations.

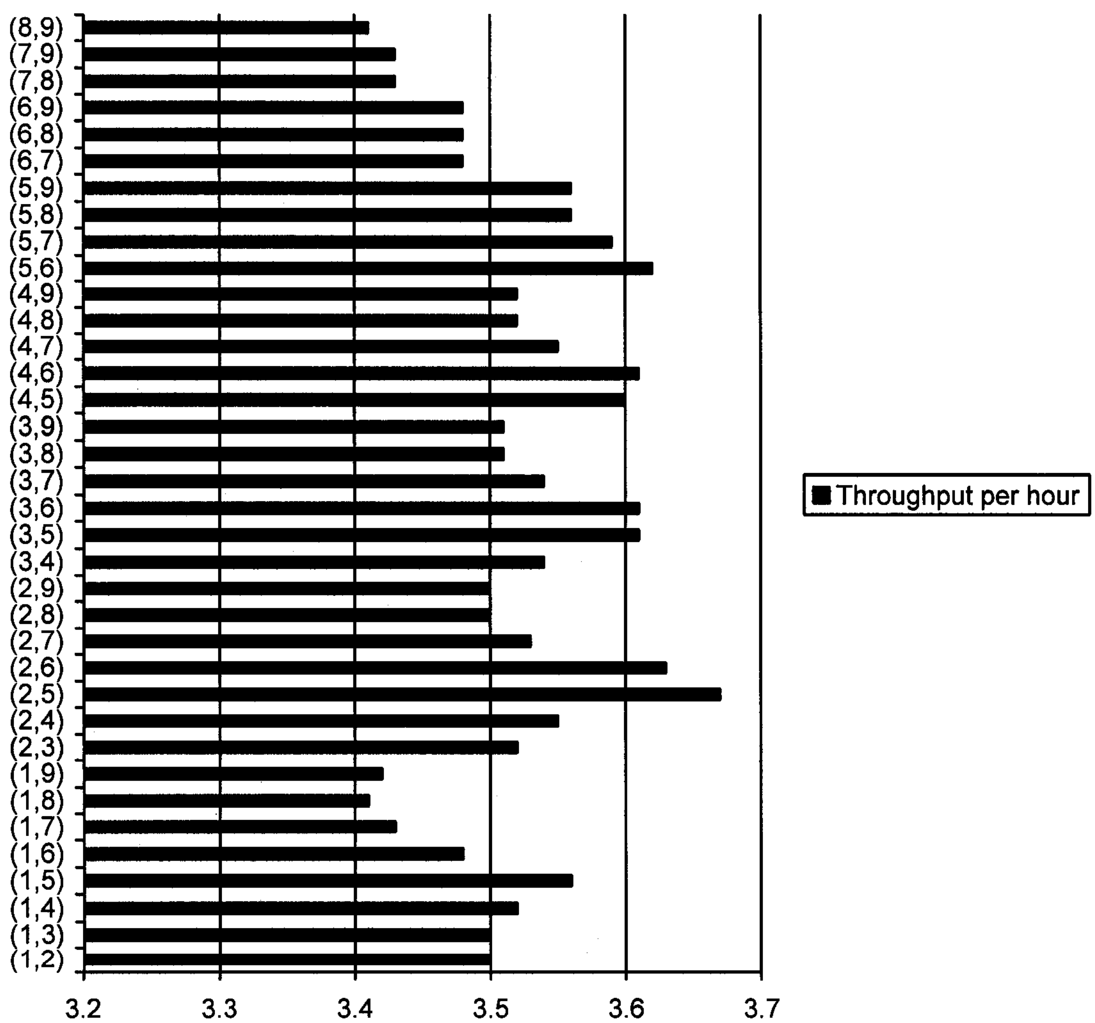


Figure 5.7: Throughput of different buffer-combinations

In addition, the stand-alone speeds of different workstations are calculated according to Equation 2.5:

$$S_1 = \frac{1}{n_1t_1 + n_2t_2 + \cdots n_nt_n} \times \frac{MTBF}{MTBF+MTTR} = \frac{100}{30\times10.4+30\times10.6+20\times13+20\times13.2} \times \frac{0.67}{0.67+0.2} = 0.0667$$

$$S_2 = \frac{1}{n_1t_1 + n_2t_2 + \cdots n_nt_n} \times \frac{MTBF}{MTBF+MTTR} = \frac{100}{30\times10.2+30\times10.4+20\times12.8+20\times13} \times \frac{8}{8+0.2} = 0.0860$$

$$S_3 = \frac{1}{n_1t_1 + n_2t_2 + \cdots n_nt_n} \times \frac{MTBF}{MTBF+MTTR} = \frac{100}{30\times10.2+30\times10.4+20\times12.7+20\times12.9} \times \frac{8}{8+0.2} = 0.0863$$

$$S_4 = \frac{1}{n_1t_1 + n_2t_2 + \cdots n_nt_n} \times \frac{MTBF}{MTBF+MTTR} = \frac{100}{30\times10.0+30\times10.2+20\times12.8+20\times12.8} \times \frac{8}{8+0.2} = 0.0873$$

$$S_5 = \frac{1}{n_1t_1 + n_2t_2 + \cdots n_nt_n} \times \frac{MTBF}{MTBF+MTTR} = \frac{100}{30\times10.3+30\times10.5+20\times13.1+20\times13.4} \times \frac{0.67}{0.67+0.2} = 0.0667$$

$$S_6 = \frac{1}{n_1t_1 + n_2t_2 + \cdots n_nt_n} \times \frac{MTBF}{MTBF+MTTR} = \frac{100}{30\times10.1+30\times10.3+20\times13+20\times13.2} \times \frac{8}{8+0.2} = 0.0859$$

Where,

a) S_1 is the stand-alone speed of base-coat
b) S_2 is the stand-alone speed of bake oven 1 and 2
c) S_3 is the stand-alone speed of cooling 1 and 2
d) S_4 is the stand-alone speed of sanding
e) S_5 is the stand-alone speed of top-coat
f) S_6 is the stand-alone speed of inspection and repair

These calculations reveal that the stand-alone speeds of base-coat and top-coat are slower than the other workstations. Therefore, two buffers should be drawn to the base-coat and top-coat workstations—based on our prediction. This is validated by Figure 5.7, where the buffer-combination of Locations 2 and 5 appears to yield a larger throughput. Candidate

Location 2 is beside the base-coat workstation, and candidate Location 5 is beside the top-coat workstation.

5.5 Corrective maintenance versus preventive maintenance

The simulation model of the simple hybrid system configuration is advanced again to select a suitable maintenance policy for operating the factory on the shop floor. Corrective and preventive maintenance policies are tested by eight kinds of products in the experiment to distinguish the throughput differences.

There are two assumptions made about maintenance polices. First, that spare parts and maintenance crew are always available whenever failure occurs. Second, that the maintenance crew keeps to the regular preventive maintenance schedule even though an accident failure occurs between two preventive maintenance periods.

- **Input data for Experiment V**

There are three sets of input data in Experiment V. The first set addresses the processing time. Since eight kinds of products are used to test maintenance polices, the first set of data is the same as in Table 5.1.

The second set of data is about the corrective maintenance policy, which is illustrated as follows:

a) Time between failures for base-coat and top-coat booths follows an uniform distribution with a mean of 40 minutes, and a standard deviation of 5 minutes.
b) Time between failures for other facilities follows an uniform distribution with a mean of 8 hours, and a standard deviation of 1 hour.
c) Time to repair is assumed to be the same for all facilities and follows an uniform distribution with a mean of 12 minutes, and a standard deviation of 2 minutes.

The last set of data is about the preventive maintenance policy, which states that all workstations are to be repaired for 8 hours every 40 hours. In addition, failures occur as the following statement between two preventive maintenance shifts.

a) Time between failures for base-coat and top-coat booths follows an uniform distribution with a mean of 400 minutes, and a standard deviation of 50 minutes.
b) Time between failures for other facilities follows an uniform distribution with a mean of 80 hours and a standard deviation of 10 hour.
c) Time to repair is assumed to be the same for all facilities and follows an uniform distribution with a mean of 12 minutes, and a standard deviation of 2 minutes

- **Simulation results and analysis of Experiment V**

Table 5.9 provides the results of the corrective and preventive maintenance tests.

	Corrective maintenance	Preventive maintenance
Average (throughput per hour)	2.90	2.80
Standard deviation	0.01	0.10
Minimum	2.88	2.7
Maximum	2.93	2.95
Median	2.90	3.81
Number of runs	10	10

Table 5.8: Corrective maintenance versus preventive maintenance

The two test data categories show the modeller that the standard deviation of preventive maintenance is much higher than that of corrective maintenance. It is reasonable, because there are eight hours of maintenance every forty hours for preventive maintenance. The manufacturing system stops completely for PM, but not for CM—which leads PM to have higher variation in terms of throughput than CM. Either way, the corrective maintenance policy performs better than the preventive maintenance policy. Its throughput is 2.90—smaller than 3.38 (shown Figure 5.3 at eight kinds of products)—due to the inclusion of random failures and repairs as factors that influence the throughput.

5.6 Final throughput

The next step is to give the factory manager a clear picture of how many products the truck paint shop can actually produce every month when all the selected factors are included. Based on this data, the manager could efficiently control the manufacturing system and meet the market requirement. To estimate the throughput per month, the end-shift change must be considered in the simulation model because it causes the manufacturing system to stop every day.

- **Input data for Experiment VI**

In addition to the input data in Table 5.1, the simulation model utilizes the above experimental results to maximize the final throughput per month when the number of product types varies from one to eight. Therefore, a number of seventeen carriers, the buffer combination of (2, 5) and the use of a corrective maintenance police are the parameters in Experiment VI.

Furthermore, the manufacturing system carries out a one-shift and five-days-a-week operation policy. So the factory operates twenty-two days or 176 hours per month.

- **Simulation results and analysis of Experiment VI**

Figure 5.8 follows. For the detailed experimental data, refer to Appendix H.

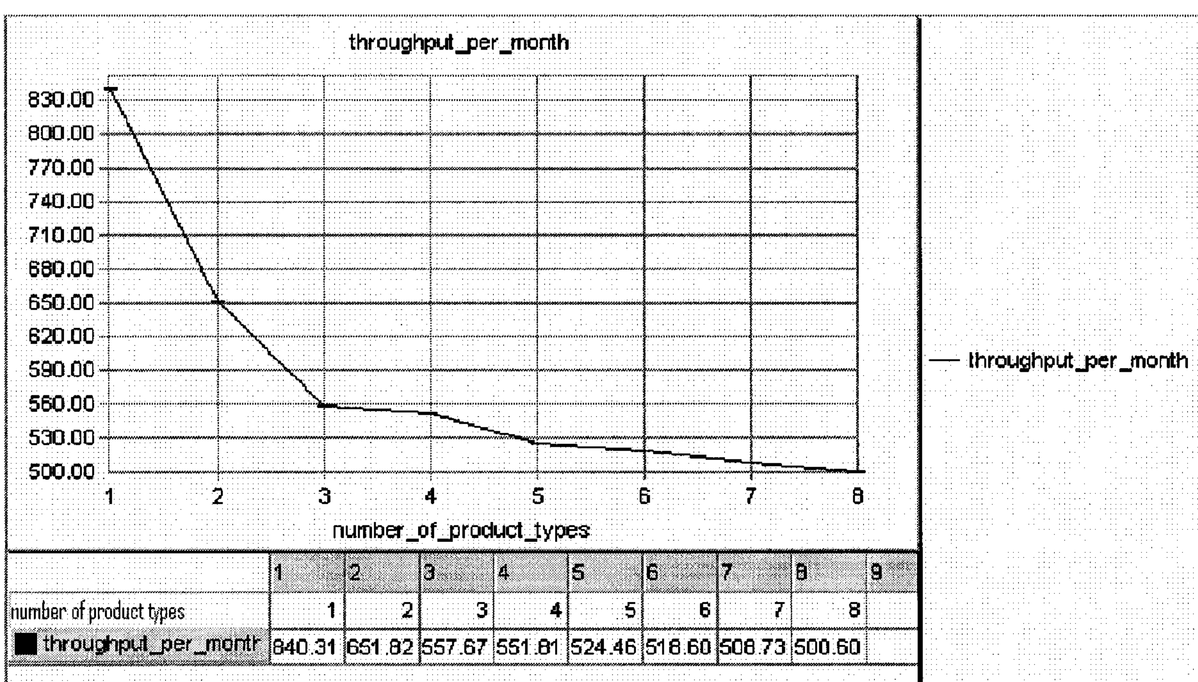


Figure 5.8: Final throughput per month

Experiment V told the modeller that the throughput is 2.90 per hour when there are eight kinds of products and the selected factors are considered. Figure 5.8 is verified with a simple calculation made in terms of average throughput per month:

$$2.90 \times 8 \times 22 = 510 \quad \text{units/month}$$

When the end-shift is included as a factor, the manufacturing system is limited by another restriction. Therefore, the final average throughput per month reduces from 510 to 500 as figure 5.8 shows (with eight kinds of products).

CHAPTER VI: CONCLUSION AND FUTURE WORK

6.1 Conclusion

This paper covers common issues found in the truck paint shop, and discusses system configurations, suitable number of carriers, optimal buffer locations and maintenance policies—all as ways of improving the throughput. The final throughput is also presented, with all the optimal factors considered.

This study is the first to thoroughly compare the two kinds of system configurations in terms of throughput. The research results of Powell and Pyke (1996) are, in turn, improved by replacing a larger mean processing time with a slower stand-alone speed. In Experiment 5.4, the simulation model verifies this improvement.

Manufacturing systems are a diverse as people, however, and have a tendency to display different personalities. Therefore, each particular manufacturing system needs to be analyzed using particular factors. This paper focused on the methodologies used to analyze the manufacturing systems, rather than on the obtained data. Simulation as a modelling technique provides the designer with a powerful tool for solving various problems.

6.2 Future work

Another significant problem in the paint shop is known as a "Single-track multi-hoist scheduling problem", which can also happen in the cross-line transportation car of our manufacturing system—particularly when there are multi-cars and many workstations to serve. Second, a conceptual comparison of the different system configurations—as illustrated in Figure 2.2—would also be required in the future. Finally, the experiments only select a factor at a time. In the future, the experiment should be designed to test all the factors at the same time and optimize the final throughput.

APPEXDIX A: CODE OF SERIAL SYSTEM CONFIGURATION

```
/*read data from input file*/
begin p_init arriving procedure

  read v_junk, v_partarrivingtime,v_junk from "arc/input.txt"
  print "part_arriving_time=" v_partarrivingtime current value to
lbl_partarrivingtime

  read v_junk, v_partdieingtime,v_junk   from "arc/input.txt"
  print "part_die_time=" v_partdieingtime current value to
lbl_partdieingtime

  read v_junk, v_minordefectrate,v_junk   from "arc/input.txt"
  print "minor_defect_rate=" v_minordefectrate current value "%" to
lbl_minordefectrate

  read v_junk, v_majordefectrate,v_junk   from "arc/input.txt"
  print "major_defect_rate=" v_majordefectrate current value "%" to
lbl_majordefectrate

  read v_junk, v1_mixrate,v_junk   from "arc/input.txt"
  print "product1_portion=" v1_mixrate current value "%" to lbl1_mixrate

  read v_junk, v2_mixrate,v_junk   from "arc/input.txt"
  print "product2_portion=" v2_mixrate current value "%" to lbl2_mixrate

  read v_junk, v3_mixrate,v_junk   from "arc/input.txt"
  print "product3_portion=" v3_mixrate current value "%" to lbl3_mixrate

  read v_junk, v4_mixrate,v_junk   from "arc/input.txt"
  print "product4_portion=" v4_mixrate current value "%" to lbl4_mixrate

  read v_junk, v5_mixrate,v_junk   from "arc/input.txt"
  print "product5_portion=" v5_mixrate current value "%" to lbl5_mixrate

  read v_junk, v6_mixrate,v_junk   from "arc/input.txt"
  print "product6_portion=" v6_mixrate current value "%" to lbl6_mixrate

  read v_junk, v7_mixrate,v_junk   from "arc/input.txt"
  print "product7_portion=" v7_mixrate current value "%" to lbl7_mixrate

  read v_junk, v8_mixrate,v_junk   from "arc/input.txt"
  print "product8_portion=" v8_mixrate current value "%" to lbl8_mixrate

  read v_junk, v_phosphatesetup,v_junk   from "arc/input.txt"
  read v_junk, v_dryupsetup,v_junk   from "arc/input.txt"
  read v_junk, v_basecoatsetup,v_junk   from "arc/input.txt"
  read v_junk, v_bakeovensetup,v_junk   from "arc/input.txt"
  read v_junk, v_coolingsetup,v_junk   from "arc/input.txt"
  read v_junk, v_sandingsetup,v_junk   from "arc/input.txt"
  read v_junk, v_topcoatsetup,v_junk   from "arc/input.txt"
  read v_junk, v_inspectionsetup,v_junk   from "arc/input.txt"

  read v_junk, v_timebetweenmaintenance,v_junk   from "arc/input.txt"
```

```
  print "time_between_maintenances=" v_timebetweenmaintenance current
value "min" to lbl_timebetweenmaintenance

  read v_junk, v_maintenancetime,v_junk   from "arc/input.txt"
  print "maintenancetime=" v_maintenancetime current value "min" to
lbl_maintenancetime

  read v_junk, v1_cycletimeofphosphate,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeofdry,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeofbasecoat,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeofbaseoven,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeofcooling,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeofsanding,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeoftopcoat,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeofinspection,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeofunload,v_junk   from "arc/input.txt"

  read v_junk, v2_cycletimeofphosphate,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeofdry,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeofbasecoat,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeofbaseoven,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeofcooling,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeofsanding,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeoftopcoat,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeofinspection,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeofunload,v_junk   from "arc/input.txt"

  read v_junk, v3_cycletimeofphosphate,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeofdry,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeofbasecoat,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeofbaseoven,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeofcooling,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeofsanding,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeoftopcoat,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeofinspection,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeofunload,v_junk   from "arc/input.txt"

  read v_junk, v4_cycletimeofphosphate,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeofdry,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeofbasecoat,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeofbaseoven,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeofcooling,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeofsanding,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeoftopcoat,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeofinspection,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeofunload,v_junk   from "arc/input.txt"

  read v_junk, v5_cycletimeofphosphate,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeofdry,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeofbasecoat,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeofbaseoven,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeofcooling,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeofsanding,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeoftopcoat,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeofinspection,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeofunload,v_junk   from "arc/input.txt"
```

```
read v_junk, v6_cycletimeofphosphate,v_junk   from "arc/input.txt"
read v_junk, v6_cycletimeofdry,v_junk   from "arc/input.txt"
read v_junk, v6_cycletimeofbasecoat,v_junk   from "arc/input.txt"
read v_junk, v6_cycletimeofbaseoven,v_junk   from "arc/input.txt"
read v_junk, v6_cycletimeofcooling,v_junk   from "arc/input.txt"
read v_junk, v6_cycletimeofsanding,v_junk   from "arc/input.txt"
read v_junk, v6_cycletimeoftopcoat,v_junk   from "arc/input.txt"
read v_junk, v6_cycletimeofinspection,v_junk   from "arc/input.txt"
read v_junk, v6_cycletimeofunload,v_junk   from "arc/input.txt"

read v_junk, v7_cycletimeofphosphate,v_junk   from "arc/input.txt"
read v_junk, v7_cycletimeofdry,v_junk   from "arc/input.txt"
read v_junk, v7_cycletimeofbasecoat,v_junk   from "arc/input.txt"
read v_junk, v7_cycletimeofbaseoven,v_junk   from "arc/input.txt"
read v_junk, v7_cycletimeofcooling,v_junk   from "arc/input.txt"
read v_junk, v7_cycletimeofsanding,v_junk   from "arc/input.txt"
read v_junk, v7_cycletimeoftopcoat,v_junk   from "arc/input.txt"
read v_junk, v7_cycletimeofinspection,v_junk   from "arc/input.txt"
read v_junk, v7_cycletimeofunload,v_junk   from "arc/input.txt"

read v_junk, v8_cycletimeofphosphate,v_junk   from "arc/input.txt"
read v_junk, v8_cycletimeofdry,v_junk   from "arc/input.txt"
read v_junk, v8_cycletimeofbasecoat,v_junk   from "arc/input.txt"
read v_junk, v8_cycletimeofbaseoven,v_junk   from "arc/input.txt"
read v_junk, v8_cycletimeofcooling,v_junk   from "arc/input.txt"
read v_junk, v8_cycletimeofsanding,v_junk   from "arc/input.txt"
read v_junk, v8_cycletimeoftopcoat,v_junk   from "arc/input.txt"
read v_junk, v8_cycletimeofinspection,v_junk   from "arc/input.txt"
read v_junk, v8_cycletimeofunload,v_junk   from "arc/input.txt"

/*print v8_cycletimeofunload current value to lbl_test*/

send to p_loadcreating

end


/*load creating*/
begin p_loadcreating arriving procedure

if v_numberofproducts = 7 then
   begin
   set v8_mixrate = 0
end

else if v_numberofproducts = 6 then
   begin
   set v8_mixrate = 0
   set v7_mixrate = 0
end

else if v_numberofproducts = 5 then
   begin
   set v8_mixrate = 0
   set v7_mixrate = 0
   set v6_mixrate = 0
end
```

```
else if v_numberofproducts = 4 then
   begin
   set v8_mixrate = 0
   set v7_mixrate = 0
   set v6_mixrate = 0
   set v5_mixrate = 0
end

else if v_numberofproducts = 3 then
   begin
   set v8_mixrate = 0
   set v7_mixrate = 0
   set v6_mixrate = 0
   set v5_mixrate = 0
   set v4_mixrate = 0
end

else if v_numberofproducts = 2 then
   begin
   set v8_mixrate = 0
   set v7_mixrate = 0
   set v6_mixrate = 0
   set v5_mixrate = 0
   set v4_mixrate = 0
   set v3_mixrate = 0
end

else if v_numberofproducts = 1 then
   begin
   set v8_mixrate = 0
   set v7_mixrate = 0
   set v6_mixrate = 0
   set v5_mixrate = 0
   set v4_mixrate = 0
   set v3_mixrate = 0
   set v2_mixrate = 0
end

while 1=1 do begin

  /*creating mixed products*/
  set a_product to oneof(v1_mixrate:1,v2_mixrate:2,v3_mixrate:3,
  v4_mixrate:4,v5_mixrate:5,v6_mixrate:6,v7_mixrate:7,v8_mixrate:8)

  /*set up processing parameters according to product type*/
  if a_product = 1 then
     begin
     set a_cycletimeofphosphate to v1_cycletimeofphosphate
     set a_cycletimeofdry to v1_cycletimeofdry
     set a_cycletimeofbasecoat to v1_cycletimeofbasecoat
     set a_cycletimeofbakeoven to v1_cycletimeofbaseoven
     set a_cycletimeofcooling to v1_cycletimeofcooling
     set a_cycletimeofsanding to v1_cycletimeofsanding
     set a_cycletimeoftopcoat to v1_cycletimeoftopcoat
     set a_cycletimeofinspection to v1_cycletimeofinspection
     set a_cycletimeofunload to v1_cycletimeofunload
```

```
   /*print a_cycletimeofunload current value to lbl_test*/
   clone 1 load to p_wbs nlt l_product1
end

else if a_product = 2 then
   begin
   set a_cycletimeofphosphate to v2_cycletimeofphosphate
   set a_cycletimeofdry to v2_cycletimeofdry
   set a_cycletimeofbasecoat to v2_cycletimeofbasecoat
   set a_cycletimeofbakeoven to v2_cycletimeofbaseoven
   set a_cycletimeofcooling to v2_cycletimeofcooling
   set a_cycletimeofsanding to v2_cycletimeofsanding
   set a_cycletimeoftopcoat to v2_cycletimeoftopcoat
   set a_cycletimeofinspection to v2_cycletimeofinspection
   set a_cycletimeofunload to v2_cycletimeofunload
  /* print a_cycletimeofunload current value to lbl_test*/
   clone 1 load to p_wbs nlt l_product2
end

else if a_product = 3 then
   begin
   set a_cycletimeofphosphate to v3_cycletimeofphosphate
   set a_cycletimeofdry to v3_cycletimeofdry
   set a_cycletimeofbasecoat to v3_cycletimeofbasecoat
   set a_cycletimeofbakeoven to v3_cycletimeofbaseoven
   set a_cycletimeofcooling to v3_cycletimeofcooling
   set a_cycletimeofsanding to v3_cycletimeofsanding
   set a_cycletimeoftopcoat to v3_cycletimeoftopcoat
   set a_cycletimeofinspection to v3_cycletimeofinspection
   set a_cycletimeofunload to v3_cycletimeofunload
   /*print a_cycletimeofunload current value to lbl_test*/
   clone 1 load to p_wbs nlt l_product3
end

else if a_product = 4 then
   begin
   set a_cycletimeofphosphate to v4_cycletimeofphosphate
   set a_cycletimeofdry to v4_cycletimeofdry
   set a_cycletimeofbasecoat to v4_cycletimeofbasecoat
   set a_cycletimeofbakeoven to v4_cycletimeofbaseoven
   set a_cycletimeofcooling to v4_cycletimeofcooling
   set a_cycletimeofsanding to v4_cycletimeofsanding
   set a_cycletimeoftopcoat to v4_cycletimeoftopcoat
   set a_cycletimeofinspection to v4_cycletimeofinspection
   set a_cycletimeofunload to v4_cycletimeofunload
   /*print a_cycletimeofunload current value to lbl_test*/
   clone 1 load to p_wbs nlt l_product4
end

else if a_product = 5 then
   begin
   set a_cycletimeofphosphate to v5_cycletimeofphosphate
   set a_cycletimeofdry to v5_cycletimeofdry
   set a_cycletimeofbasecoat to v5_cycletimeofbasecoat
   set a_cycletimeofbakeoven to v5_cycletimeofbaseoven
   set a_cycletimeofcooling to v5_cycletimeofcooling
   set a_cycletimeofsanding to v5_cycletimeofsanding
```

```
      set a_cycletimeoftopcoat to v5_cycletimeoftopcoat
      set a_cycletimeofinspection to v5_cycletimeofinspection
      set a_cycletimeofunload to v5_cycletimeofunload
      /*print a_cycletimeofunload current value to lbl_test*/
      clone 1 load to p_wbs nlt l_product5
   end

   else if a_product = 6 then
      begin
      set a_cycletimeofphosphate to v6_cycletimeofphosphate
      set a_cycletimeofdry to v6_cycletimeofdry
      set a_cycletimeofbasecoat to v6_cycletimeofbasecoat
      set a_cycletimeofbakeoven to v6_cycletimeofbaseoven
      set a_cycletimeofcooling to v6_cycletimeofcooling
      set a_cycletimeofsanding to v6_cycletimeofsanding
      set a_cycletimeoftopcoat to v6_cycletimeoftopcoat
      set a_cycletimeofinspection to v6_cycletimeofinspection
      set a_cycletimeofunload to v6_cycletimeofunload
      /*print a_cycletimeofunload current value to lbl_test*/
      clone 1 load to p_wbs nlt l_product6
   end

   else if a_product = 7 then
      begin
      set a_cycletimeofphosphate to v7_cycletimeofphosphate
      set a_cycletimeofdry to v7_cycletimeofdry
      set a_cycletimeofbasecoat to v7_cycletimeofbasecoat
      set a_cycletimeofbakeoven to v7_cycletimeofbaseoven
      set a_cycletimeofcooling to v7_cycletimeofcooling
      set a_cycletimeofsanding to v7_cycletimeofsanding
      set a_cycletimeoftopcoat to v7_cycletimeoftopcoat
      set a_cycletimeofinspection to v7_cycletimeofinspection
      set a_cycletimeofunload to v7_cycletimeofunload
      /*print a_cycletimeofunload current value to lbl_test*/
      clone 1 load to p_wbs nlt l_product7
   end

   else if a_product = 8 then
      begin
      set a_cycletimeofphosphate to v8_cycletimeofphosphate
      set a_cycletimeofdry to v8_cycletimeofdry
      set a_cycletimeofbasecoat to v8_cycletimeofbasecoat
      set a_cycletimeofbakeoven to v8_cycletimeofbaseoven
      set a_cycletimeofcooling to v8_cycletimeofcooling
      set a_cycletimeofsanding to v8_cycletimeofsanding
      set a_cycletimeoftopcoat to v8_cycletimeoftopcoat
      set a_cycletimeofinspection to v8_cycletimeofinspection
      set a_cycletimeofunload to v8_cycletimeofunload
      clone 1 load to p_wbs nlt l_product8
   end

   wait for e v_partarrivingtime min
   /*time between load arrivals*/

  end

end
```

```
begin p_wbs arriving procedure

 move into q_wbs /*check the average number in wbs*/
 send to p_loading

end


begin p_loading arriving procedure

  move into conv:sta_load
  set a_timestamp to ac
  use r_load for u 3, 1 min
  /*fix white truck body to the carrier*/
  send to p_phosphate

end


begin p_phosphate arriving procedure

  travel to conv:sta_phosphate

  set v_phosphate_new to a_product
  if v_phosphate_old <> v_phosphate_new then
     begin
     get r_phosphate
     /*the set-up time belonging to the operation time*/
     use r_phosphate_operator for v_phosphatesetup min
     /*setup time*/
     free r_phosphate
     /*time spending on set up is included in processing*/
     print v_phosphate_new current value to lbl_test
     print v_phosphate_old current value to lbl_test1
  end

  set v_phosphate_old to a_product
  increment v_phosphate_count by 1
  /*count how many parts are processed*/
  use r_phosphate for u a_cycletimeofphosphate, 1 min
  /*actually processing time*/

  /*print a_cycletimeofphosphate current value to lbl_test*/

  send to p_dry

end


begin p_dry arriving procedure

  travel to conv:sta_drying

  if v_dry_old <> a_product then
     begin
```

```
      get r_dry
      /*the set-up time belonging to the operation time*/
      use r_dry_operator for v_dryupsetup min
      /*setup time*/
      free r_dry
      /*time spending on set up is included in processing*/
   end

   set v_dry_old to a_product
   increment v_dry_count by 1
   /*count how many parts are processed*/
   use r_dry for u a_cycletimeofdry, 1 min

   /*print a_cycletimeofphosphate current value to lbl_test*/

   send to p_basecoat

end


begin p_basecoat arriving procedure

   travel to conv:sta_basecoat

   if v_basecoat_old <> a_product then
      begin
      get r_basecoat
      /*the set-up time belonging to the operation time*/
      use r_basecoat_operator for v_basecoatsetup min
      /*setup time*/
      free r_basecoat
      /*time spending on set up is included in processing*/
   end

   set v_basecoat_old to a_product
   increment v_basecoat_count by 1
   /*count how many parts are processed*/
   use r_basecoat for u a_cycletimeofbasecoat, 3 min
   send to p_buffer1

end


begin p_buffer1 arriving procedure

   /*for storage*/
   travel to conv:sta_buffer1
   send to p_bakeoven1

end


begin p_bakeoven1 arriving procedure

   travel to conv:sta_bakeoven1

   if v_bakeoven1_old <> a_product then
```

```
      begin
      get r_bakeoven1
      /*the set-up time belonging to the operation time*/
      use r_bakeoven1_operator for v_bakeovensetup min
      /*setup time*/
      free r_bakeoven1
      /*time spending on set up is included in processing*/
   end

   set v_bakeoven1_old to a_product
   increment v_bakeoven1_count by 1
   /*count how many parts are processed*/

   use r_bakeoven1 for u a_cycletimeofbakeoven, 2 min

   send to p_cooling1

end


begin p_cooling1 arriving procedure

   travel to conv:sta_cooling1

   if v_cooling1_old <> a_product then
      begin
      get r_cooling1
      /*the set-up time belonging to the operation time*/
      use r_cooling1_operator for v_coolingsetup min
      /*setup time*/
      free r_cooling1
      /*time spending on set up is included in processing*/
   end

   set v_cooling1_old to a_product
   increment v_cooling1_count by 1
   /*count how many parts are processed*/
   use r_cooling1 for u a_cycletimeofcooling, 1 min
   send to p_sanding

end


begin p_sanding arriving procedure

   travel to conv:sta_sanding

   if v_sanding_old <> a_product then
      begin
      get r_sanding
      /*the set-up time belonging to the operation time*/
      use r_sanding_operator for v_sandingsetup min
      /*setup time*/
      free r_sanding
      /*time spending on set up is included in processing*/
   end
```

```
  set v_sanding_old to a_product
  increment v_sanding_count by 1
  /*count how many parts are processed*/
  use r_sanding for u a_cycletimeofsanding, 1 min
  send to p_topcoat

end


begin p_topcoat arriving procedure

  travel to conv:sta_topcoat

  if v_topcoat_old <> a_product then
     begin
     get r_topcoat
     /*the set-up time belonging to the operation time*/
     use r_topcoat_operator for v_topcoatsetup min
     /*setup time*/
     free r_topcoat
     /*time spending on set up is included in processing*/
  end

  set v_topcoat_old to a_product
  increment v_topcoat_count by 1
  /*count how many parts are processed*/
  use r_topcoat for u a_cycletimeoftopcoat, 3 min
  send to p_buffer2

end


begin p_buffer2 arriving procedure

  /*for storage*/
  travel to conv:sta_buffer2
  send to p_bakeoven2

end

begin p_bakeoven2 arriving procedure

  travel to conv:sta_bakeoven2

  if v_bakeoven2_old <> a_product then
     begin
     get r_bakeoven2
     /*the set-up time belonging to the operation time*/
     use r_bakeoven2_operator for v_bakeovensetup min
     /*setup time*/
     free r_bakeoven2
     /*time spending on set up is included in processing*/
  end

  set v_bakeoven2_old to a_product
  increment v_bakeoven2_count by 1
  /*count how many parts are processed*/
```

```
  use r_bakeoven2 for u a_cycletimeofbakeoven, 2 min
  send to p_cooling2

end


begin p_cooling2 arriving procedure

  travel to conv:sta_cooling2

  if v_cooling2_old <> a_product then
     begin
     get r_cooling2
     /*the set-up time belonging to the operation time*/
     use r_cooling2_operator for v_coolingsetup min
     /*setup time*/
     free r_cooling2
     /*time spending on set up is included in processing*/
  end

  set v_cooling2_old to a_product
  increment v_cooling2_count by 1
  /*count how many parts are processed*/
  use r_cooling2 for u a_cycletimeofcooling, 1 min
  send to p_inspection

end


begin p_inspection arriving procedure

  travel to conv:sta_inspection

  if v_inspection_old <> a_product then
     begin
     get r_inspection
     /*the set-up time belonging to the operation time*/
     use r_inspection_operator for v_inspectionsetup min
     /*setup time*/
     free r_inspection
     /*time spending on set up is included in processing*/
  end

  set v_inspection_old to a_product
  increment v_inspection_count by 1
  /*count how many parts are processed*/
  use r_inspection for u a_cycletimeofinspection, 1 min
  send to p_unload

end


begin p_unload arriving procedure

  travel to conv:sta_unload

  /*unloading time*/
```

```
  use r_unload for u a_cycletimeofunload, 1 min
  tabulate (ac - a_timestamp-4100)/60 in t_wip
  send to p_pbs

end


begin p_pbs arriving procedure

 move into q_pbs

 /*calculate the output*/
 increment v_throughput by 1
 send to die

end
```

APPEXDIX B: CODE OF SIMPLE HYBRID SYSTEM CONFIGURATION

```
/*read data from input file*/
begin p_init arriving procedure

  read v_junk, v_partarrivingtime,v_junk from "arc/input.txt"
  print "part_arriving_time=" v_partarrivingtime current value to
lbl_partarrivingtime

  read v_junk, v_partdieingtime,v_junk   from "arc/input.txt"
  print "part_die_time=" v_partdieingtime current value to
lbl_partdieingtime

  read v_junk, v_minordefectrate,v_junk   from "arc/input.txt"
  print "minor_defect_rate=" v_minordefectrate current value "%" to
lbl_minordefectrate

  read v_junk, v_majordefectrate,v_junk   from "arc/input.txt"
  print "major_defect_rate=" v_majordefectrate current value "%" to
lbl_majordefectrate

  read v_junk, v1_mixrate,v_junk   from "arc/input.txt"
  print "product1_portion=" v1_mixrate current value "%" to lbl1_mixrate

  read v_junk, v2_mixrate,v_junk   from "arc/input.txt"
  print "product2_portion=" v2_mixrate current value "%" to lbl2_mixrate

  read v_junk, v3_mixrate,v_junk   from "arc/input.txt"
  print "product3_portion=" v3_mixrate current value "%" to lbl3_mixrate

  read v_junk, v4_mixrate,v_junk   from "arc/input.txt"
  print "product4_portion=" v4_mixrate current value "%" to lbl4_mixrate

  read v_junk, v5_mixrate,v_junk   from "arc/input.txt"
  print "product5_portion=" v5_mixrate current value "%" to lbl5_mixrate

  read v_junk, v6_mixrate,v_junk   from "arc/input.txt"
  print "product6_portion=" v6_mixrate current value "%" to lbl6_mixrate

  read v_junk, v7_mixrate,v_junk   from "arc/input.txt"
  print "product7_portion=" v7_mixrate current value "%" to lbl7_mixrate

  read v_junk, v8_mixrate,v_junk   from "arc/input.txt"
  print "product8_portion=" v8_mixrate current value "%" to lbl8_mixrate

  read v_junk, v_phosphatesetup,v_junk   from "arc/input.txt"
  read v_junk, v_dryupsetup,v_junk   from "arc/input.txt"
  read v_junk, v_basecoatsetup,v_junk   from "arc/input.txt"
  read v_junk, v_bakeovensetup,v_junk   from "arc/input.txt"
  read v_junk, v_coolingsetup,v_junk   from "arc/input.txt"
  read v_junk, v_sandingsetup,v_junk   from "arc/input.txt"
  read v_junk, v_topcoatsetup,v_junk   from "arc/input.txt"
  read v_junk, v_inspectionsetup,v_junk   from "arc/input.txt"

  read v_junk, v_timebetweenmaintenance,v_junk   from "arc/input.txt"
```

```
  print "time_between_maintenances=" v_timebetweenmaintenance current
value "min" to lbl_timebetweenmaintenance

  read v_junk, v_maintenancetime,v_junk   from "arc/input.txt"
  print "maintenancetime=" v_maintenancetime current value "min" to
lbl_maintenancetime

  read v_junk, v1_cycletimeofphosphate,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeofdry,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeofbasecoat,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeofbaseoven,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeofcooling,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeofsanding,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeoftopcoat,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeofinspection,v_junk   from "arc/input.txt"
  read v_junk, v1_cycletimeofunload,v_junk   from "arc/input.txt"

  read v_junk, v2_cycletimeofphosphate,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeofdry,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeofbasecoat,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeofbaseoven,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeofcooling,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeofsanding,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeoftopcoat,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeofinspection,v_junk   from "arc/input.txt"
  read v_junk, v2_cycletimeofunload,v_junk   from "arc/input.txt"

  read v_junk, v3_cycletimeofphosphate,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeofdry,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeofbasecoat,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeofbaseoven,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeofcooling,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeofsanding,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeoftopcoat,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeofinspection,v_junk   from "arc/input.txt"
  read v_junk, v3_cycletimeofunload,v_junk   from "arc/input.txt"

  read v_junk, v4_cycletimeofphosphate,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeofdry,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeofbasecoat,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeofbaseoven,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeofcooling,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeofsanding,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeoftopcoat,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeofinspection,v_junk   from "arc/input.txt"
  read v_junk, v4_cycletimeofunload,v_junk   from "arc/input.txt"

  read v_junk, v5_cycletimeofphosphate,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeofdry,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeofbasecoat,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeofbaseoven,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeofcooling,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeofsanding,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeoftopcoat,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeofinspection,v_junk   from "arc/input.txt"
  read v_junk, v5_cycletimeofunload,v_junk   from "arc/input.txt"
```

```
  read v_junk, v6_cycletimeofphosphate,v_junk   from "arc/input.txt"
  read v_junk, v6_cycletimeofdry,v_junk   from "arc/input.txt"
  read v_junk, v6_cycletimeofbasecoat,v_junk   from "arc/input.txt"
  read v_junk, v6_cycletimeofbaseoven,v_junk   from "arc/input.txt"
  read v_junk, v6_cycletimeofcooling,v_junk   from "arc/input.txt"
  read v_junk, v6_cycletimeofsanding,v_junk   from "arc/input.txt"
  read v_junk, v6_cycletimeoftopcoat,v_junk   from "arc/input.txt"
  read v_junk, v6_cycletimeofinspection,v_junk   from "arc/input.txt"
  read v_junk, v6_cycletimeofunload,v_junk   from "arc/input.txt"

  read v_junk, v7_cycletimeofphosphate,v_junk   from "arc/input.txt"
  read v_junk, v7_cycletimeofdry,v_junk   from "arc/input.txt"
  read v_junk, v7_cycletimeofbasecoat,v_junk   from "arc/input.txt"
  read v_junk, v7_cycletimeofbaseoven,v_junk   from "arc/input.txt"
  read v_junk, v7_cycletimeofcooling,v_junk   from "arc/input.txt"
  read v_junk, v7_cycletimeofsanding,v_junk   from "arc/input.txt"
  read v_junk, v7_cycletimeoftopcoat,v_junk   from "arc/input.txt"
  read v_junk, v7_cycletimeofinspection,v_junk   from "arc/input.txt"
  read v_junk, v7_cycletimeofunload,v_junk   from "arc/input.txt"

  read v_junk, v8_cycletimeofphosphate,v_junk   from "arc/input.txt"
  read v_junk, v8_cycletimeofdry,v_junk   from "arc/input.txt"
  read v_junk, v8_cycletimeofbasecoat,v_junk   from "arc/input.txt"
  read v_junk, v8_cycletimeofbaseoven,v_junk   from "arc/input.txt"
  read v_junk, v8_cycletimeofcooling,v_junk   from "arc/input.txt"
  read v_junk, v8_cycletimeofsanding,v_junk   from "arc/input.txt"
  read v_junk, v8_cycletimeoftopcoat,v_junk   from "arc/input.txt"
  read v_junk, v8_cycletimeofinspection,v_junk   from "arc/input.txt"
  read v_junk, v8_cycletimeofunload,v_junk   from "arc/input.txt"

  /*print v8_cycletimeofunload current value to lbl_test
  */
  send to p_loadcreating

end


/*load creating*/
begin p_loadcreating arriving procedure

  /*creat desired number of carriers*/
  clone v_numberofcarriers loads to p_carriersstorage nlt l_carrier
  /*print v_numberofcarriers current value to lbl_test*/

  /*clone 1 load to p_shift*/  /*used for simulate the daily operation*/

  clone 1 load to p_preventivemaintenance
  /*used for simulate the preventive maintenance*/

  if v_numberofproducts = 7 then
     begin
     set v8_mixrate = 0
  end

  else if v_numberofproducts = 6 then
     begin
     set v8_mixrate = 0
```

```
   set v7_mixrate = 0
end

else if v_numberofproducts = 5 then
   begin
   set v8_mixrate = 0
   set v7_mixrate = 0
   set v6_mixrate = 0
end

else if v_numberofproducts = 4 then
   begin
   set v8_mixrate = 0
   set v7_mixrate = 0
   set v6_mixrate = 0
   set v5_mixrate = 0
end

else if v_numberofproducts = 3 then
   begin
   set v8_mixrate = 0
   set v7_mixrate = 0
   set v6_mixrate = 0
   set v5_mixrate = 0
   set v4_mixrate = 0
end


else if v_numberofproducts = 2 then
   begin
   set v8_mixrate = 0
   set v7_mixrate = 0
   set v6_mixrate = 0
   set v5_mixrate = 0
   set v4_mixrate = 0
   set v3_mixrate = 0
end

else if v_numberofproducts = 1 then
   begin
   set v8_mixrate = 0
   set v7_mixrate = 0
   set v6_mixrate = 0
   set v5_mixrate = 0
   set v4_mixrate = 0
   set v3_mixrate = 0
   set v2_mixrate = 0
end


while 1=1 do begin

  /*creating mixed products*/
  set a_product to oneof(v1_mixrate:1, v2_mixrate:2, v3_mixrate:3,
   v4_mixrate:4, v5_mixrate:5, v6_mixrate:6,v7_mixrate:7, v8_mixrate:8)

  /*set up processing parameters according to product type*/
```

```
if a_product = 1 then
   begin
   set a_cycletimeofphosphate to v1_cycletimeofphosphate
   set a_cycletimeofdry to v1_cycletimeofdry
   set a_cycletimeofbasecoat to v1_cycletimeofbasecoat
   set a_cycletimeofbakeoven to v1_cycletimeofbaseoven
   set a_cycletimeofcooling to v1_cycletimeofcooling
   set a_cycletimeofsanding to v1_cycletimeofsanding
   set a_cycletimeoftopcoat to v1_cycletimeoftopcoat
   set a_cycletimeofinspection to v1_cycletimeofinspection
   set a_cycletimeofunload to v1_cycletimeofunload
   /*print a_product current value to lbl_test*/
   clone 1 load to p_wbs nlt l_product1
end

else if a_product = 2 then
   begin
   set a_cycletimeofphosphate to v2_cycletimeofphosphate
   set a_cycletimeofdry to v2_cycletimeofdry
   set a_cycletimeofbasecoat to v2_cycletimeofbasecoat
   set a_cycletimeofbakeoven to v2_cycletimeofbaseoven
   set a_cycletimeofcooling to v2_cycletimeofcooling
   set a_cycletimeofsanding to v2_cycletimeofsanding
   set a_cycletimeoftopcoat to v2_cycletimeoftopcoat
   set a_cycletimeofinspection to v2_cycletimeofinspection
   set a_cycletimeofunload to v2_cycletimeofunload
   /*print a_product current value to lbl_test*/
   clone 1 load to p_wbs nlt l_product2
end

else if a_product = 3 then
   begin
   set a_cycletimeofphosphate to v3_cycletimeofphosphate
   set a_cycletimeofdry to v3_cycletimeofdry
   set a_cycletimeofbasecoat to v3_cycletimeofbasecoat
   set a_cycletimeofbakeoven to v3_cycletimeofbaseoven
   set a_cycletimeofcooling to v3_cycletimeofcooling
   set a_cycletimeofsanding to v3_cycletimeofsanding
   set a_cycletimeoftopcoat to v3_cycletimeoftopcoat
   set a_cycletimeofinspection to v3_cycletimeofinspection
   set a_cycletimeofunload to v3_cycletimeofunload
   clone 1 load to p_wbs nlt l_product3
end

else if a_product = 4 then
   begin
   set a_cycletimeofphosphate to v4_cycletimeofphosphate
   set a_cycletimeofdry to v4_cycletimeofdry
   set a_cycletimeofbasecoat to v4_cycletimeofbasecoat
   set a_cycletimeofbakeoven to v4_cycletimeofbaseoven
   set a_cycletimeofcooling to v4_cycletimeofcooling
   set a_cycletimeofsanding to v4_cycletimeofsanding
   set a_cycletimeoftopcoat to v4_cycletimeoftopcoat
   set a_cycletimeofinspection to v4_cycletimeofinspection
   set a_cycletimeofunload to v4_cycletimeofunload
   clone 1 load to p_wbs nlt l_product4
end
```

```
else if a_product = 5 then
   begin
   set a_cycletimeofphosphate to v5_cycletimeofphosphate
   set a_cycletimeofdry to v5_cycletimeofdry
   set a_cycletimeofbasecoat to v5_cycletimeofbasecoat
   set a_cycletimeofbakeoven to v5_cycletimeofbaseoven
   set a_cycletimeofcooling to v5_cycletimeofcooling
   set a_cycletimeofsanding to v5_cycletimeofsanding
   set a_cycletimeoftopcoat to v5_cycletimeoftopcoat
   set a_cycletimeofinspection to v5_cycletimeofinspection
   set a_cycletimeofunload to v5_cycletimeofunload
   clone 1 load to p_wbs nlt l_product5
end

else if a_product = 6 then
   begin
   set a_cycletimeofphosphate to v6_cycletimeofphosphate
   set a_cycletimeofdry to v6_cycletimeofdry
   set a_cycletimeofbasecoat to v6_cycletimeofbasecoat
   set a_cycletimeofbakeoven to v6_cycletimeofbaseoven
   set a_cycletimeofcooling to v6_cycletimeofcooling
   set a_cycletimeofsanding to v6_cycletimeofsanding
   set a_cycletimeoftopcoat to v6_cycletimeoftopcoat
   set a_cycletimeofinspection to v6_cycletimeofinspection
   set a_cycletimeofunload to v6_cycletimeofunload
   clone 1 load to p_wbs nlt l_product6
end

else if a_product = 7 then
   begin
   set a_cycletimeofphosphate to v7_cycletimeofphosphate
   set a_cycletimeofdry to v7_cycletimeofdry
   set a_cycletimeofbasecoat to v7_cycletimeofbasecoat
   set a_cycletimeofbakeoven to v7_cycletimeofbaseoven
   set a_cycletimeofcooling to v7_cycletimeofcooling
   set a_cycletimeofsanding to v7_cycletimeofsanding
   set a_cycletimeoftopcoat to v7_cycletimeoftopcoat
   set a_cycletimeofinspection to v7_cycletimeofinspection
   set a_cycletimeofunload to v7_cycletimeofunload
   clone 1 load to p_wbs nlt l_product7
end

else if a_product = 8 then
   begin
   set a_cycletimeofphosphate to v8_cycletimeofphosphate
   set a_cycletimeofdry to v8_cycletimeofdry
   set a_cycletimeofbasecoat to v8_cycletimeofbasecoat
   set a_cycletimeofbakeoven to v8_cycletimeofbaseoven
   set a_cycletimeofcooling to v8_cycletimeofcooling
   set a_cycletimeofsanding to v8_cycletimeofsanding
   set a_cycletimeoftopcoat to v8_cycletimeoftopcoat
   set a_cycletimeofinspection to v8_cycletimeofinspection
   set a_cycletimeofunload to v8_cycletimeofunload
   clone 1 load to p_wbs nlt l_product8
end
```

```
    wait for e v_partarrivingtime min
    /*time between load arrivals*/

  end

end


begin p_carriersstorage arriving procedure

  move into q_carriers
  wait to be ordered on ol_carriers
  /*waiting for parts arrival*/

end


/*begin p_shift arriving procedure

  /*simulate the operation time 8 hours per day*/
  while 1 = 1 do

    begin
    wait for 7.8 hr
    take down r1_basecoat
    take down r1_topcoat
    take down r2_basecoat
    take down r2_topcoat

    wait for 0.2 hr
    take down r1_bakeoven1
    take down r1_bakeoven2
    take down r2_bakeoven1
    take down r2_bakeoven2

    take down r_load
    take down r_phosphate
    take down r_dry

    take down r1_cooling1
    take down r1_sanding
    take down r1_cooling2
    take down r1_inspection

    take down r2_cooling1
    take down r2_sanding
    take down r2_cooling2
    take down r2_inspection

    take down r_unload

    wait for 16 hr
    bring up r1_basecoat
    bring up r1_topcoat
    bring up r2_basecoat
    bring up r2_topcoat
    bring up r1_bakeoven1
```

```
    bring up r1_bakeoven2
    bring up r2_bakeoven1
    bring up r2_bakeoven2
    bring up r_load
    bring up r_phosphate
    bring up r_dry
    bring up r1_cooling1
    bring up r1_sanding
    bring up r1_cooling2
    bring up r1_inspection
    bring up r2_cooling1
    bring up r2_sanding
    bring up r2_cooling2
    bring up r2_inspection
    bring up r_unload

  end

end*/


begin p_preventivemaintenance arriving procedure

  while 1 = 1 do
    begin
    wait for v_timebetweenmaintenance hr
    take down r1_basecoat
    take down r1_topcoat
    take down r2_basecoat
    take down r2_topcoat

    wait for 0.1 hr
    take down r1_bakeoven1
    take down r1_bakeoven2
    take down r2_bakeoven1
    take down r2_bakeoven2

    take down r_load
    take down r_phosphate
    take down r_dry

    take down r1_cooling1
    take down r1_sanding
    take down r1_cooling2
    take down r1_inspection

    take down r2_cooling1
    take down r2_sanding
    take down r2_cooling2
    take down r2_inspection

    take down r_unload

    wait for v_maintenancetime hr
    bring up r1_basecoat
    bring up r1_topcoat
    bring up r2_basecoat
```

```
    bring up r2_topcoat
    bring up r1_bakeoven1
    bring up r1_bakeoven2
    bring up r2_bakeoven1
    bring up r2_bakeoven2
    bring up r_load
    bring up r_phosphate
    bring up r_dry
    bring up r1_cooling1
    bring up r1_sanding
    bring up r1_cooling2
    bring up r1_inspection
    bring up r2_cooling1
    bring up r2_sanding
    bring up r2_cooling2
    bring up r2_inspection
    bring up r_unload
  end

end


begin p_wbs arriving procedure

 move into q_wbs /*check the average number in wbs*/
 send to p_loading

end


begin p_loading arriving procedure

   /*make sure there is carrier in carriers storage*/
  wait until ol_carriers current value > 0
  order 1 load from ol_carriers to die

  move into conv:sta_load
  use r_load for u 3, 1 min
  /*fix white truck body to the carrier*/
  send to p_phosphate

end


begin p_phosphate arriving procedure

  travel to conv:sta_phosphate

  if v_phosphate_count = 0 then
     begin
     set v_phosphate_old to 1 /*initializing the start value*/
  end

  set v_phosphate_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
```

```
  if v_phosphate_old <> v_phosphate_new then
     begin
     get r_phosphate
     /*the set-up time belonging to the operation time*/
     use r_phosphate_operator for v_phosphatesetup min /*setup time*/
     free r_phosphate /*time spending on set up is included in
processing*/
  end

  set v_phosphate_old to a_product
  increment v_phosphate_count by 1 /*count how many parts are processed*/
  use r_phosphate for u a_cycletimeofphosphate, 1 min /*actually
processing time*/

  /*print a_cycletimeofphosphate current value to lbl_test*/

  send to p_dry

end


begin p_dry arriving procedure

  travel to conv:sta_drying

  if v_dry_count = 0 then
     begin
     set v_dry_old to 1 /*initializing the start value*/
  end

  set v_dry_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v_dry_old <> v_dry_new then
     begin
     get r_dry
     /*the set-up time belonging to the operation time*/
     use r_dry_operator for v_dryupsetup min /*setup time*/
     free r_dry /*time spending on set up is included in processing*/
  end

  set v_dry_old to a_product
  increment v_dry_count by 1 /*count how many parts are processed*/
  use r_dry for u a_cycletimeofdry, 1 min

  /*print a_cycletimeofphosphate current value to lbl_test*/

  send to p_grouping

end


begin p_grouping arriving procedure

  if v_numberofproducts = 8 then
     begin /*grouping several types of product into basecoat1*/
```

```
    if a_product = 1 then
       begin
       send to p1_basecoat
    end
    else if a_product = 2 then
       begin
       send to p1_basecoat
    end
    else if a_product = 7 then
       begin
       send to p1_basecoat
    end
    else if a_product = 8 then
       begin
       send to p1_basecoat
    end

    else
    send to p2_basecoat

end

if v_numberofproducts = 7 then
  begin /*grouping several types of product into basecoat1*/

    if a_product = 1 then
       begin
       send to p1_basecoat
    end
    else if a_product = 3 then
       begin
       send to p1_basecoat
    end
    else if a_product = 7 then
       begin
       send to p1_basecoat
    end

    else
    send to p2_basecoat


end

if v_numberofproducts = 6 then
  begin /*grouping several types of product into basecoat1*/

    if a_product = 1 then
       begin
       send to p1_basecoat
    end
    else if a_product = 3 then
       begin
       send to p1_basecoat
    end
    else if a_product = 6 then
```

```
      begin
      send to p1_basecoat
   end

   else
   send to p2_basecoat

end

if v_numberofproducts = 5 then
  begin /*grouping several types of product into basecoat1*/

   if a_product = 4 then
      begin
      send to p1_basecoat
   end
   else if a_product = 5 then
      begin
      send to p1_basecoat
   end

   else
   send to p2_basecoat

end

if v_numberofproducts = 4 then
  begin /*grouping several types of product into basecoat1*/

   if a_product = 1 then
      begin
      send to p1_basecoat
   end
   else if a_product = 4 then
      begin
      send to p1_basecoat
   end

   else
   send to p2_basecoat
end

if v_numberofproducts = 3 then
  begin /*grouping several types of product into basecoat1*/

   if a_product = 1 then
      begin
      send to p1_basecoat
   end
   else if a_product = 2 then
      begin
      send to p1_basecoat
   end

   else
   send to p2_basecoat
```

```
    end

    if v_numberofproducts = 2 then
      begin /*grouping several types of product into basecoat1*/

        if a_product = 1 then
           begin
           send to p1_basecoat
        end

        else if a_product = 2 then
           begin
           send to p2_basecoat
        end

    end

    if v_numberofproducts = 1 then
      begin
        send to oneof(1:p1_basecoat,1:p2_basecoat)
    end

end


begin p1_basecoat arriving procedure

  /*check where this load come from*/
  if a_site = 0 then
  begin
  travel to conv:sta_transfer1
  move into pm.cp1
  end

  /*check where this load come from*/
  if a_site = 1 then
  begin
  travel to conv:sta_transfer5
  move into pm.cp2
  end

  travel to pm.cp1
  move into conv:sta_transfer2
  travel to conv:sta1_basecoat

  if v1_basecoat_count = 0 then
     begin
     set v1_basecoat_old to 1 /*initializing the start value*/
  end

  set v1_basecoat_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v1_basecoat_old <> v1_basecoat_new then
     begin
     get r1_basecoat
```

```
     /*the set-up time belonging to the operation time*/
     use r1_basecoat_operator for e v_basecoatsetup min /*setup time*/
     free r1_basecoat /*time spending on set up is included in
processing*/
  end

  set v1_basecoat_old to a_product
  increment v1_basecoat_count by 1 /*count how many parts are processed*/
  use r1_basecoat for u a_cycletimeofbasecoat, 3 min

  send to p1_buffer1

end


begin p1_buffer1 arriving procedure

  /*for storage*/
  travel to conv:sta1_buffer1
  send to p1_bakeoven1

end


begin p1_bakeoven1 arriving procedure

  travel to conv:sta1_bakeoven1

  if v1_bakeoven1_count = 0 then
     begin
     set v1_bakeoven1_old to 1 /*initializing the start value*/
  end

  set v1_bakeoven1_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v1_bakeoven1_old <> v1_bakeoven1_new then
     begin
     get r1_bakeoven1
     /*the set-up time belonging to the operation time*/
     use r1_bakeoven1_operator for e v_bakeovensetup min /*setup time*/
     free r1_bakeoven1 /*time spending on set up is included in
processing*/
  end

  set v1_bakeoven1_old to a_product
  increment v1_bakeoven1_count by 1 /*count how many parts are processed*/

  use r1_bakeoven1 for u a_cycletimeofbakeoven, 2 min

  send to p1_cooling1

end


begin p1_cooling1 arriving procedure
```

```
  travel to conv:sta1_cooling1

  if v1_cooling1_count = 0 then
     begin
     set v1_cooling1_old to 1 /*initializing the start value*/
  end

  set v1_cooling1_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v1_cooling1_old <> v1_cooling1_new then
     begin
     get r1_cooling1
     /*the set-up time belonging to the operation time*/
     use r1_cooling1_operator for e v_coolingsetup min /*setup time*/
     free r1_cooling1 /*time spending on set up is included in
processing*/
  end

  set v1_cooling1_old to a_product
  increment v1_cooling1_count by 1 /*count how many parts are processed*/

  use r1_cooling1 for u a_cycletimeofcooling, 1 min

  send to p1_sanding

end


begin p1_sanding arriving procedure

  travel to conv:sta1_sanding

  if v1_sanding_count = 0 then
     begin
     set v1_sanding_old to 1 /*initializing the start value*/
  end

  set v1_sanding_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v1_sanding_old <> v1_sanding_new then
     begin
     get r1_sanding
     /*the set-up time belonging to the operation time*/
     use r1_sanding_operator for e v_sandingsetup min /*setup time*/
     free r1_sanding /*time spending on set up is included in processing*/
  end

  set v1_sanding_old to a_product
  increment v1_sanding_count by 1 /*count how many parts are processed*/

  use r1_sanding for u a_cycletimeofsanding, 1 min
```

```
  send to p1_buffer2

end

begin p1_buffer2 arriving procedure

  /*for storage*/
  travel to conv:sta_transfer3
  move into pm2.cp5
  travel to pm2.cp6
  move into conv:sta_transfer4
  travel to conv:sta1_buffer2
  send to p1_topcoat

end

begin p1_topcoat arriving procedure

  travel to conv:sta1_topcoat

  if v1_topcoat_count = 0 then
     begin
     set v1_topcoat_old to 1 /*initializing the start value*/
  end

  set v1_topcoat_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v1_topcoat_old <> v1_topcoat_new then
     begin
     get r1_topcoat
     /*the set-up time belonging to the operation time*/
     use r1_topcoat_operator for e v_topcoatsetup min /*setup time*/
     free r1_topcoat /*time spending on set up is included in processing*/
  end

  set v1_topcoat_old to a_product
  increment v1_topcoat_count by 1 /*count how many parts are processed*/

  use r1_topcoat for u a_cycletimeoftopcoat, 3 min

  send to p1_bakeoven2

end


begin p1_bakeoven2 arriving procedure

  travel to conv:sta1_bakeoven2

  if v1_bakeoven2_count = 0 then
     begin
     set v1_bakeoven2_old to 1 /*initializing the start value*/
  end

  set v1_bakeoven2_new to a_product
```

```
  /*check whether the comming part is the same as the last one to set up
facility*/
  if v1_bakeoven2_old <> v1_bakeoven2_new then
     begin
     get r1_bakeoven2
     /*the set-up time belonging to the operation time*/
     use r1_bakeoven2_operator for e v_bakeovensetup min /*setup time*/
     free r1_bakeoven2 /*time spending on set up is included in
processing*/
  end

  set v1_bakeoven2_old to a_product
  increment v1_bakeoven2_count by 1 /*count how many parts are processed*/

  use r1_bakeoven2 for u a_cycletimeofbakeoven, 2 min

  send to p1_cooling2

end


begin p1_cooling2 arriving procedure

  travel to conv:sta1_cooling2

  if v1_cooling2_count = 0 then
     begin
     set v1_cooling2_old to 1 /*initializing the start value*/
  end

  set v1_cooling2_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v1_cooling2_old <> v1_cooling2_new then
     begin
     get r1_cooling2
     /*the set-up time belonging to the operation time*/
     use r1_cooling2_operator for e v_coolingsetup min /*setup time*/
     free r1_cooling2 /*time spending on set up is included in
processing*/
  end

  set v1_cooling2_old to a_product
  increment v1_cooling2_count by 1 /*count how many parts are processed*/

  use r1_cooling2 for u a_cycletimeofcooling, 1 min

  send to p1_inspection

end


begin p1_inspection arriving procedure

  travel to conv:sta1_inspectrepair
```

```
  if v1_inspection_count = 0 then
     begin
     set v1_inspection_old to 1 /*initializing the start value*/
  end

  set v1_inspection_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v1_inspection_old <> v1_inspection_new then
     begin
     get r1_inspection
     /*the set-up time belonging to the operation time*/
     use r1_inspection_operator for e v_inspectionsetup min /*setup time*/
     free r1_inspection /*time spending on set up is included in
processing*/
  end

  set v1_inspection_old to a_product
  increment v1_inspection_count by 1 /*count how many parts are
processed*/

  use r1_inspection for u a_cycletimeofinspection, 1 min

  set a_site to 1

  /*send to oneof(v_majordefectrate:p1_basecoat,(100-
v_majordefectrate):p_unload)*/
  send to p_unload

end


begin p_unload arriving procedure

  if a_site = 1 then
     begin
     travel to conv:sta_transfer5
     move into pm.cp2
  end

   if a_site = 2 then
     begin
     travel to conv:sta_transfer9
     move into pm.cp4
  end

  travel to pm.cp3
  move into conv:sta_transfer10
  travel to conv:sta_unload

  /*unloading time*/
  use r_unload for u a_cycletimeofunload, 1 min

  /*send carrier back to carrier storage*/
  clone 1 load to p_carriersstorage nlt l_carrier
```

```
  /*send painted body to storage*/
  send to p_pbs

end


begin p_pbs arriving procedure

 move into q_pbs

 /*calculate the output*/
 increment v_throughput by 1

 set v_average_rate to (v_throughput*60)/ac
 tabulate v_average_rate in t_average_rate
 /*send to other work shop*/
 wait for u  v_partdieingtime, 1 min
 send to die

end


begin p2_basecoat arriving procedure

  /*check where this load come from*/
  if a_site = 0 then
  begin
  travel to conv:sta_transfer1
  move into pm.cp1
  end

  /*check where this load come from*/
  if a_site = 2 then
  begin
  travel to conv:sta_transfer9
  move into pm.cp4
  end

  travel to pm.cp3
  move into conv:sta_transfer6
  travel to conv:sta2_basecoat

  if v2_basecoat_count = 0 then
     begin
     set v2_basecoat_old to 1 /*initializing the start value*/
  end

  set v2_basecoat_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v2_basecoat_old <> v2_basecoat_new then
     begin
     get r2_basecoat
     /*the set-up time belonging to the operation time*/
     use r2_basecoat_operator for e v_basecoatsetup min /*setup time*/
```

```
     free r2_basecoat /*time spending on set up is included in
processing*/
  end

  set v2_basecoat_old to a_product
  increment v2_basecoat_count by 1 /*count how many parts are processed*/
  use r2_basecoat for u a_cycletimeofbasecoat, 3 min

  send to p2_buffer1

end


begin p2_buffer1 arriving procedure

  /*for storage*/
  travel to conv:sta2_buffer1
  send to p2_bakeoven1

end


begin p2_bakeoven1 arriving procedure

  travel to conv:sta2_bakeoven1

  if v2_bakeoven1_count = 0 then
     begin
     set v2_bakeoven1_old to 1 /*initializing the start value*/
  end

  set v2_bakeoven1_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v2_bakeoven1_old <> v2_bakeoven1_new then
     begin
     get r2_bakeoven1
     /*the set-up time belonging to the operation time*/
     use r2_bakeoven1_operator for e v_bakeovensetup min /*setup time*/
     free r2_bakeoven1 /*time spending on set up is included in
processing*/
  end

  set v2_bakeoven1_old to a_product
  increment v2_bakeoven1_count by 1 /*count how many parts are processed*/

  use r2_bakeoven1 for u a_cycletimeofbakeoven, 2 min

  send to p2_cooling1

end


begin p2_cooling1 arriving procedure

  travel to conv:sta2_cooling1
```

```
  if v2_cooling1_count = 0 then
     begin
     set v2_cooling1_old to 1 /*initializing the start value*/
  end

  set v2_cooling1_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v2_cooling1_old <> v2_cooling1_new then
     begin
     get r2_cooling1
     /*the set-up time belonging to the operation time*/
     use r2_cooling1_operator for e v_coolingsetup min /*setup time*/
     free r2_cooling1 /*time spending on set up is included in
processing*/
  end

  set v2_cooling1_old to a_product
  increment v2_cooling1_count by 1 /*count how many parts are processed*/

  use r2_cooling1 for u a_cycletimeofcooling, 1 min

  send to p2_sanding

end


begin p2_sanding arriving procedure

  travel to conv:sta2_sanding

  if v2_sanding_count = 0 then
     begin
     set v2_sanding_old to 1 /*initializing the start value*/
  end

  set v2_sanding_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v2_sanding_old <> v2_sanding_new then
     begin
     get r2_sanding
     /*the set-up time belonging to the operation time*/
     use r2_sanding_operator for e v_sandingsetup min /*setup time*/
     free r2_sanding /*time spending on set up is included in processing*/
  end

  set v2_sanding_old to a_product
  increment v2_sanding_count by 1 /*count how many parts are processed*/

  use r2_sanding for u a_cycletimeofsanding, 1 min

  send to p2_buffer2
```

```
end

begin p2_buffer2 arriving procedure

  /*for storage*/
  travel to conv:sta_transfer7
  move into pm2.cp7
  travel to pm2.cp8
  move into conv:sta_transfer8
  travel to conv:sta2_buffer2
  send to p2_bakeoven2

end

begin p2_topcoat arriving procedure


  travel to conv:sta2_topcoat

  if v2_topcoat_count = 0 then
     begin
     set v2_topcoat_old to 1 /*initializing the start value*/
  end

  set v2_topcoat_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v2_topcoat_old <> v2_topcoat_new then
     begin
     get r2_topcoat
     /*the set-up time belonging to the operation time*/
     use r2_topcoat_operator for e v_topcoatsetup min /*setup time*/
     free r2_topcoat /*time spending on set up is included in processing*/
  end

  set v2_topcoat_old to a_product
  increment v2_topcoat_count by 1 /*count how many parts are processed*/

  use r2_topcoat for u a_cycletimeoftopcoat, 3 min

  send to p2_bakeoven2

end


begin p2_bakeoven2 arriving procedure

  travel to conv:sta2_bakeoven2

  if v2_bakeoven2_count = 0 then
     begin
     set v2_bakeoven2_old to 1 /*initializing the start value*/
  end

  set v2_bakeoven2_new to a_product
```

```
  /*check whether the comming part is the same as the last one to set up
facility*/
  if v2_bakeoven2_old <> v2_bakeoven2_new then
     begin
     get r2_bakeoven2
     /*the set-up time belonging to the operation time*/
     use r2_bakeoven2_operator for e v_bakeovensetup min /*setup time*/
     free r2_bakeoven2 /*time spending on set up is included in
processing*/
  end

  set v2_bakeoven2_old to a_product
  increment v2_bakeoven2_count by 1 /*count how many parts are processed*/

  use r2_bakeoven2 for u a_cycletimeofbakeoven, 2 min

  send to p2_cooling2

end


begin p2_cooling2 arriving procedure

  travel to conv:sta2_cooling2

  if v2_cooling2_count = 0 then
     begin
     set v2_cooling2_old to 1 /*initializing the start value*/
  end

  set v2_cooling2_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v2_cooling2_old <> v2_cooling2_new then
     begin
     get r2_cooling2
     /*the set-up time belonging to the operation time*/
     use r2_cooling2_operator for e v_coolingsetup min /*setup time*/
     free r2_cooling2 /*time spending on set up is included in
processing*/
  end

  set v2_cooling2_old to a_product
  increment v2_cooling2_count by 1 /*count how many parts are processed*/

  use r2_cooling2 for u a_cycletimeofcooling, 1 min

  send to p2_inspection

end


begin p2_inspection arriving procedure

  travel to conv:sta2_inspectrepair
```

```
  if v2_inspection_count = 0 then
     begin
     set v2_inspection_old to 1 /*initializing the start value*/
  end

  set v2_inspection_new to a_product

  /*check whether the comming part is the same as the last one to set up
facility*/
  if v2_inspection_old <> v2_inspection_new then
     begin
     get r2_inspection
     /*the set-up time belonging to the operation time*/
     use r2_inspection_operator for e v_inspectionsetup min /*setup time*/
     free r2_inspection /*time spending on set up is included in
processing*/
  end

  set v2_inspection_old to a_product
  increment v2_inspection_count by 1 /*count how many parts are
processed*/

  use r2_inspection for u a_cycletimeofinspection, 1 min

  set a_site to 2

  /*send to oneof(v_majordefectrate:p2_basecoat,(100-
v_majordefectrate):p_unload)*/
  send to p_unload

end
```

APPEXDIX C: CODE OF DEFINING OPTIMAL BUFFER LOCATIONS

```
begin p_init arriving procedure

  read v_junk, v1_mixrate, v_junk from "arc/input.txt"
  read v_junk, v2_mixrate, v_junk from "arc/input.txt"
  read v_junk, v3_mixrate, v_junk from "arc/input.txt"
  read v_junk, v4_mixrate, v_junk from "arc/input.txt"

  read v_junk, v1_cycle_basecoat, v_junk from "arc/input.txt"
  read v_junk, v1_cycle_bakeoven, v_junk from "arc/input.txt"
  read v_junk, v1_cycle_cooling, v_junk from "arc/input.txt"
  read v_junk, v1_cycle_sanding, v_junk from "arc/input.txt"
  read v_junk, v1_cycle_topcoat, v_junk from "arc/input.txt"
  read v_junk, v1_cycle_inspection, v_junk from "arc/input.txt"

  read v_junk, v2_cycle_basecoat, v_junk from "arc/input.txt"
  read v_junk, v2_cycle_bakeoven, v_junk from "arc/input.txt"
  read v_junk, v2_cycle_cooling, v_junk from "arc/input.txt"
  read v_junk, v2_cycle_sanding, v_junk from "arc/input.txt"
  read v_junk, v2_cycle_topcoat, v_junk from "arc/input.txt"
  read v_junk, v2_cycle_inspection, v_junk from "arc/input.txt"

  read v_junk, v3_cycle_basecoat, v_junk from "arc/input.txt"
  read v_junk, v3_cycle_bakeoven, v_junk from "arc/input.txt"
  read v_junk, v3_cycle_cooling, v_junk from "arc/input.txt"
  read v_junk, v3_cycle_sanding, v_junk from "arc/input.txt"
  read v_junk, v3_cycle_topcoat, v_junk from "arc/input.txt"
  read v_junk, v3_cycle_inspection, v_junk from "arc/input.txt"

  read v_junk, v4_cycle_basecoat, v_junk from "arc/input.txt"
  read v_junk, v4_cycle_bakeoven, v_junk from "arc/input.txt"
  read v_junk, v4_cycle_cooling, v_junk from "arc/input.txt"
  read v_junk, v4_cycle_sanding, v_junk from "arc/input.txt"
  read v_junk, v4_cycle_topcoat, v_junk from "arc/input.txt"
  read v_junk, v4_cycle_inspection, v_junk from "arc/input.txt"

  print v4_cycle_inspection current value to lbl_test
  send to p_loadcreating

end


begin p_loadcreating arriving procedure


  while 1 = 1 do
    begin
    set a_product to oneof(v1_mixrate:1,v2_mixrate:2,v3_mixrate:3
                           v4_mixrate:4)

    if a_product = 1 then
       begin
       set a_cycle_basecoat to v1_cycle_basecoat
       set a_cycle_bakeoven to v1_cycle_bakeoven
```

```
      set a_cycle_cooling to v1_cycle_cooling
      set a_cycle_sanding to v1_cycle_sanding
      set a_cycle_topcoat to v1_cycle_topcoat
      set a_cycle_inspection to v1_cycle_inspection
      clone 1 load to p_buffer1 nlt l_product1
   end

   if a_product = 2 then
      begin
      set a_cycle_basecoat to v2_cycle_basecoat
      set a_cycle_bakeoven to v2_cycle_bakeoven
      set a_cycle_cooling to v2_cycle_cooling
      set a_cycle_sanding to v2_cycle_sanding
      set a_cycle_topcoat to v2_cycle_topcoat
      set a_cycle_inspection to v2_cycle_inspection
      clone 1 load to p_buffer1 nlt l_product2
   end


   if a_product = 3 then
      begin
      set a_cycle_basecoat to v3_cycle_basecoat
      set a_cycle_bakeoven to v3_cycle_bakeoven
      set a_cycle_cooling to v3_cycle_cooling
      set a_cycle_sanding to v3_cycle_sanding
      set a_cycle_topcoat to v3_cycle_topcoat
      set a_cycle_inspection to v3_cycle_inspection
      clone 1 load to p_buffer1 nlt l_product3
   end

   if a_product = 4 then
      begin
      set a_cycle_basecoat to v4_cycle_basecoat
      set a_cycle_bakeoven to v4_cycle_bakeoven
      set a_cycle_cooling to v4_cycle_cooling
      set a_cycle_sanding to v4_cycle_sanding
      set a_cycle_topcoat to v4_cycle_topcoat
      set a_cycle_inspection to v4_cycle_inspection
      clone 1 load to p_buffer1 nlt l_product4
   end

   wait for e 2 min

  end

end

begin p_buffer1 arriving procedure

  move into q_buffer1
  send to p_basecoat

end

begin p_basecoat arriving procedure

  move into q_basecoat
```

```
  use r_basecoat for u a_cycle_basecoat, 3 min
  send to p_buffer2

end


begin p_buffer2 arriving procedure

  move into q_buffer2
  send to p_bakeoven1

end

begin p_bakeoven1 arriving procedure

  move into q_bakeoven1
  use r_bakeoven1 for u a_cycle_bakeoven,2 min
  send to p_cooling1

end


begin p_cooling1 arriving procedure

  move into q_cooling1
  use r_cooling1 for u a_cycle_cooling, 1 min
  send to p_sanding

end

begin p_sanding arriving procedure

  move into q_sanding
  use r_sanding for u a_cycle_sanding, 1 min
  send to p_topcoat

end

begin p_topcoat arriving procedure

  move into q_topcoat
  use r_topcoat for u a_cycle_topcoat, 3 min
  send to p_bakeoven2

end


begin p_bakeoven2 arriving procedure

  move into q_bakeoven2
  use r_bakeoven2 for u a_cycle_bakeoven,2 min
  send to p_cooling2

end


begin p_cooling2 arriving procedure
```

```
  move into q_cooling2
  use r_cooling2 for u a_cycle_cooling, 1 min
  send to p_inspection

end

begin p_inspection arriving procedure

  move into q_inspection
  use r_inspection for u a_cycle_inspection, 1 min
  send to die

end
```

APPEXDIX D: EXPERIMENTAL DATA OF THE SERIAL SYSTEM CONFIGURATION

number_of_product_types: 1										
	Run 1	Run 9	Run 17	Run 25	Run 33	Run 41	Run 49	Run 57	Run 65	Run 73
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
set-up percentage for bake oven 1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
set-up percentage for bake oven 2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
set-up percentage for base coat	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
set-up percentage for cooling 1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
set-up percentage for cooling 2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
set-up percentage for dry up	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
set-up percentage for inpection and repai	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
set-up percentage for phosphate	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
set-up percentage for sanding	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
set-up percentage for top coat	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
throughput_per_hour	7.062	7.077	7.056	7.072	7.067	7.062	7.077	7.064	7.067	7.072
work in process	38.8502	38.5868	38.7411	38.7349	38.6039	38.6226	38.7099	38.8286	38.7044	38.5059
number_of_product_types: 2										
	Run 2	Run 10	Run 18	Run 26	Run 34	Run 42	Run 50	Run 58	Run 66	Run 74
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
set-up percentage for bake oven 1	0.26	0.26	0.26	0.27	0.26	0.27	0.27	0.27	0.27	0.27
set-up percentage for bake oven 2	0.26	0.26	0.26	0.27	0.26	0.27	0.27	0.27	0.27	0.27
set-up percentage for base coat	0.26	0.26	0.26	0.27	0.26	0.27	0.27	0.27	0.27	0.27
set-up percentage for cooling 1	0.31	0.31	0.3	0.31	0.3	0.31	0.31	0.31	0.32	0.31
set-up percentage for cooling 2	0.31	0.31	0.3	0.31	0.3	0.31	0.31	0.31	0.32	0.31
set-up percentage for dry up	0.18	0.17	0.17	0.18	0.17	0.18	0.18	0.18	0.18	0.18
set-up percentage for inpection and repai	0.26	0.26	0.26	0.27	0.26	0.27	0.27	0.27	0.27	0.27
set-up percentage for phosphate	0.15	0.15	0.15	0.16	0.15	0.15	0.16	0.16	0.16	0.16
set-up percentage for sanding	0.35	0.35	0.35	0.36	0.35	0.35	0.36	0.36	0.36	0.36
set-up percentage for top coat	0.35	0.35	0.35	0.36	0.35	0.35	0.36	0.36	0.36	0.36
throughput_per_hour	2.669	2.667	2.684	2.655	2.665	2.664	2.659	2.659	2.659	2.671
work in process	191.119	191.1175	189.7549	192.5289	191.6756	191.4529	191.9159	192.4319	192.3348	191.0808
number_of_product_types: 3										
	Run 3	Run 11	Run 19	Run 27	Run 35	Run 43	Run 51	Run 59	Run 67	Run 75
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
set-up percentage for bake oven 1	0.34	0.34	0.34	0.33	0.34	0.34	0.34	0.34	0.34	0.34
set-up percentage for bake oven 2	0.34	0.34	0.35	0.33	0.34	0.34	0.34	0.34	0.34	0.34
set-up percentage for base coat	0.34	0.34	0.34	0.33	0.34	0.34	0.34	0.34	0.34	0.34
set-up percentage for cooling 1	0.39	0.4	0.4	0.39	0.39	0.39	0.39	0.4	0.4	0.4
set-up percentage for cooling 2	0.39	0.39	0.4	0.39	0.39	0.39	0.39	0.4	0.4	0.4
set-up percentage for dry up	0.22	0.23	0.23	0.22	0.22	0.22	0.22	0.23	0.23	0.23
set-up percentage for inpection and repai	0.34	0.34	0.35	0.33	0.34	0.34	0.34	0.34	0.34	0.34
set-up percentage for phosphate	0.2	0.2	0.2	0.19	0.2	0.2	0.2	0.2	0.2	0.2
set-up percentage for sanding	0.45	0.45	0.46	0.44	0.45	0.45	0.45	0.45	0.45	0.46
set-up percentage for top coat	0.45	0.45	0.46	0.44	0.45	0.45	0.45	0.45	0.45	0.46
throughput_per_hour	2.541	2.544	2.536	2.544	2.536	2.534	2.544	2.538	2.533	2.53
work in process	206.8121	206.5322	207.6223	206.1277	207.3622	207.4698	206.4227	207.1461	207.7871	208.3086

APPEXDIX D, CONT'D

number_of_product_types: 4										
	Run 4	Run 12	Run 20	Run 28	Run 36	Run 44	Run 52	Run 60	Run 68	Run 76
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
set-up percentage for bake oven 1	0.38	0.37	0.37	0.37	0.37	0.37	0.38	0.37	0.37	0.37
set-up percentage for bake oven 2	0.38	0.37	0.37	0.37	0.37	0.37	0.38	0.37	0.37	0.37
set-up percentage for base coat	0.38	0.37	0.37	0.37	0.37	0.37	0.38	0.37	0.37	0.37
set-up percentage for cooling 1	0.44	0.43	0.43	0.43	0.43	0.43	0.44	0.43	0.43	0.43
set-up percentage for cooling 2	0.44	0.43	0.43	0.43	0.43	0.43	0.44	0.43	0.43	0.43
set-up percentage for dry up	0.25	0.25	0.24	0.25	0.24	0.25	0.25	0.25	0.25	0.25
set-up percentage for inpection and repai	0.38	0.37	0.37	0.37	0.37	0.37	0.38	0.37	0.37	0.37
set-up percentage for phosphate	0.22	0.22	0.21	0.22	0.21	0.22	0.22	0.22	0.22	0.22
set-up percentage for sanding	0.5	0.49	0.49	0.49	0.49	0.49	0.5	0.5	0.49	0.49
set-up percentage for top coat	0.5	0.49	0.49	0.49	0.49	0.49	0.5	0.5	0.49	0.49
throughput_per_hour	2.476	2.487	2.491	2.484	2.481	2.486	2.486	2.482	2.49	2.489
work in process	215.1789	214.1072	213.598	214.2568	214.1582	213.9908	214.3218	214.6074	213.6156	213.741
number_of_product_types: 5										
	Run 5	Run 13	Run 21	Run 29	Run 37	Run 45	Run 53	Run 61	Run 69	Run 77
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
set-up percentage for bake oven 1	0.4	0.39	0.4	0.39	0.39	0.39	0.39	0.39	0.39	0.39
set-up percentage for bake oven 2	0.4	0.39	0.4	0.39	0.39	0.39	0.39	0.39	0.39	0.39
set-up percentage for base coat	0.4	0.39	0.4	0.39	0.39	0.39	0.39	0.39	0.39	0.39
set-up percentage for cooling 1	0.46	0.45	0.46	0.45	0.46	0.46	0.46	0.46	0.46	0.46
set-up percentage for cooling 2	0.46	0.45	0.46	0.45	0.46	0.46	0.46	0.46	0.46	0.46
set-up percentage for dry up	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26
set-up percentage for inpection and repai	0.4	0.39	0.4	0.39	0.39	0.39	0.39	0.39	0.39	0.39
set-up percentage for phosphate	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23
set-up percentage for sanding	0.53	0.52	0.53	0.52	0.52	0.52	0.52	0.52	0.52	0.52
set-up percentage for top coat	0.53	0.52	0.53	0.52	0.52	0.52	0.52	0.52	0.52	0.52
throughput_per_hour	2.455	2.459	2.451	2.458	2.449	2.455	2.458	2.456	2.454	2.454
work in process	218.6001	218.0531	219.1956	218.2143	219.2604	218.4716	218.1969	218.3967	218.715	218.7595
number_of_product_types: 6										
	Run 6	Run 14	Run 22	Run 30	Run 38	Run 46	Run 54	Run 62	Run 70	Run 78
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
set-up percentage for bake oven 1	0.41	0.4	0.4	0.4	0.4	0.4	0.41	0.41	0.41	0.41
set-up percentage for bake oven 2	0.41	0.4	0.4	0.4	0.4	0.4	0.41	0.41	0.41	0.41
set-up percentage for base coat	0.41	0.4	0.4	0.4	0.4	0.4	0.41	0.41	0.41	0.41
set-up percentage for cooling 1	0.48	0.47	0.47	0.47	0.47	0.47	0.48	0.48	0.48	0.48
set-up percentage for cooling 2	0.48	0.47	0.47	0.47	0.47	0.47	0.48	0.48	0.48	0.48
set-up percentage for dry up	0.28	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27
set-up percentage for inpection and repai	0.41	0.4	0.4	0.4	0.4	0.4	0.41	0.41	0.41	0.41
set-up percentage for phosphate	0.24	0.24	0.24	0.24	0.24	0.23	0.24	0.24	0.24	0.24
set-up percentage for sanding	0.55	0.54	0.54	0.54	0.54	0.54	0.54	0.54	0.54	0.55
set-up percentage for top coat	0.55	0.54	0.54	0.54	0.54	0.54	0.54	0.54	0.54	0.55
throughput_per_hour	2.434	2.434	2.437	2.432	2.436	2.436	2.438	2.429	2.432	2.437
work in process	221.977	221.5244	221.4186	221.7921	221.305	221.4569	221.2539	222.3817	222.2751	221.6295

APPEXDIX D, CONT'D

number_of_product_types: 7										
	Run 7	Run 15	Run 23	Run 31	Run 39	Run 47	Run 55	Run 63	Run 71	Run 79
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
set-up percentage for bake oven 1	0.42	0.41	0.42	0.42	0.41	0.41	0.41	0.42	0.41	0.42
set-up percentage for bake oven 2	0.42	0.41	0.42	0.42	0.41	0.41	0.41	0.42	0.41	0.42
set-up percentage for base coat	0.41	0.41	0.42	0.42	0.41	0.41	0.41	0.42	0.41	0.42
set-up percentage for cooling 1	0.48	0.48	0.49	0.48	0.48	0.48	0.48	0.48	0.48	0.49
set-up percentage for cooling 2	0.48	0.48	0.49	0.48	0.48	0.48	0.48	0.48	0.48	0.49
set-up percentage for dry up	0.28	0.28	0.28	0.28	0.28	0.27	0.28	0.28	0.27	0.28
set-up percentage for inpection and repai	0.41	0.41	0.42	0.42	0.41	0.41	0.41	0.42	0.41	0.42
set-up percentage for phosphate	0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24
set-up percentage for sanding	0.55	0.55	0.56	0.55	0.55	0.55	0.55	0.55	0.55	0.56
set-up percentage for top coat	0.55	0.55	0.56	0.55	0.55	0.55	0.55	0.55	0.55	0.56
throughput_per_hour	2.417	2.417	2.414	2.415	2.42	2.42	2.421	2.413	2.424	2.413
work in process	224.0226	224.1882	224.6188	224.2874	223.6537	223.6924	223.5234	224.5728	223.2239	224.6255
number_of_product_types: 8										
	Run 8	Run 16	Run 24	Run 32	Run 40	Run 48	Run 56	Run 64	Run 72	Run 80
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
set-up percentage for bake oven 1	0.42	0.42	0.43	0.42	0.42	0.42	0.42	0.42	0.42	0.42
set-up percentage for bake oven 2	0.42	0.42	0.43	0.42	0.42	0.42	0.42	0.42	0.42	0.42
set-up percentage for base coat	0.42	0.42	0.43	0.42	0.42	0.42	0.42	0.42	0.42	0.42
set-up percentage for cooling 1	0.5	0.49	0.5	0.49	0.5	0.49	0.49	0.49	0.49	0.49
set-up percentage for cooling 2	0.49	0.49	0.5	0.49	0.5	0.49	0.49	0.49	0.49	0.49
set-up percentage for dry up	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28
set-up percentage for inpection and repai	0.42	0.42	0.43	0.42	0.42	0.42	0.42	0.42	0.42	0.42
set-up percentage for phosphate	0.25	0.25	0.25	0.24	0.25	0.24	0.25	0.25	0.25	0.25
set-up percentage for sanding	0.57	0.56	0.57	0.56	0.57	0.56	0.56	0.56	0.56	0.56
set-up percentage for top coat	0.57	0.56	0.57	0.56	0.57	0.56	0.56	0.56	0.56	0.56
throughput_per_hour	2.402	2.406	2.402	2.405	2.405	2.406	2.405	2.402	2.402	2.404
work in process	226.2588	225.795	226.321	225.989	225.9885	225.871	226.0561	226.3561	226.2041	226.0741

APPEXDIX E: EXPERIMENTAL DATA OF THE SIMPLE HYBRID SYSTEM CONFIGURATION

number_of_products: 1										
	Run 1	Run 9	Run 17	Run 25	Run 33	Run 41	Run 49	Run 57	Run 65	Run 73
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
throughput_per_hour	6.169	6.191	6.222	6.145	6.142	6.194	6.221	6.243	6.239	6.201
work in process	43.8663	43.7361	43.6628	44.0995	43.9243	43.787	43.5717	43.6378	43.5932	43.7038
number_of_products: 2										
	Run 2	Run 10	Run 18	Run 26	Run 34	Run 42	Run 50	Run 58	Run 66	Run 74
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
throughput_per_hour	4.311	4.293	4.307	4.317	4.307	4.319	4.322	4.305	4.315	4.314
work in process	52.3012	52.2408	52.3304	52.2819	52.2605	52.2095	52.348	52.2711	52.2125	52.0818
number_of_products: 3										
	Run 3	Run 11	Run 19	Run 27	Run 35	Run 43	Run 51	Run 59	Run 67	Run 75
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
throughput_per_hour	3.694	3.67	3.677	3.669	3.643	3.668	3.724	3.724	3.665	3.706
work in process	107.427	108.7431	106.7698	107.014	110.8525	106.5581	105.9195	103.0634	110.6002	107.2849
number_of_products: 4										
	Run 4	Run 12	Run 20	Run 28	Run 36	Run 44	Run 52	Run 60	Run 68	Run 76
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
throughput_per_hour	3.717	3.663	3.65	3.678	3.722	3.688	3.693	3.674	3.683	3.709
work in process	117.5743	120.0771	120.5385	120.737	114.9256	120.0751	118.0985	117.5402	119.0649	117.4928
number_of_products: 5										
	Run 5	Run 13	Run 21	Run 29	Run 37	Run 45	Run 53	Run 61	Run 69	Run 77
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
throughput_per_hour	3.517	3.478	3.512	3.52	3.475	3.498	3.498	3.534	3.49	3.504
work in process	132.5799	134.6949	132.9624	133.2126	134.2224	132.6628	133.6139	131.0818	132.4609	133.7991
number_of_products: 6										
	Run 6	Run 14	Run 22	Run 30	Run 38	Run 46	Run 54	Run 62	Run 70	Run 78
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
throughput_per_hour	3.485	3.513	3.504	3.495	3.469	3.5	3.507	3.53	3.458	3.513
work in process	137.0448	137.0997	137.8437	138.3107	137.7079	138.5002	138.1096	135.5919	138.9167	135.6164
number_of_products: 7										
	Run 7	Run 15	Run 23	Run 31	Run 39	Run 47	Run 55	Run 63	Run 71	Run 79
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
throughput_per_hour	3.366	3.369	3.348	3.359	3.396	3.373	3.412	3.417	3.37	3.396
work in process	147.765	146.7173	148.3504	147.8626	143.9134	146.5754	146.3668	142.3516	146.0875	143.3613
number_of_products: 8										
	Run 8	Run 16	Run 24	Run 32	Run 40	Run 48	Run 56	Run 64	Run 72	Run 80
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
throughput_per_hour	3.38	3.366	3.373	3.384	3.402	3.377	3.394	3.371	3.374	3.402
work in process	148.5687	147.4548	147.4596	146.7708	147.0013	148.859	146.6686	145.7873	147.9499	147.932

APPENDIX F: EXPERIMENTAL DATA OF THE EFFECT OF VARYING THE NUMBER OF CARRIERS ON THROUGHPUT IN SIMPLE HYBRID SYSTEM CONFIGURATION

number_of_products: 8, number_of_carriers: 7										
	Run 81	Run 105	Run 129	Run 153	Run 177	Run 201	Run 225	Run 249	Run 273	Run 297
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	2.032	2.03	2.027	2.031	2.036	2.034	2.04	2.035	2.032	2.034
number_of_products: 8, number_of_carriers: 8										
	Run 82	Run 106	Run 130	Run 154	Run 178	Run 202	Run 226	Run 250	Run 274	Run 298
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	2.265	2.272	2.273	2.276	2.282	2.289	2.284	2.279	2.275	2.279
number_of_products: 8, number_of_carriers: 9										
	Run 83	Run 107	Run 131	Run 155	Run 179	Run 203	Run 227	Run 251	Run 275	Run 299
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	2.503	2.499	2.515	2.512	2.507	2.524	2.535	2.501	2.534	2.498
number_of_products: 8, number_of_carriers: 10										
	Run 84	Run 108	Run 132	Run 156	Run 180	Run 204	Run 228	Run 252	Run 276	Run 300
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	2.726	2.708	2.732	2.741	2.732	2.724	2.728	2.719	2.73	2.702
number_of_products: 8, number_of_carriers: 11										
	Run 85	Run 109	Run 133	Run 157	Run 181	Run 205	Run 229	Run 253	Run 277	Run 301
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	2.925	2.926	2.898	2.922	2.923	2.937	2.934	2.933	2.95	2.919
number_of_products: 8, number_of_carriers: 12										
	Run 86	Run 110	Run 134	Run 158	Run 182	Run 206	Run 230	Run 254	Run 278	Run 302
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.07	3.089	3.078	3.092	3.096	3.102	3.089	3.11	3.11	3.093
number_of_products: 8, number_of_carriers: 13										
	Run 87	Run 111	Run 135	Run 159	Run 183	Run 207	Run 231	Run 255	Run 279	Run 303
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.254	3.233	3.204	3.223	3.239	3.235	3.231	3.236	3.231	3.248
number_of_products: 8, number_of_carriers: 14										
	Run 88	Run 112	Run 136	Run 160	Run 184	Run 208	Run 232	Run 256	Run 280	Run 304
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.289	3.333	3.308	3.338	3.309	3.308	3.345	3.303	3.314	3.311
number_of_products: 8, number_of_carriers: 15										
	Run 89	Run 113	Run 137	Run 161	Run 185	Run 209	Run 233	Run 257	Run 281	Run 305
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.35	3.4	3.369	3.345	3.365	3.375	3.344	3.36	3.369	3.362
number_of_products: 8, number_of_carriers: 16										
	Run 90	Run 114	Run 138	Run 162	Run 186	Run 210	Run 234	Run 258	Run 282	Run 306
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.368	3.399	3.398	3.392	3.365	3.394	3.401	3.366	3.382	3.425
number_of_products: 8, number_of_carriers: 17										
	Run 91	Run 115	Run 139	Run 163	Run 187	Run 211	Run 235	Run 259	Run 283	Run 307
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.384	3.385	3.395	3.418	3.356	3.348	3.377	3.384	3.396	3.394

APPENDIX F, CONT'D

number_of_products: 8, number_of_carriers: 18										
	Run 92	Run 116	Run 140	Run 164	Run 188	Run 212	Run 236	Run 260	Run 284	Run 308
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.342	3.375	3.389	3.408	3.399	3.393	3.352	3.374	3.377	3.393
number_of_products: 8, number_of_carriers: 19										
	Run 93	Run 117	Run 141	Run 165	Run 189	Run 213	Run 237	Run 261	Run 285	Run 309
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.342	3.375	3.385	3.398	3.399	3.393	3.345	3.374	3.386	3.393
number_of_products: 8, number_of_carriers: 20										
	Run 94	Run 118	Run 142	Run 166	Run 190	Run 214	Run 238	Run 262	Run 286	Run 310
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.342	3.375	3.385	3.398	3.399	3.393	3.345	3.374	3.386	3.393
number_of_products: 8, number_of_carriers: 21										
	Run 95	Run 119	Run 143	Run 167	Run 191	Run 215	Run 239	Run 263	Run 287	Run 311
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.342	3.375	3.385	3.398	3.399	3.393	3.345	3.374	3.386	3.393
number_of_products: 8, number_of_carriers: 22										
	Run 96	Run 120	Run 144	Run 168	Run 192	Run 216	Run 240	Run 264	Run 288	Run 312
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.342	3.375	3.385	3.398	3.399	3.393	3.345	3.374	3.386	3.393
number_of_products: 8, number_of_carriers: 23										
	Run 97	Run 121	Run 145	Run 169	Run 193	Run 217	Run 241	Run 265	Run 289	Run 313
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.342	3.375	3.385	3.398	3.399	3.393	3.345	3.374	3.386	3.393
number_of_products: 8, number_of_carriers: 24										
	Run 98	Run 122	Run 146	Run 170	Run 194	Run 218	Run 242	Run 266	Run 290	Run 314
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.342	3.375	3.385	3.398	3.399	3.393	3.345	3.374	3.386	3.393
number_of_products: 8, number_of_carriers: 25										
	Run 99	Run 123	Run 147	Run 171	Run 195	Run 219	Run 243	Run 267	Run 291	Run 315
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.342	3.375	3.385	3.398	3.399	3.393	3.345	3.374	3.386	3.393
number_of_products: 8, number_of_carriers: 26										
	Run 100	Run 124	Run 148	Run 172	Run 196	Run 220	Run 244	Run 268	Run 292	Run 316
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.342	3.375	3.385	3.398	3.399	3.393	3.345	3.374	3.386	3.393
number_of_products: 8, number_of_carriers: 27										
	Run 101	Run 125	Run 149	Run 173	Run 197	Run 221	Run 245	Run 269	Run 293	Run 317
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.342	3.375	3.385	3.398	3.399	3.393	3.345	3.374	3.386	3.393
number_of_products: 8, number_of_carriers: 28										
	Run 102	Run 126	Run 150	Run 174	Run 198	Run 222	Run 246	Run 270	Run 294	Run 318
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_hour	3.342	3.375	3.385	3.398	3.399	3.393	3.345	3.374	3.386	3.393

APPEXDIX G: EXPERIMENTAL DATA OF THROUGHPUT OF DIFFERENT BUFFER-COMBINATIONS

	Run 1	Run 2	Run 3	Run 4	Run 5	Run 6	Run 7	Run 8	Run 9	Run 10
Throughput per hour of (1,2)	3.5	3.49	3.51	3.49	3.49	3.5	3.5	3.49	3.49	3.5
Throughput per hour of (1,3)	3.49	3.5	3.51	3.5	3.51	3.51	3.51	3.49	3.49	3.51
Throughput per hour of (1,4)	3.51	3.52	3.53	3.52	3.51	3.53	3.52	3.52	3.53	3.53
Throughput per hour of (1,5)	3.56	3.56	3.58	3.56	3.55	3.55	3.55	3.56	3.56	3.58
Throughput per hour of (1,6)	3.49	3.48	3.48	3.5	3.48	3.48	3.48	3.49	3.49	3.48
Throughput per hour of (1,7)	3.42	3.44	3.44	3.44	3.41	3.43	3.43	3.44	3.42	3.45
Throughput per hour of (1,8)	3.41	3.42	3.42	3.41	3.43	3.42	3.41	3.42	3.41	3.41
Throughput per hour of (1,9)	3.41	3.42	3.43	3.42	3.41	3.41	3.43	3.4	3.41	3.43
Throughput per hour of (2,3)	3.5	3.5	3.52	3.52	3.53	3.51	3.51	3.51	3.53	3.54
Throughput per hour of (2,4)	3.53	3.56	3.54	3.55	3.55	3.53	3.55	3.54	3.56	3.55
Throughput per hour of (2,5)	3.67	3.66	3.68	3.68	3.67	3.68	3.68	3.66	3.67	3.68
Throughput per hour of (2,6)	3.63	3.63	3.63	3.63	3.63	3.65	3.64	3.63	3.63	3.64
Throughput per hour of (2,7)	3.53	3.53	3.54	3.54	3.52	3.51	3.55	3.51	3.53	3.54
Throughput per hour of (2,8)	3.48	3.48	3.53	3.49	3.49	3.5	3.51	3.48	3.51	3.52
Throughput per hour of (2,9)	3.49	3.49	3.5	3.5	3.51	3.52	3.51	3.51	3.51	3.5
Throughput per hour of (3,4)	3.53	3.55	3.54	3.54	3.53	3.52	3.54	3.52	3.55	3.56
Throughput per hour of (3,5)	3.6	3.61	3.61	3.61	3.61	3.62	3.61	3.61	3.62	3.62
Throughput per hour of (3,6)	3.62	3.62	3.64	3.61	3.6	3.61	3.61	3.6	3.61	3.61
Throughput per hour of (3,7)	3.53	3.53	3.54	3.54	3.53	3.54	3.55	3.51	3.54	3.54
Throughput per hour of (3,8)	3.5	3.52	3.53	3.49	3.49	3.52	3.52	3.51	3.52	3.52
Throughput per hour of (3,9)	3.5	3.52	3.54	3.51	3.5	3.51	3.5	3.48	3.5	3.52
Throughput per hour of (4,5)	3.59	3.58	3.61	3.59	3.59	3.6	3.6	3.59	3.61	3.6
Throughput per hour of (4,6)	3.6	3.62	3.63	3.61	3.59	3.61	3.62	3.6	3.61	3.61
Throughput per hour of (4,7)	3.55	3.54	3.57	3.54	3.54	3.55	3.55	3.54	3.55	3.55
Throughput per hour of (4,8)	3.52	3.52	3.53	3.52	3.51	3.51	3.53	3.51	3.53	3.52
Throughput per hour of (4,9)	3.51	3.52	3.53	3.52	3.51	3.53	3.52	3.52	3.53	3.53
Throughput per hour of (5,6)	3.63	3.61	3.63	3.62	3.61	3.62	3.62	3.61	3.62	3.63
Throughput per hour of (5,7)	3.6	3.58	3.59	3.57	3.59	3.59	3.58	3.58	3.59	3.59
Throughput per hour of (5,8)	3.55	3.55	3.58	3.57	3.54	3.56	3.57	3.55	3.57	3.57
Throughput per hour of (5,9)	3.56	3.56	3.58	3.56	3.55	3.55	3.55	3.56	3.56	3.58
Throughput per hour of (6,7)	3.47	3.49	3.49	3.49	3.47	3.49	3.5	3.48	3.48	3.48
Throughput per hour of (6,8)	3.52	3.52	3.52	3.46	3.38	3.46	3.48	3.48	3.44	3.5
Throughput per hour of (6,9)	3.49	3.48	3.5	3.48	3.47	3.49	3.48	3.48	3.5	3.48
Throughput per hour of (7,8)	3.43	3.43	3.45	3.43	3.43	3.43	3.43	3.43	3.41	3.45
Throughput per hour of (7,9)	3.42	3.44	3.44	3.44	3.41	3.43	3.43	3.44	3.42	3.45
Throughput per hour of (8,9)	3.41	3.42	3.42	3.41	3.43	3.42	3.41	3.42	3.41	3.41

APPEXDIX H: EXPERIMENTAL DATA OF FINAL THROUGHPUT PER MONTH OF THE SIMPLE HYBRID SYSTEM CONFIGURATION

number_of_product_types: 1										
	Run 1	Run 2	Run 3	Run 4	Run 5	Run 6	Run 7	Run 8	Run 9	Run 10
	RN Set 1	RN Set 2	RN Set 3	RN Set 4	RN Set 5	RN Set 6	RN Set 7	RN Set 8	RN Set 9	RN Set 10
throughput_per_month	842.688	837.936	841.104	845.856	843.744	837.408	838.992	825.792	831.6	840.048
number_of_product_types: 2										
	Run 11	Run 12	Run 13	Run 14	Run 15	Run 16	Run 17	Run 18	Run 19	Run 20
	RN Set 11	RN Set 12	RN Set 13	RN Set 14	RN Set 15	RN Set 16	RN Set 17	RN Set 18	RN Set 19	RN Set 20
throughput_per_month	652.08	648.384	661.584	650.496	652.608	652.608	655.776	660.528	657.36	650.496
number_of_product_types: 3										
	Run 21	Run 22	Run 23	Run 24	Run 25	Run 26	Run 27	Run 28	Run 29	Run 30
	RN Set 21	RN Set 22	RN Set 23	RN Set 24	RN Set 25	RN Set 26	RN Set 27	RN Set 28	RN Set 29	RN Set 30
throughput_per_month	553.344	555.984	553.344	571.824	544.368	554.4	550.176	554.4	546.48	558.096
number_of_product_types: 4										
	Run 31	Run 32	Run 33	Run 34	Run 35	Run 36	Run 37	Run 38	Run 39	Run 40
	RN Set 31	RN Set 32	RN Set 33	RN Set 34	RN Set 35	RN Set 36	RN Set 37	RN Set 38	RN Set 39	RN Set 40
throughput_per_month	550.176	548.064	548.064	564.432	550.176	551.76	547.536	541.2	543.84	549.648
number_of_product_types: 5										
	Run 41	Run 42	Run 43	Run 44	Run 45	Run 46	Run 47	Run 48	Run 49	Run 50
	RN Set 41	RN Set 42	RN Set 43	RN Set 44	RN Set 45	RN Set 46	RN Set 47	RN Set 48	RN Set 49	RN Set 50
throughput_per_month	531.696	523.776	528	525.888	521.664	523.248	523.776	528	524.832	521.664
number_of_product_types: 6										
	Run 51	Run 52	Run 53	Run 54	Run 55	Run 56	Run 57	Run 58	Run 59	Run 60
	RN Set 51	RN Set 52	RN Set 53	RN Set 54	RN Set 55	RN Set 56	RN Set 57	RN Set 58	RN Set 59	RN Set 60
throughput_per_month	512.16	516.912	535.92	516.912	514.272	519.024	516.912	516.912	519.024	522.192
number_of_product_types: 7										
	Run 61	Run 62	Run 63	Run 64	Run 65	Run 66	Run 67	Run 68	Run 69	Run 70
	RN Set 61	RN Set 62	RN Set 63	RN Set 64	RN Set 65	RN Set 66	RN Set 67	RN Set 68	RN Set 69	RN Set 70
throughput_per_month	513.216	511.104	507.408	516.912	511.632	506.88	515.856	500.016	506.88	503.184
number_of_product_types: 8										
	Run 71	Run 72	Run 73	Run 74	Run 75	Run 76	Run 77	Run 78	Run 79	Run 80
	RN Set 71	RN Set 72	RN Set 73	RN Set 74	RN Set 75	RN Set 76	RN Set 77	RN Set 78	RN Set 79	RN Set 80
throughput_per_month	496.32	497.376	498.432	493.68	505.296	501.6	500.016	499.488	495.792	498.432

REFERENCE

J. M. Alden, Estimating performance of two workstations in series with downtime and unequal speeds, Research Publication R&D-9434, General Motors R&D Center, Warren, Michigan, 2002.

Robert F. Webbink and S. Jack Hu, Automated Generation of Assembly System-Design Solutions, IEEE Transactions on Automation Science and Engineering, 2005

SAWYER, J.H.F., Line Balancing by J.H.F.SAWYER, Brighton: Machinery Publishing, 1970

Stephen G. Powell and David F. Pyke, Allocation of Buffers to serial Production Lines with Bottlenecks, IIE Transactions, 1996

Zhun Zhang, Maintenance Planning and Cost Effective Replacement Strategies, University of Alberta, 2005

Prasad S. Mahajan, Ricki G. Ingalls, Evaluation of Methods Used to Detect Warm-up Period in Steady State Simulation, Proceeding of the 2004 winter simulation conference

David W.Graehl, Insights into Carrier Control: a Simulation of a Power and Free Conveyor Through an Automotive Paint Shop, Proceedings of the 1992 Simulation Conference

Conway, R. W., Maxwell, W. L., McClain, J. O. and Thomas, L. J., the ROLE OF Work-in-Process Inventories in Serial Production Lines. Operations Research, 1988, 36, 229-241

Jingshan Li, Throughput Analysis in Automotive Paint Shops: A Case Study. IEEE Transactions on Automation Science and Engineering, Vol. 1, No. 1, July 2004

VITA AUCTORIS

NAME:	Guangming Qiu
PLACE OF BIRTH:	Nong'an China
YEAR OF BIRTH:	1974
EDUCATION:	Nong'an Experimental High School, China
	1989-1992
	Taiyuan Heavy Machine University, China
	1992-1996 B.Sc
	University of Windsor, Windsor, Ontario
	2005-2006 M.Sc