DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

A HIGH-LEVEL PETRI NET BASED DECISION SUPPORT SYSTEM FOR REAL-TIME SCHEDULING AND CONTROL OF FLEXIBLE MANUFACTURING SYSTEMS: AN OBJECT-ORIENTED APPROACH

by

Gonca TUNÇEL

December, 2005

İZMİR

A HIGH-LEVEL PETRI NET BASED DECISION SUPPORT SYSTEM FOR REAL-TIME SCHEDULING AND CONTROL OF FLEXIBLE MANUFACTURING SYSTEMS: AN OBJECT-ORIENTED APPROACH

A Thesis Submitted to the

Graduate School of Natural and Applied Sciences of Dokuz Eylül University

In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Industrial Engineering, Industrial Engineering Program

by

Gonca TUNÇEL

December, 2005

İZMİR

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled "**A HIGH-LEVEL PETRI NET BASED DECISION SUPPORT SYSTEM FOR REAL-TIME SCHEDULING AND CONTROL OF FLEXIBLE MANUFACTURING SYSTEMS: AN OBJECT-ORIENTED APPROACH**" completed by **GONCA TUNÇEL** under supervision of **Prof. Dr. GUNHAN MİRAÇ BAYHAN** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

Prof.Dr.G. Miraç BAYHAN

Supervisor

Prof.Dr.Demir ASLAN	Assist.Prof.Dr.Adil ALPKOÇAK
Thesis Committee Member	Thesis Committee Member
Assist.Prof.Dr.Aşkıner GÜNGÖR	Assist.Prof.Dr.Mehmet ÇAKMAKÇI
Examining Committee Member	Examining Committee Member

Prof.Dr. Cahit HELVACI
Director
Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENTS

First and foremost I would like to express my deep gratitude and thanks to my advisor Prof.Dr.G.Miraç BAYHAN for her continuous support, guidance, and valuable advice throughout the progress of this dissertation.

I sincerely acknowledge and thank the members of my thesis committee, Prof.Dr.Demir ASLAN and Assist.Pof.Dr.Adil ALPKOÇAK, for their helpful comments, encouragement, and suggestions.

I gratefully thank Anil PANICKER, Enrico GHILLINO, and Mehmet BAYRAK for providing me the evaluation license of Artifex.

I would also like to thank my friend Derya EREN AKYOL and all my colleagues for their support, encouragement, and friendship.

Finally, I would like to express my indebtedness and many thanks to my parents, Fevzi and Ayten TUNÇEL, and my sister Ayça for their confidence, encouragement and endless support in my whole life.

Gonca TUNÇEL

A HIGH-LEVEL PETRI NET BASED DECISION SUPPORT SYSTEM FOR REAL-TIME SCHEDULING AND CONTROL OF FLEXIBLE MANUFACTURING SYSTEMS: AN OBJECT-ORIENTED APPROACH

ABSTRACT

Increasing automation and complexity of today's production systems affect the constraints of real-world scheduling problems. To deal with this challenge, the existing scheduling techniques are improved or new scheduling approaches are developed in recent years. In this thesis, we attempt to propose an object-oriented approach for modeling and analysis of shop floor scheduling problem of FMSs using high-level PNs. The graphical nature and mathematical foundation has made Petri net based methods appealing in real-time scheduling and control of flexible manufacturing systems (FMSs). In the proposed approach, we firstly construct an object modeling diagram of an FMS and develop a heuristic rule-base to solve the resource contention problem, then formulate the dynamic behavior of the system by high-level PNs. The modeling methodology is illustrated by an FMS, and the proposed rule-based system is compared with several dispatching rules with respect to part flow time and tardiness related performance measures. Finally, the system performance under different operational configurations is analyzed to find out the best implementing policy for the system under consideration.

Keywords: Petri nets, Object-oriented modeling, Flexible manufacturing systems, Real-time scheduling.

ESNEK İMALAT SİSTEMLERİNİN GERÇEK ZAMANLI ÇİZELGELEME VE KONTROL PROBLEMİ İÇİN YÜKSEK SEVİYE PETRİ AĞLARINA DAYALI BİR KARAR DESTEK SİSTEMİ: NESNEYE YÖNELİK BİR YAKLAŞIM

ÖZ

Günümüz imalat sistemlerinin artan otomasyon ve karmaşık yapısı gerçek hayat çizelgeleme problemlerinin kısıtlarını da etkilemektedir. Bu kısıtlarla başa çıkabilmek için, son yıllarda mevcut çizelgeleme teknikleri iyileştirilmekte veya yeni çizelgeleme yaklaşımları geliştirilmektedir. Bu tezde, esnek imalat sistemlerinin atölye bazlı çizelgeleme probleminin modellenmesi ve analizi için yüksek seviye Petri ağları kullanılarak nesneye yönelik bir yaklaşım önerilmiştir. Grafiksel yapısı ve matematiksel temeli, Petri ağlarını esnek imalat sistemlerinin gerçek zamanlı çizelgeleme ve kontrol problemleri için cazip hale getirmektedir. Önerilen yaklaşımda, ilk olarak bir esnek imalat sisteminin nesne modelleme diyagramı oluşturulur ve kaynak paylaşım problemini çözmek için sezgisel kural esaslı bir yaklaşım geliştirilir, daha sonra sistemin dinamik yapısı yüksek seviye Petri ağlarıyla formüle edilir. Modelleme metodolojisi bir esnek imalat sistemi üzerinde gösterilmiştir ve önerilen kural esaslı sistem, parça akış zamanı ve gecikme ile ilişkili performans ölçütlerine göre bazı sevk etme kurallarıyla karşılaştırılmıştır. Son olarak, sistem performansı farklı yapılandırmalar altında analiz edilerek, ele alınan sistem için en iyi işletim politikası belirlenmeye çalışılmıştır.

Anahtar sözcükler: Petri ağları, Nesneye yönelik modelleme, Esnek imalat sistemleri, Gerçek zamanlı çizelgeleme.

CONTENTS

Page

CHAPTER ONE
INTRODUCTION

In this chapter, the background, motivation and objectives of this work are stated, and the organization of this dissertation is outlined.

1.1 Background and Motivation

Today, the manufacturing environment is characterized as having diverse products, high quality, short delivery time and unstable customer demand. In order to provide wide product variety and quick response to changes in marketplace, flexible manufacturing systems (FMSs) have been adopted broadly in modern production environments (Chen & Chen, 2003). An FMS consists of numerically controlled machining centers linked together by an automated material handling system, and can be quickly configured to produce multiple type of products. Flexibility in a manufacturing system provides various advantages such as increased productivity, reduced work-in-process inventory, reduced lead times, increased machine utilization, and reduced set-up costs. Despite these productivity improvements, FMSs induce new management and control problems encountered through the design, planning, scheduling, and control (Saygin, Chen, Singh, 2001).

Scheduling and control problem of FMS differs considerably from the problems appeared in traditional flow shop and job-shop environments due to the different set of operating conditions (Sivagnanavelu, 2000). The main characteristics of an FMS include multi-layer resource sharing and routing flexibility of the jobs. Machine routings in process plan specify the machines that are required for each operation of a given job. Routing flexibility means the capability of processing a part type using alternate routings through machines. Thus the same operation can be processed on

different machines so parts can follow various machine sequencing. As Byrne & Chutima (1997) explain that when a flexible process-planning model is embedded into the scheduling paradigm, the solution space of the scheduling problem is enlarged due to the range of options generated by the use of alternative plans. While the routing flexibility increases the complexity of scheduling and control of FMSs, it provides important gain on the system performance like increase in throughput rate, handling unexpected situations (i.e. machine breakdowns, tool problems, and demand changes). In order to achieve the full capacity of a flexible manufacturing system with routing flexibility, an effective and efficient alternate machine selection policy is needed. Moreover, concurrent flow of multiple jobs and complex interaction of the limited resources can cause deadlock situations. For this reason, scheduling and control methods for FMSs should also consider deadlock conditions (Xiong, 1996).

Due to the high complexity of FMSs, some mathematical programming related methods such as integer programming, linear programming, and dynamic programming often lacks to describe the practical constraints, and dynamic features of complex scheduling problems, or lack of providing analytical solutions within a reasonable time. Besides, analytical methods usually require repetition as the status of the system changes, and assume that all the parts are ready at time zero, which is rarely the case in practice. On the other hand, classical scheduling techniques such as branch-and-bound and neighbourhood search techniques cause substantial increase in state enumeration, and exponentially growing time in computation as the problem size increases (Jeng & Chen, 1998). This makes them computationally inefficient to be real-time decision criteria for alternate route selection.

Recently, either existing techniques are improved or new scheduling approaches are developed to deal with the practical constraints of the real-world scheduling problems of FMSs. These methods include the simulation based approaches, expert systems, Petri nets based methods, AI-based search techniques, and hybrid methods (Lin & Lee, 1997). Petri Nets (PNs) have been extended and applied to the broader

range of problems including the scheduling of production systems by means of their modeling capabilities and formulation advantages (Zhou & DiCesare, 1993). PNs can be explicitly and concisely model concurrent, asynchronous activities, multi-layer resource sharing, routing flexibility, limited buffers, and precedence constraints in manufacturing systems (Murata, 1989; Zurawski & Zhou, 1994). Activities, resources, and constraints of a manufacturing system can be represented in a single coherent formulation. Particularly, in the sequencing and scheduling problems, PNs provide a better understanding of dynamic behavior of the operations owing to token flow and the execution of transition firings. The scheduling problem is thus to find a firing sequence which reaches the goal state in minimum time. PNs can also provide an explicit way for considering deadlock situations, and thus facilitate deadlock free scheduling of FMSs (Xiong, 1996).

However, modeling and analysis of complex or large size systems require too much effort, since considerable number of states and transition requirements cause state explosion problem. Furthermore, PNs are generally system dependent and lacks some features like modularity, maintability, and reusability, which are the properties of Object-Oriented approach (Wang, 1996). However, object-oriented models lack the capability of mathematically analyzing the control logic of a system (Wang & Wu, 1998). Therefore, recently, there has been a growing interest in merging PNs and object-oriented approach to combine graphical representation and mathematical foundation of PNs with the modularity, reusability, and maintability features of object-orientation.

On the other hand, scheduling decisions are influenced by various factors such as process plans, due date requirements, release dates, job priorities, machine setup requirements, number of machines, operators, and material handling system. Scheduling and shop-floor control decisions must be capable of handling simultaneously these diverse factors. Therefore, rather than designing an optimum scheduler, there is a definitive need for a flexible and integrated scheduling in order to handle the dynamic and stochastic nature of real-world. To deal with the real-time scheduling problem of FMSs is the driving force behind this thesis.

1.2 Research Objective

Here, we present a PN based Decision Support System (DSS) for real-time shop floor scheduling and control of FMSs, which have part routing flexibility and machine setup operations. High-level PNs and Object Oriented Design (OOD) approach were used for system modeling, and a heuristic rule (knowledge) based approach was employed for scheduling /dispatching in control logic. The presented DSS helps managers to take control decisions effectively and efficiently by considering the current status of the shop floor. Flexibility and quick response to the changes in demand are major advantages of the proposed methodology. The developed rule-based DSS aims to solve the resource contention problem, and to determine the best route(s) of the parts, which have routing flexibility. Decisions are taken on real-time basis by checking product types, due dates, alternative process plans, next possible destination resource, setup status, and resource utilizations rates. The DSS takes a global view of the system state before making decision about resource assignment, and proposes a dynamic approach to solve the conflict problems. It has adaptive and autonomous ability for obtaining intelligent control, and can be used for different production systems by only changing some system parameters (i.e. number of operators and stations, types of products, and process plan information) in Object Classes. Performances of different system configurations are examined in order to find out the best implementing policy.

The proposed methodology is illustrated on an example FMS. Finally the performance of the proposed rule-based system is compared with some dispatching rules such as earliest due date (EDD), first-in-first-out (FIFO), largest number of remaining operations (LNRO), smallest number of remaining operations (SNRO), and critical ratio (CR). Many researchers have employed PNs for scheduling and control of manufacturing systems (Hatano, Yamagata, & Tamura, 1991; Lee & DiCesare, 1994; Chetty & Gnanasekaran, 1996; Song & Lee, 1998; Jain, 2001; Moro, Yu, & Kelleher, 2002; Jain, Swarnkar, & Tiwari, 2003; Yu, Reyes, Cang, & Lloyd, 2003). However, Petri-net based dynamic scheduling of FMSs dealing with

production of several product types with flexible routes, setup times, material handling system, and operator constraints has not been given much attention. Briefly, the goal of this dissertation is to develop a High-level PN based approach for deadlock free scheduling and control of FMSs with routing flexibility and machine setup times. The specific objectives are as follows:

- To develop a modeling methodology capable of describing complicated relations among jobs, machines and alternative routing for the shop-floor scheduling and control problem based on high-level Petri Nets and Object Modeling Technique.

- To propose a Decision Support System based on Object Oriented Petri Nets in order to solve the scheduling and control problem in a flexible manufacturing system that includes flexible routing, setup times, and multiple batch sizes.

- To propose a heuristic rule based approach to determine the best route of the jobs, which has some alternative routes on the workstations by considering the setup requirements of the machines in the system.

- To investigate an FMS for the implementation of the proposed modeling methodology and the rule-based system.

- To describe alternative system configuration such as, number of operator, number of material handling device, different dispatching rules, and heuristics for part routing in order to evaluate the performance of the integrated model under different system configuration.

- To examine the system performance for various policies in a multi-batch type manufacturing system to find out the best implementing policy with respect to the considered measures, and to analyze the implementing results of the

proposed heuristic rule-based system, and to compare with several dispatching rules.

1.3 Organization of the Thesis

This dissertation is organized as follows. In chapter two, an overview of Petri nets and a review of the recent works on PN applications in production scheduling are given, and both theoretical developments and practical experiences are discussed. Chapter three includes detailed explanation of the modeling methodology and implementation of the proposed Decision Support System. Finally, chapter four concludes the dissertation and provides potential extensions and future work.

CHAPTER TWO
PETRI NETS AND APPLICATIONS IN PRODUCTION SCHEDULING

Carl A. Petri introduced PNs in 1962 for the study of "communication with automata". PNs, are mathematical and graphical tools. They provide uniform environment for modeling and analyzing of concurrent, asynchronous, distributed, parallel, and deterministic or stochastic systems. Recent theoretical researches and practical applications in PNs point out that PN based methods have been recognized as a promising tool for a wide variety of problems, including the scheduling of manufacturing systems. The purpose of this chapter is to give an overview of Petri nets and a review of the recent works on PN applications in production scheduling, and to discuss both theoretical developments and practical experiences.

2.1 Overview of Petri Nets

Formally, a Petri net (PN) can be defined as a 5-tuple, $PN = (P, T, A, W, M_0)$; where

$P = \{p_1, p_2, ...p_n\}$ is a finite set of places,

$T = \{t_1, t_2, ...t_q\}$ is a finite set of transitions,

$A \subseteq (P \text{ x } T) \cup (T \text{ x } P)$ is a finite set of arcs,

$W : A \rightarrow \{1,2,.....\}$ is the weight function attached to the arcs,

$M_0 : P \rightarrow \{0,1,2,.....\}$ is the initial marking.

$P \cap T = \varnothing$ and $P \cup T \neq \varnothing$.

A Petri net structure without a specific initial marking is denoted by N, and a PN with a given initial marking is denoted by (N, M_0). A given marking, M_i, indicates the current state of a PN, and it is denoted by M, an $n * 1$ vector, where n is the total

number of places. The p^{th} component of M, denoted by $M(p)$, is the number of tokens in place p. A marking changes when a transition "fires", i.e., when an event or activity occurs. Transition firing is equivalent to a state change and describes the PN's dynamic behavior. Considering the set of functions of input places (I) and of output places (O) for a transition t, as $I(t) = \{p \mid (p, t) \subset A\}$, and $O(t) = \{p \mid (t, p) \subset A\}$, respectively, a transition t is enabled in a marking M_i if and only if $M_i(p) \geq w(p, t)$ $\forall p \in I(t)$, where $w(p, t)$ is the weight of the arc from p to t. When a transition fires, the new marking is defined as the following;

$$M_{i+1} = \begin{cases} M_i - w(p, t) & \forall p \in I(t); \\ M_i + w(t, p) & \forall p \in O(t); \\ M_i & otherwise \end{cases}$$

A transition without any input place is called a source transition, and one without any output place is called a sink transition. A source transition is unconditionally enabled. A sink transition consumes tokens, but does not produce any. A source transition is often used, in a manufacturing system model, to represent the arrival of raw material or semi-finished parts in the system, while sink transitions are often used to represent the departure of finished or semi-finished parts.

A pair of a place p and a transition t is called a self-loop if p is both an input and output place of t. A PN is pure if it has no self-loops. A PN with self-loops can be converted to a pure PN by adding a place and transition to each self-loop in the original PN.

An infinite capacity net is a PN where each place can accommodate an unlimited number of tokens. A finite capacity net is a PN where each place, p, has an associated capacity, $K(p)$, meaning that p can hold a maximum of $K(p)$ tokens. A finite capacity net can be converted to an infinite capacity net using the complementary-place transformation rule (Murata, 1989).

The zero testing, i.e. the ability to test whether a place has no token, can be done with PNs by using an inhibitor arc in the model. In the presence of the inhibitor arc, a transition is regarded as enabled if each input place, connected to the transition by a normal arc, contains at least the number of tokens equal to the weight of the arc, and no tokens are present on each input place connected to the transition by the inhibitor arc. The transition firing rules are the same as for normally connected places. The firing, however, does not change the marking in the inhibitor arc connected places.

A PN as a graphical and mathematical tool possesses a number of properties, and these properties allow one to perform a formal check of the properties related to the underlying system's behavior. Some of the important properties are as follow (Zhou and Xiong, 2001):

- A PN is said to be *K*-bounded or simply bounded if the number of tokens in each place does not exceed a finite number *K* for any marking reachable from M_0. PN is said to be safe if it is 1-bounded. For a bounded PN, there are a limited number of markings reachable from the initial marking M_0 through firing various sequences of transitions.

- A PN is said to be live if, no matter what marking has been reached from M_0, it is possible to ultimately fire any transition of the net by progressing through some further firing sequence. This means that a live PN guarantees deadlock-free operation, no matter what firing sequence is chosen.

- A PN is said to be reversible if for each marking *M* in $\in R$ (Z, M_0), M_0 is reachable from *M*. Therefore in a reversible net one can always get back to the initial marking.

The boundedness, liveness, and reversibility of PNs have their significance in manufacturing systems. Boundedness or safeness implies the absence of capacity overflows. Liveness implies the absence of deadlocks. This property guarantees that

a system can successfully produce without being deadlocked. Reversibility implies cyclic behavior of a system and repetitive production in manufacturing. It means that the system can be initialized from any reachable state.

2.1.1 Analysis of Petri Nets

Methods of analysis for Petri nets may be classified into the following three groups:

1. The coverability (reachability) tree method,
2. The matrix-equation approach,
3. Reduction or decomposition techniques.

The first method involves essentially the enumeration of all reachable markings or their coverable markings. It should be able to apply to all classes of nets, but is limited to “small” nets due to the complexity of the state-space explosion. On the other hand, matrix equations and reduction techniques are powerful but many cases they are applicable only to special subclasses of Petri net or special situations.

Reachability Tree

Reachability analysis shows whether a system can reach a certain state. The reachability tree of a Petri net represents all the states and their relationships. Given a Petri net (N, M_0), from the initial marking M_0, we can obtain as many “new” markings as the number of the enabled transitions. From each new marking, we can again reach more markings. This process results in a tree representation of the markings (reachability tree). Nodes in a reachability tree represent markings generated from M_0 (the root) and its successors, and each arc represents a transition firing, which transforms one marking to another.

The above tree representation, however, will grow infinitely large if the net is unbounded. To keep the tree finite, a special symbol *w* is introduced, which can be thought of as "infinity". It has the properties that for each integer *n*, $w > n$, $w \pm n = w$ and $w \geq w$ (Murata, 1989). If a marking M* obtained at a given level is such that there exists a marking *M* on the path joining M_0 to *M** which verifies:

- $M^*(p) \geq M(p)$ for all the places of the PN,
- $M^*(p^*) > M(p^*)$ for at least one place p^* of the PN,

then the marking of p^* is denoted "*w*", where *w* stands for infinity. The marking of p^* will remain *w* in all the marking derived from *M** (Proth & Xie, 1996). This tree is a tool used to analyze the behavior of a Petri net model.

Some of the properties that can be studied by using the coverability tree *T* for a PN (N, M_0) are the following (Proth & Xie, 1996; Murata, 1989):

1) A Petri net (N, M_0) is bounded and thus $R(M_0)$ is finite if and only if *w* does not appear in any node labels in *T*, otherwise the net is bounded. Since it is important to be able to detect the inventories which may increase indefinitely, especially where an automated manufacturing system is concerned: reachability tree helps to prevent mistakes when designing the control (or management) systems.

2) A PN (N, M_0) is safe if and only if 0's and 1's appear in node labels in *T*.

3) A transition *t* is dead if and only if it does not appear as an arc label in *T*. This help in detecting the states of a manufacturing system which may prohibit some operations in the future.

4) If *M* is reachable from M_0 then there exists a node labeled M' such that $M \leq M'$. When the system is bounded, the reachability tree provides all the

markings reachable from the initial marking. From a manufacturing system point of view, the reachability tree provides all the states which can be reached from the initial state when the system is bounded.

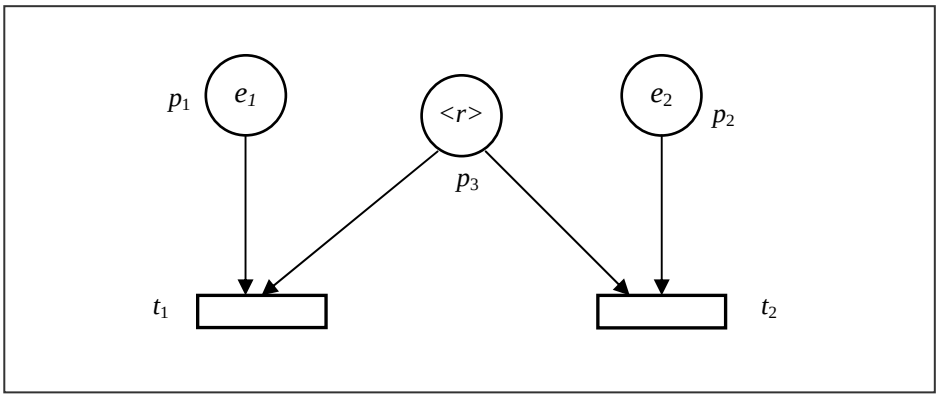
For a bounded PN, the coverability tree (reachability tree) contains, as nodes, all possible markings reachable from the initial marking M_0. In this case, it is called the reachability tree. For a reachability tree any analysis question can be solved by inspection. If it is possible to compute all reachable markings, $M \in R(N, M_0)$, and their reachability relationships, all qualitative behavioral properties should be analyzable.

2.1.2 Conflict Analysis

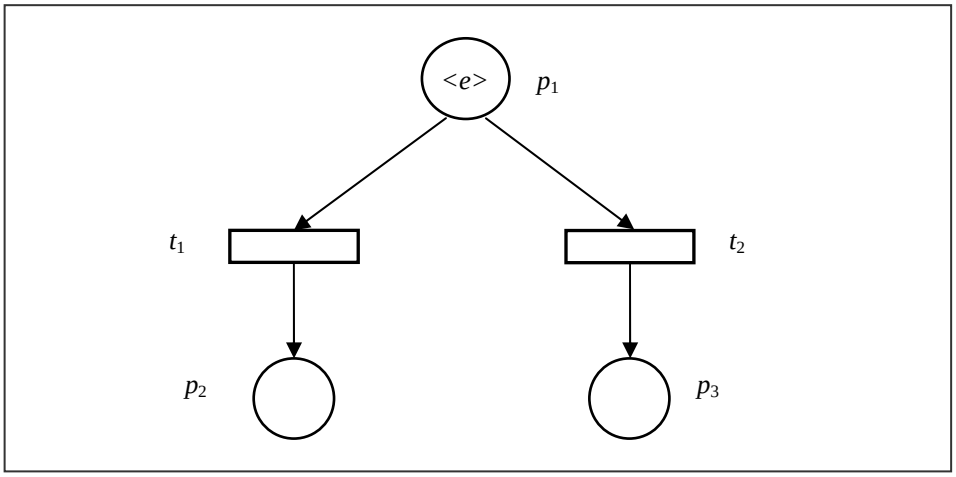
As Lin & Lee (1997) explain, a *conflict* results from two or more processes (or flows) competing for the same resources at the same time or from a process with several alternative routings. Conflicts can be analyzed directly from the PN model of a system. Conflicts in a PN model are classified into four types: *shared-resource*, *selectable-resource*, *alternative-activity* and *selectable-entity*. Figure 2.1 shows the main structures of the conflict situations in a PN model. A *shared-resource* (Figure 2.1 a) conflict refers to a resource (p_3) being shared by two or more processes at a time (t_1 and t_2). A *selectable resource* conflict (Figure 2.1 b) denotes that a process has several available resources (i.e. $<r_i>$, $i = 1,\ldots, m$) at any given time. An *alternative-activity* conflict (Figure 2.1 c) denotes that an entity in a process place (p_1) has two or more enabling transitions at a time (i.e. t_1 and t_2). Finally, a *selectable-entity* conflict (Figure 2.1 d) denotes that several entities in a place (i.e. $<e_i>$ in p_1, $j = 1,\ldots, n$) compete for the same resource (i.e. $<r>$ in p_3). Note that the four types of conflicts may occur in the same transitions simultaneously.

The presence of the conflict structures in a PN requires a conflict resolution mechanism to be introduced. Some conflicts have significant influence on the scheduling performance of flexible manufacturing system, and such conflicts are

referred to as effective conflicts. The others are called ineffective conflicts. The effective and ineffective conflicts can be classified by using the token simulation approach. For instance, selection of machine (i.e. part routings), selection of WIP pallets in the buffers and assignment of AGV or operator can be considered as some examples of effective conflicts in scheduling (Lin & Lee, 1997).

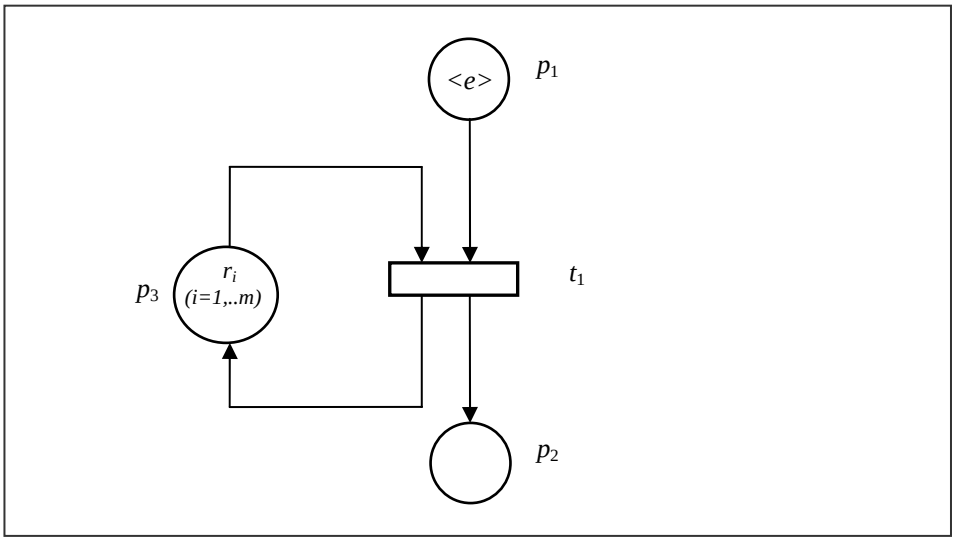


(a) Shared-resource conflict

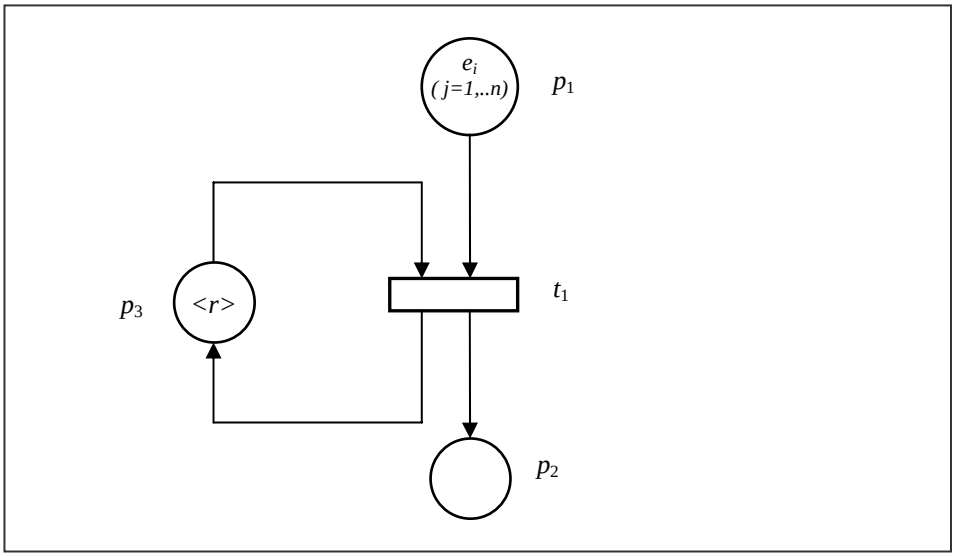


(b) *Alternative-activity conflict*

Figure 2.1 Classification of conflicts in a colored Petri net model (Lin & Lee, 1997)



(c) *Selectable-resource conflict*



(d) *Selectable-entity conflict*

Figure 2.1 (continued)

2.1.3 Extensions of Standard Petri Nets

A basic PN, as defined by its elements along with the enabling and firing rules, is powerful formalism. Nevertheless, to apply the tool to more realistic situations, some common extensions such as time, color and hierarchy are employed (Russo & Sasso, 2005).

2.1.3.1 Timed Petri Nets

The Petri net model of the logical and causal dependencies in a manufacturing system is not sufficient to answer time related questions, such as performance analysis of the system. A usual extension of standard Petri nets is to add timing concept into the model. The addition of time allows for such a temporal or qualitative performance analysis. “Time concept was introduced in the mid-1970s and early 1980s by Ramachandani (1974), Merlin (1976), Florin & Matkin (1982), Molloy (1982), and others” (Moore & Gupta, 1996). There are different ways to introduce time into the classical Petri net such as timed transitions PNs, timed places PNs, and timed places/transitions PNs. In a timed Petri net model is called a deterministic timed net if the delays are deterministically given or a stochastic net if the delays are probabilistically specified. The addition of timing is necessary if Petri nets are to be used to analyze properties such as system performance, cycle time, or scheduling behavior.

In timed Petri net models, marking condition alone (i.e. the token distribution) is not sufficient for describing the system, and the current time of the system has to be taken into account when analyzing the dynamic behavior of the system e.g. which transitions are enabled. In a transition-timed Petri nets, tokens are removed from the input places when a transition is fired, but new tokens are not released into the output places until a certain time has passed in the system. In a place-timed Petri nets, the firing of transitions are immediate, but a token introduced in an output place must wait for a specified time before it can be used as input for a transition. In other words, when a time is associated with a place, it represents the minimal amount of time tokens have to remain in the place when they arrive in this place as a result of a firing. Place-timed and transition-timed PNs are equivalent to each other with respect to modeling power.

Usually time is associated with transitions, since transitions represent operations and operations are time-consuming. Let the time associated with a transition is Q, and that the firing of t starts at time T_0. Then, firing t consists of:

(i) removing $W(p, t)$ tokens from each $p \in {}^0t$ at time T_0,
(ii) adding $W(t, p)$ tokens to any $p \in t^0$ at time $T_0 + Q$.

Between instants T_0 and $T_0 + Q$, it is supposed that tokens remain in the transition which fired. This represents a part remaining on a machine during the period in which an operation is performed.

The use of the deterministic timed Petri nets for the performance evaluation has been restricted to the choice-free or conflict-free nets, which can be modeled as marked graphs, or event graphs, as the presence of the conflict structures in a PN requires a conflict resolution mechanism typically based on a probabilistic function which cause the stochastic net (Zurawski & Zhou, 1994).

“Stochastic timed Petri nets (STPNs) were introduced independently by Florin & Natkin (1982), and Molloy (1982)” (Moore & Gupta, 1996). When time delays are modeled as random variables, or probabilistic distributions are added to the deterministic timed Petri net models for the conflict resolution, stochastic timed PN models are yielded. In such models, it has become a convention to associate time delays with the transitions only. When the random variables are of general distribution or both deterministic and random variables are involved, the resulting net models can not be solved analytically for general cases. Thus simulation or approximation methods are required. The stochastic timed PNs in which the time delay for each transition is assumed to be stochastic and exponentially distributed are called stochastic Petri nets (SPN). Some nets combine immediate transitions (that fire in zero time) with stochastic transitions to model situations in which some events occur instantaneously and others take a random amount of time. These are referred to as generalized stochastic Petri nets (GSPNs) (Marsan, Balbo, Conte, Donatelli, &

Franceshinis, 1995). Both models, including extensions such as priority transitions, inhibitor arcs, and probabilistic arcs can be converted into their equivalent Markov process representations, and analyzed analytically. When arbitrary distributions are allowed, simulation is often needed (Zhou & Venkatesh, 1998).

2.1.3.2 High-Level Petri Nets (HLPNs)

Although PNs provide a powerful formalism for modeling the dynamic behaviour of discrete concurrent systems, PN graphs of basic place/transitions models for the representation of complex systems are still very large and therefore usually become illegible and difficult to analyze. Information and its transmission in a net are also difficult to model. In order to construct more compact graphs that include information flow some extensions has been introduced to basic PNs (DiCesare, Harhalakis, Proth, Silva & Vernadat, 1993). One of the first extensions is to let individual tokens in a place be distinguishable by giving them "color". The input and output arcs of a transition are labeled with formal expressions, constraining the color combinations of tokens in input places that are allowed when firing the transition, and determining the color of the new tokens released to the output places. These kinds of nets are called Colored Petri nets.

But nowadays, the colors to differentiate the tokens each other become insufficient to model complex and large size systems, and a further extension to standard Petri nets leads to the so-called High-Level Petri Nets (HLPNs) (DiCesare *et al.*, 1993; Lin & Marinescu, 1988). In HLPNs, tokens are allowed to carry arbitrary complex data types (e.g. product data, operator information) and to let expressions for arcs and transitions be constructed with a special programming language. High-level PNs allow us to make much more compact models, and the complexity is hidden in the token attributes, functions, associated with arcs, and execution rules. However their analysis is presently not mature as for place/transition nets. Compared to basic Petri nets, high-level PNs are structured computer programming languages.

The main differences between the two types of high-level Petri nets and the basic Petri net models are (DiCesare *et al*., 1993);

- ✓ in the high-level Petri nets tokens have a type (also called color) and carry information to represent structured object and information flow, and

- ✓ formal expressions (also called inscriptions) constraining token occurrences, to be used as inputs or outputs of a transition, are attached to the arcs of the models. This information can be inspected and modified when a transition fires, thus imposing conditions on transition firing on the basis of token values.

The addition of data to tokens allows describing actions that the control system may execute and the way to label arcs with guard conditions, which is another characteristic of high-level Petri nets. By this way, a predicate can be attached to the transitions in addition to the enabling rule, and to fire a transition the guard condition that labels each incoming arc of a transition must also be satisfied by the data of one of the tokens in the connected input place. Only if at least one token can be found that satisfies this guard (predicate) condition in the input places of the transition, the transition will fire. While firing, a transition may also execute a function that modifies data in the tokens being manipulated.

The conciseness of the resulting model which is compensated by more complex inscriptions attached to the arcs of the net makes it possible to fold several basic similar subnets into a single, more concise net. Thus Nets can also be hierarchically organized using so-called *hierarchical PNs*. Firing of a transition in a hierarchical Petri net may be governed by the execution of a self-contained subordinate Petri net. As the subordinate Petri net completes execution, the transition in the higher level Petri net completes its firing process. Similarly, a single place in higher level Petri nets may also subordinate net may also represent a complete subordinate Petri net. In this case the subordinate net must have a single start place and a single end place. A

token entering the place in the higher level Petri net is equivalent to the token entering the start place of the subordinate net. When the token reaches the end place of the subordinate, it becomes available in the higher level net to enable outgoing transitions (Russo & Sasso, 2005).

A formal definition of a generalized stochastic high-level PN is as follows (Koriem & Patnaik, 1997):

GSHLPN = $\{P, T, A, H, Mo, R, S\}$

where

P is the set of places; $P = \{p_1, p_2, \ldots\ldots, p_n\}$,

T is the set of transitions; $T = \{t_1, t_2, \ldots\ldots, t_m\}$,

A is the set of arcs; $A \subseteq (P \text{ x } T) \cup (T \text{ x } P)$,

H is the set of inhibitor arcs; $H \subset P \text{ x } T, A \cap H = \varnothing$,

R is the set of firing rates associated with TPTs; $R = \{r_1, r_2, \ldots\ldots r_m\}$,

M_0 is the initial marking M0 = $\{n_{at}(p_1), n_{at}(p2), \ldots\ldots n_{at}(p_n)\}$,

where

$n_{at}(p_i)$ denotes the number of n-attributes of individual tokens in place $p_i \in P$,

S is the structure set defined as follows;

$S = \{S_{x1}, S_{x2}, \ldots\ldots, S_{xi}\ ; S_{y1}, S_{y2}, \ldots\ldots, S_{yj}\ ; S_{z1}, S_{z2}, \ldots\ldots, S_{zk}\}$

where

$\{S_{x1}, \ldots\ldots S_{xi}\}$ is a set of individual tokens called the domain of S,

$\{S_{y1}, \ldots\ldots S_{yj}\}$ are functions (i.e., operations) in S

$\{S_{z1}, \ldots\ldots S_{zk}\}$ are relations in S.

Firing rules:

The detailed firing rules for high level PNs are as follows:

1. Each element of *T* represents a class of possible changes of markings. Such a change is also called transition firing. This transition firing consists of removing tokens from a subset of places and adding them to other subsets according to the expressions labeling the arcs.

2. A transition is enabled whenever
   - individual tokens are assigned to the variables that satisfy the predicate associated with the transition;
   - all input places contain enough copies of proper tokens; and
   - the capacity *K* of all output places does not exceed by adding to the respective copies of tokens.

   The reachability set of HLPN model consists of the set of all markings connected to the initial marking through such occurrence of firing.

3. Once a transition is enabled, it fires with a probability depending on whether the transition is timed or immediate.

A HLPN system design provides the following benefits (Tricas & Martinez, 1998):

- A HLPN are a formal specification language, so ambiguities are avoided.
- A graphical representation that makes easier the understanding of the specification is provided.
- The analysis theory makes possible the error-detection on design phase.
- The model representation obtained is independent from the implementation.

2.2 A Review on Applications of Petri Nets in Production Scheduling

Because of the stiff competition in today's marketplace, production systems have to respond to changes quickly by integrated planning, scheduling and control. Scheduling is the process of selection among alternative plans and assigning resources and times to the set of activities in the selected plan so that the constraints of the system are met and the performance criteria are optimized. In a manufacturing system, resources represent machines, operators, robots, tools and buffers, and activities symbolize the processing of products on machines or the transport of products between machines. A manufacturing scheduling can be viewed as having as its goal the optimization of one or more objectives. Since the computation time required to obtain the global optimal schedule grows exponentially with the problem size, scheduling problems are known to be very complex and NP-hard for general cases. Mathematical programming techniques, neighborhood search methods, heuristic dispatching methods, and AI based techniques are the main approaches used for the solution of scheduling problems (Baker, 1974; Morton & Pentico, 1993; Pinedo, 1995; Jones & Rabelo, 1998). Nevertheless, mathematical programming approaches has some limitations over complex systems, since it is difficult to mathematically describe some practical constraints of complex scheduling systems related to material handling system, complex resource sharing situations, and routing flexibility. Most of neighborhood search methods still require considerable computation and rather long running time for practical applications, and lack to handle all the constraints of today's real-world production scheduling problems. On the other hand, heuristic dispatching rules have often limited information about the entire system, and may be inefficient for scheduling of complex manufacturing systems. AI based techniques generate feasible solutions, and it is rarely possible to tell how the solution obtained is close to the optimal solution.

Recently, scheduling based on PNs, which are discrete event based methods has gained a growing interest of researchers as they directly describe the actual dynamic

behavior and the control logic of the systems. They are easy to develop and to extend. Researchers have a better understanding of the dynamic behavior of the system by means of tokens and transitions. Activities, resources, and constraints of a manufacturing system can be represented in a single coherent formulation. Concurrent and asynchronous activities, multilayer resource sharing, routing flexibility, limited buffers, and precedence constraints can be explicitly and concisely modelled by PNs (for further reading see Desrochers & Al-Jaar, 1995; Hack, 1972; Moore & Gupta, 1996; Murata, 1989; Proth & Xie, 1996; Zhou & Venkatesh, 1998; Zurawski & Zhou, 1994). Timed PNs give a state-space structure representation of possible schedules of a manufacturing system, and this structure is transformed to a scheduling problem which can be solved by combining PNs execution with other techniques such as branch-and-bound, heuristic search, artificial intelligence search, heuristic dispatching, and expert systems.

Recently, scheduling based on Petri nets (PNs) has gained a growing interest of researchers as they directly describe the actual dynamic behavior and the control logic of the systems. Through the following section we review the literature of scheduling based on PNs by considering the techniques combined with PNs execution, and give some conclusions and future research directions in the last section (Tuncel & Bayhan, 2003). We give a review of PNs research in 20 major production and operations management journals from 1989-2005, discuss both theoretical developments and practical experiences, and identify research trends and publication sources of applications.

2.2.1 PNs with a Heuristic Rule-based System

Early use of PNs in scheduling applications began in the late 1980's. In one of the first studies, Hatano, Yamagata, & Tamura (1991) employed stochastic Petri nets (SPNs) to describe the uncertain events in FMSs, such as failure of machine tools, repair time, and processing time. They developed a rule-based on-line scheduling system that generates appropriate priority rules to select a transition to be fired from conflicting transitions, but could not handle the routing flexibility and deadlock

situations. Performance of the rule base was evaluated on an FMS simulation model built by authors.

Raju & Chetty (1993) proposed a new class of PNs called *Priority Nets* for modeling and simulation of FMSs to impart dynamic decision-making capability and flexibility. The authors described a four level hierarchical structure of decision making for dynamic scheduling. The levels of this structure include loading the parts into the system, the selection of machine tools and the jobs, and dispatching of the transport devices. Some decision rules are handled with priority net aided simulation to determine the operating policies under varying conditions of FMS operation.

In another study, Yim & Linn (1993) used PN-based simulation to investigate the effect of different AGV dispatching rules on the performance of an FMS. The proposed model integrates the PN model with the AGV dispatcher that controls AGV tokens in the PN model. Performances of push based and pull based AGV dispatching rules were investigated when the FMS was set in a busy state. This study would be extended for non-busy shop case in order to see the behavior of the dispatching rules.

The study of Camurri, Franchi, Gandolfo, & Zaccaria (1993) deals with automatically creating a PN model from the knowledge base of an FMS. Colored transition-timed PNs were used in the modeling of FMS. The authors presented a general purpose scheduling strategy where a random selection is used to resolve the conflicts in each trial, then the best solution found is chosen. They also proposed a special purpose scheduler based on priority rules that can be used in real-time.

Hu, Wong, & Loh (1995) proposed a decision support system for scheduling and control of an FMS. *Generalised Stochastic PNs* were used for modeling the system, and a set of priority rules is associated with the conflicting transitions to solve the resource allocation and job sequencing problems in real-time basis. The authors point out the requirement of the user-friendly interactive decision support system that can handle the large scale scheduling problems.

In dynamic scheduling problems, integration of PNs execution with a rule-based scheduling system has been stimulated by the desire of obtaining an acceptable solution in a shorter time. Chincholkar & Chetty (1996) presented a heuristic rule base for the dynamic scheduling of an FMS. The performance criterion was makespan minimization. The authors used Stochastic Colored Petri Nets (SCPNs) to model the system. The performance of the proposed algorithm for two FMSs with a fixed and a flexible routing was compared with those of several priority rules such as Shortest Total Processing Time, First-come First-served, Shortest Imminent Processing Time, Largest Number of Remaining Processes, and Smallest Ratio of Imminent Processing Time and Total Processing Time. The proposed rule base performed satisfactorily when part routing was fixed, and gave better results for limited cases when alternative machines were included.

In the following year, Lin & Lee (1997) presented an integrated hierarchical control and scheduling scheme for flexible manufacturing cells (FMCs). In this approach, scheduling of cells is performed by combining dispatching rules with the Colored PN (CPN) simulation. The CPN model is also used to represent the control logic of the designed flows. Easy implementation of the rescheduling process is one of the advantages of a common model usage.

Since ordinary PNs are limited for modeling complex nature of production systems, they have been extended to high-level PNs: colored PNs, Evaluation PNs (E-nets), and predicate/transition PNs. Yan, Wang, Cui, & Zhang (1997) and Yan, Wang, Zhang, & Cui (1998) introduced and extended high level evaluation PNs (EHLEP-N) and extended high level evaluation stochastic PNs (EHLESP-N), respectively, by combining features of the evaluation nets (E-Nets), the predicate/transition PNs, and the colored PNs for modeling and control of FMSs. In these studies, a rule-based expert system is described for real-time scheduling.

Jain (2001) introduced a new concept of P-levels for the solution of resource contention and circular wait by assigning priorities to the parts waiting for a common

resource on a real-time basis. The author used stochastic PNs to model an example FMS, and proposed a priority rule-based system controller to resolve the resource contention problems. The developed methodology extends the capabilities of the system controller by enabling it to take into account the priorities assigned to all other waiting parts for a common resource before making a final decision about the resource assignment.

2.2.2 PNs with a Search Algorithm

Generative scheduling methods have been paid more attention than simulation models with heuristic dispatching rules which were generally developed for particular applications. By these methods, after the PN model of a system is constructed, a scheduling algorithm is used to expand the Reachability Graph from the initial marking to the final or goal marking. Theoretically, an optimal schedule can be obtained by generating the entire reachability graph of the PN model, and then finding the optimal path from the initial marking to the final marking. But due to exponential growth of the number of states, generating of the entire reachability graph is very difficult even for a simple PN model of a small size system. Therefore, to find near optimal schedules, researchers try to reduce this state space by using search algorithms including heuristic functions, and thus generate only a portion of the reachability graph. In one of the first studies, which apply Petri net theory for solving scheduling problems, Shih & Sekiguchi (1991) presented a transition timed PN and beam search technique for an FMS. When a scheduling conflict has to be solved, in other words, if there are two or more conflicting transitions that are enabled, the scheduling system calls for a beam search decision module that constructs partial schedules within the beam depth, and then selects the best one. The cycle is repeated until a complete schedule is obtained. Although the method doesn't guarantee the global optimization, good results were obtained.

Lee & DiCesare’s study (1994a) is also one of the first studies combining an intelligent global search and PN execution. The authors developed a heuristic search algorithm named L1 algorithm based on the A* graph search algorithm, and used

timed-place PNs. L1 algorithm generates and searches only the necessary portion of the PN's reachability graph to find optimal or near optimal feasible schedules. To limit the search space, four heuristic functions that differ in the choice of evaluation functions for the nodes are used. The first function considers the depth of the marking and favors the marking that is deeper in the reachability graph to reach the final marking. The second one estimates the minimum remaining operation time and encourages a marking having an operation ending soon. The third one is a combination of the first two functions, and the last heuristic function compensates the cost of the considered PN marking by the weight depth of the marking. These functions do not guarantee the admissible condition that has to be satisfied if optimality is to be guaranteed during the search. However, the proposed algorithm reduces the search space and can be used for large size scheduling problems. Lee & DiCesare (1994b) extended their previous work and examined two AGV scheduling policies by using the L1 algorithm proposed in their previous work. They concluded that, the proposed rules give better results than Shortest Queue Length and Shortest Processing Time rules.

In another study, Sun, Cheng, & Fu (1994) developed a timed-place PN model for an FMS. This study also includes the control of a multiple AGV system. For further reducing the average search time of A* algorithm, the authors employed *the Limited-Expansion A algorithm* based on modified A* heuristic search algorithm, and used the fourth evaluation function in Lee & DiCesare (1994a). The algorithm is similar to a global beam search method since it sets a maximum value for the number of unexplored nodes that are kept in memory. The limited Expansion Algorithm has lower complexity and reduces memory requirement.

Although Petri nets are well suited for understanding and formulating scheduling problems, they tend to become very large even for a moderate-size system (Murata, 1989). The analysis of a large PN requires a high computation effort and memory requirement, since the search space grows rapidly. The resulting complexity problem is handled by some truncation or decomposition techniques. In the study of Chen, Luh, & Shen (1994), a truncation technique, which divides the original PN into

several smaller subnets, was employed in order to reduce the complexity of combining the execution of a large size timed PN with the modified branch-and-bound technique. The authors developed algorithms that can be used to search for a proper schedule.

Zhou & Xiong (1995) presented PN based branch-and-bound method for solving the scheduling problem of FMSs. In case of the conflicting jobs, the authors employed heuristic dispatching rules such as shortest processing time (SPT) to select one transition from the candidate sets. The generated schedule is used to transform PN model of the system into the Timed Marked Graph, a special class of timed PNs, then performance analysis of the system is performed by means of the properties of these graphs.

In the next study, Xiong, Zhou, & Caudill (1996) proposed a hybrid heuristic search strategy combining the heuristic best-first strategy, the *A* graph search algorithm* similar in Lee & DiCesare (1994a), with the controlled backtracking strategy. The aim of the study is to reduce memory requirement and the computation time of the search process of the scheduling for minimizing makespan. In the proposed method, the search process begins with best-first strategy until a depth-bound is reached in the search tree, then it continues with the controlled back tracking search using the best marking in the current state as a starting node.

For makespan minimization, Chetty & Gnanasekaran (1996) proposed a Colored PN based scheduling approach for flexible assembly systems (FASs). In this study, precedence diagrams are used to define the sequences of the operations, which are performed on different products, and a controlled search algorithm is employed to identify near optimal solutions by varying urgency factor and due date parameters.

Recently, there has been a growing interest in merging PNs and object-oriented approaches to combine graphical representation and mathematical foundation of PNs with the abstraction, encapsulation, and inheritance features of object-orientation (Chen & Chen, 2003; Venkatesh & Zhou, 1998; Wang, 1996; Wang & Xie, 1996).

Wang (1996) presents an object-oriented PN (OOPN) paradigm that incorporates the scheduling/ dispatching knowledge in the control logic. In the following study, Wang & Wu (1998) extend OOPNs by adding colored tokens (Colored OOPN) for modeling and analyzing an automated manufacturing system, and then apply modified L1 search algorithm proposed in Lee & DiCesare (1994a) to generate near-optimal schedule. This study also includes a deadlock detection algorithm in order to detect and avoid all possible deadlock situations.

In Xiong & Zhou's study (1998), hybrid heuristic search strategy *Best First -Back Tracking* (BF-BT) is compared with another hybrid strategy *Back Tracking-Best First* (BT-BF) in a semiconductor test facility with multiple lot sizes for each job type. Scheduling results show that the BT-BF strategy performs better than the BF-BT strategy.

For scheduling of FMSs with the objective of makespan minimization, Jeng & Chen (1998) proposed a heuristic search approach based on analytic theory of the PN state equations. They used an approximate solution to an integer-programming problem with the constraints defined by the PN, which incorporate the sufficiently global information into the evaluation function. The authors report better scheduling results in shorter computational time and less search state space than those reported by Lee & DiCesare (1994a). However, both formulating and solving the PNs state equations can be quite complex for large size systems. On the other hand, as the proposed approach uses state equations based on the incidence matrix of PNs, this approach can not be used with the Colored PNs.

In the following year, Jeng, Lin, & Huang (1999) extended the previous work presented by Jeng & Chen (1998), and proposed a new heuristic search method. This method is more effective in different system configurations including generalized symmetric nets (GSNs) and generalized asymmetric nets (GANs). In light of the simulation results, the authors concluded that the proposed heuristic gives better results than their previous method and the method proposed by Lee & DiCesare (1994a). However, the implementation of the proposed heuristic search method may

still require extensive computational effort for large system since formulating and solving the PN state equations is quite complex.

A new class of high-level timed PNs named *Chameleon Systems* was introduced by Kis, Kiritsis, Xirouchakis, & Neuendorf (2000). In this study, a simple scheduling problem with 3 jobs and 2 machines was modeled by *Chameleon Systems* and then analyzed for makespan minimization. The greedy heuristic was used for scheduling. *Chameleon Systems* provide a modular construction, and all classical known PN analysis methods can be performed using the unfolding of the *Chameleon systems* to its corresponding interval time PN. In spite of these advantages, the direct use of the proposed approach may still not practical to obtain optimal schedules in real-world manufacturing systems, since it has difficulties to deal with large and complex systems.

In recent years, disassembly planning and demanufacturing has gained a growing importance due to increasing economic and environmental pressures. Tang, Zhou, & Caudill (2001) introduced three extended PN models for disassembly planning and demanufacturing scheduling for an integrated flexible demanufacturing system. These extended PN models are *Product PN* including all feasible disassembly sequences and the *EOL* (end-of-life) values; a *Workstation PN* to model the status of workstations; and a *Scheduling PN* for machine scheduling. In *Scheduling PNs*, processing and delay time functions assigned to transitions are used to deal with machine assignment in each workstation. The proposed methodology deals with the problem of maximizing *EOL* value of products and system's throughput.

A PN based approach, which automatically generates a disassembly PN model from the geometrically-based disassembly precedence matrix of the product, was presented by Moore, Gungor, & Gupta (2001) for product recycling and remanufacturing. A reduced reachability tree algorithm which alternates a limited depth-first-search with a branch and bound is used to generate feasible disassembly process plans. The cost function of the heuristic employed in the algorithm incorporates tool changes, changes in direction of movement, and individual part

characteristics. The proposed approach can be used for products containing AND/OR disassembly precedence relationships.

As a result of customer driven environment of today's manufacturing systems and the competitive pressure of the global economy, the growing importance of flexibility and quick response of manufacturing systems to changes force to develop new managerial approaches and to search new production environments. This issue is addressed by Jiang, Liu, & Zhao (2000) who proposed Virtual Production Systems (VPSs) for enhancing the agility of manufacturing systems. These systems need a new dynamic scheduling method, which could handle any change and disturbance efficiently. Fung, Jiang, Zuo, & Tu (2002) proposed an adaptive production scheduling method to minimize makespan for virtual production systems, and employed modified A* algorithm to obtain optimal or near optimal schedule. The proposed approach is different from the conventional real-time scheduling methods that update schedules by only reallocating the tasks over the existing resources and routings in the system. In this study, Object-Oriented PNs with changeable structure are used to formulate scheduling problem of VPSs. The presented method allows allocating tasks to existing resources and possibly to other resources newly added into the VPS subject to limited resources.

An FMS consists of resources with limited capacities, and in this system different types of products are produced concurrently according to the process plans that may include alternative routing policies. However, the limited resource capacities can lead to deadlocks during resource allocation. For deadlock-free scheduling, Abdallah, ElMaraghy, & ElMekkawy (2002) developed an algorithm for a class of FMSs called *Systems of Sequential Systems with Shared Resources (S^4R).* The aim of the study was to minimize the mean flow time. The authors used a search algorithm based on the branch-and-bound and the depth-first-search strategy with a siphon truncation technique to obtain efficient feasible solutions, and to reduce the search effort for large scheduling problems. This study points out that PNs are more suitable to dealing with the deadlock-free scheduling problem than the mathematical

programming approach, as the deadlock states are explicitly defined in the PN framework and no equation is needed to describe the deadlock avoidance constraints.

In another study which combine PN modeling and AI heuristic search for scheduling FMSs, Moro, Yu, & Kelleher (2002) proposed a hybrid PN based scheduling algorithm called a *Dynamic Limited-Selection Limited-Backtracking Algorithm (DLSS*).* This algorithm employs PN-based dynamic local stage search A*, and a branching scheme for *DLSS** called *Controlled Generator of Successors* that avoids the generation of both schedule permutations caused by concurrent transitions and certain inactive schedules. The aim is to reduce the search effort while maximizing admissibility. The authors presented a comparison with some previous works in Lee & DiCesare (1994a) and Xiong & Zhou (1998) to show the superiority of their approach.

In one of the recent studies, Yu, Reyes, Cang, & Lloyd (2003) proposed a new modelling and scheduling approach for FMSs using PNs and AI based heuristic search methods to reduce the scope of the A* algorithm and to enhance the power of the heuristic function by using PN properties. A new class of PNs called *Buffer-nets (B-nets)* was introduced, and a new heuristic function was derived from the concept of resource cost reachability matrix built on the properties of *B-nets*. This heuristic function attempts to give a theoretical lower bound for minimum makespan among the several states of an FMS. Although the proposed approach provides promising improvements on reducing the search effort, it assumes that some of the scheduling problem constraints are relaxed (i.e. there is no conflict among the users of shared resources). This means that the number of resources is infinite, and the problem is only constrained by the operation sequences among jobs.

Although most of the scheduling studies using PNs focus on optimizing the makespan, Elmekkawy & Elmaraghy (2003) proposed three heuristic functions for deadlock-free FMS scheduling to minimize the mean-flow time. These heuristic functions are: Average Operation Waiting Times (AOWT), Remaining Processing Time (RPT), and Shortest Processing Time (SPT) dispatching rule. The aim is to

reduce the complexity of scheduling problem of large-size systems. The RPT heuristic, which is a summation of the minimum remaining processing time of the places that have a token under the marking and the negative value of the marking depth in the Reachability Graph, has some similarities with the third heuristic function proposed in Lee and DiCesare's study (1994a). The authors used best-first search technique with backtracking and average flow time criterion, whereas L1 Algorithm based on A* search algorithm and makespan minimization criterion were employed in Lee and DiCesare (1994a). The experimental results show that the AOWT outperforms the other two functions with respect to average flow time and CPU time.

Korbaa, Benasser, & Yim (2003) developed two different scheduling methods for FMSs. The performance criterion is makespan minimization. The first method tends to solve the generic scheduling problem (a cyclic scheduling problem) using constraint programming to avoid exhaustive search. The second one is dedicated to cyclic scheduling problem. The authors show the advantage and disadvantage of the proposed methods by using some illustrative examples.

Lee & Korbaa's study (2004) deals with product ratio-driven FMS cyclic scheduling problem with minimization of the cycle time and work-in-process inventory using timed PNs. The authors used unfolding PNs to analyze the sequencing process, and to avoid the state space explosion. An *unfolding* is obtained by unfolding a PN that has the reachability information and properties of the original model. Thus structural analysis on *unfolding* becomes much easier than on the original model. An algorithm is developed to solve resource sharing conflicts using the transitive matrix based on the behavioral properties of the net. The proposed approach was applied to a system with 2 machines and 2 jobs.

In a recent study, Ghaeli, Bahri, Lee, & Gu (2005) also presented a PN based approach for the short-term scheduling of simple batch plant. The authors formulated the scheduling problem by using a timed PN model with 9 transitions and 14 places, and used A* graph search algorithm that generates and checks the markings in the

reachability tree of the model like in some previous studies such as Lee and DiCesare (1994a), Sun, Cheng, & Fu (1994), and Xiong, Zhou, & Caudill (1996). The implementation results were compared with the results obtained by traditional mixed integer linear programming (MILP) and difficulties in formulating the scheduling problem in batch plants with MILP was also outlined by the authors.

2.2.3 PNs with Mathematical Programming Approaches

PNs have been also combined with mathematical techniques for solving scheduling problems. In one of these studies, Hillion & Proth (1989) presented a scheduling approach based on timed Event Graphs which is a special class of PNs for the job-shop systems with repetitive demands. The authors used the properties of the Event Graphs to evaluate the system performance in steady state. Firstly, the elementary circuits (i.e. processing circuits, command circuits and mixed circuits) of the system were determined, then a 0-1 linear programming problem for minimizing the cycle time of the system while minimizing the number of the tokens in the model was described. Since the number of circuits increases very fast with the problem size, a heuristic algorithm developed in Hillion & Proth (1989) was used to solve the described problem. Finally, the optimal control problem was addressed in order to obtain a near optimal scheduling of a job-shop using the heuristic algorithm given in Hillion, Proth, & Xie (1987) which leads to a good firing schedule of the transitions.

In the following study, Proth & Sauer (1998) address a two-stage scheduling algorithm based on a subclass of PNs called Controllable Output net (CO nets). In this study, CO nets are used to model special case of manufacturing systems where the production flows of the different types of products can be considered as being constant on the scheduling horizon. First stage of the algorithm includes a mathematical programming approach in order to distribute the workload among the resources, and the second stage derives a schedule in the net. The goal is to maximize the productivity and to meet the required rates while minimizing the Work-In-Process inventory.

In the same year, Song & Lee (1998) transformed the cyclic job shop with finite buffers into a cyclic job shop with no buffer using the Timed Marked Graphs, and presented a mixed integer-programming model for the cyclic job-shop with no buffer to find a deadlock-free optimal schedule. The objective is to minimize the cycle time, and the constraints are the precedence relationships, blocking and deadlock free conditions.

2.2.4 PNs with Meta Heuristics

In recent years, modeling capabilities of PNs were also combined with meta-heuristics for solving complex scheduling problems. A decomposition methodology proposed by He, Strege, Tolle, & Kusiak (2000) solves the complexity problem in modeling and scheduling of manufacturing systems. By this methodology, manufacturing system is decomposed into modules considering the similarity of resources, and then sub PN models are aggregated to obtain a hierarchical PN model. The authors developed a sequential cluster identification algorithm to decompose a manufacturing system as an Integrated Definition 3 (IDEF3) model, which can describe the logical relationships among the activities Larson & Kusiak (1996). In this algorithm, firstly, the partial firing sequence of each sub PN models are determined by using the scheduling system that is a combination of a simulated annealing scheduling algorithm and a deadlock recovery procedure based on PNs. The deadlock recovery procedure combines a deterministic backtracking and a Monte Carlo forward tracking steps. After the partial firing sequences for each sub PN were determined, sub nets are integrated into an aggregated schedule. For this purpose, a heuristic rule-based simulation is performed to determine the transition firing sequences of shared resources. Authors illustrated the proposed methodology with a flexible disassembly cell. The computational results showed that the developed methodology reduces the complexity and computational time without a significant change in its quality.

Recently, much attention has been given in shop-floor scheduling of wafer fabrication area owing to capital intensive and complex nature of semiconductor

manufacturing Zhou & Jeng (1998). Due to the high complexity of wafer fabrication, a descriptive representation is required to solve its scheduling problem. Here, PNs can capture the complex process flows in wafer fabrication by means of their modeling capabilities and formulating advantages. Chen, Fu, Lin, & Huang (2001) proposed a systematic colored PN model for a wafer fabrication, and used a genetic algorithm based scheduler that dynamically searches for the appropriate dispatching rules for each machine group or a processing unit. The authors used priority rule-based representation of chromosomes in the genetic algorithm. The implementation results show that the proposed GA scheduler approach performs better than the rules First-Come First-Serve, Earliest Due Date, Smallest Remaining Processing Time, and Minimum Inventory at the Next Station first.

The scheduling problem of wafer fabrication was also investigated by Jain, Swarnkar, & Tiwari (2003). The authors used Generalized Stochastic Petri Nets for modeling of a wafer fabrication system, and a simulated annealing based scheduling methodology with mean cycle time and tardiness criteria is proposed to obtain the efficient schedules on a real-time basis.

A new approach based on Neuro-Expert Petri net (NEPN) model was introduced by Kumar, Tiwari, & Allada (2004) for modeling and rescheduling of a wafer fabrication line involving machine unreliability. The failure and repair rates are estimated by decomposing the NEPN model of the re-entrant semiconductor wafer fabrication line to aid the rescheduling phase of the production system. The computational results revealed that the proposed method performs relatively well for makespan criteria.

Since resource allocations may lead to system deadlock situations in discrete event systems, much research has been devoted to the deadlock-free scheduling. As mentioned before, PNs are well suited to describe dynamic feature of the discrete event systems such as concurrence, resource sharing, conflicts and deadlock situations. Gang & Zhiming (2004) proposed a deadlock free scheduling approach which combines the search method of the genetic algorithm with the formal

reachability graph method of the PN so that deadlocks can be avoided in advance. The authors used a bottom-up approach to model the whole system. The proposed approach does not take into consideration the material handling system constraint for the simplification of the analytic process.

Cavory, Dupas, & Goncalves (2005) proposed an approach to the resolution of cyclic job-shop scheduling problem with linear constraints. The authors used PNs for the modeling of linear precedence constraints between cyclic tasks, and developed a conflict resolution algorithm based on the coupling of a genetic algorithm. The implementation results point out that the proposed approach overcomes the Random Search strategy.

Table 2.1 classifies the papers reviewed above by considering the techniques combined with PNs execution for scheduling of production systems.

Table 2.1 Scheduling applications in production systems

(with heuristic rule based systems)			
Year	**Author(s)**	**Scheduling Approach**	**Application Area**
1991	Hatano *et al.*	Stochastic PNs and a heuristic rule based system	FMS
1993	Raju & Chetty	Priority Nets with a rule-based system	FMS
1993	Yim & Linn	Colored PNs with inhibited arcs and capacitated timed places and AGV dispatcher	AGV system in an FMS
1993	Camurri *et al.*	Colored transition-timed PNs and Priority rules	FMS
1995	Hu *et al.*	Generalised Stochastic PNs and Priority rules	FMS
1996	Chincholkar & Chetty	Stochastic Colored Petri Nets with a heuristic rule-based system	FMS
1996	Wang	Object Oriented PNs and Scheduling / dispatching knowledge system	FMS
1997	Lin & Lee	Colored PNs with dispatching rules	FMCs
1997	Yan *et al.*	Extended High Level Evaluation PNs (EHLEP-N) and A Rule-Based Expert System	FMS
1998	Yan *et al.*	Extended Stochastic High Level Evaluation PNs (ESHLEP-N) and A Rule-Based Expert System	FMS
2001	Jain	Stochastic PNs and a rule-based approach with *P-Levels*	FMS

(Table 2.1 is continued)

(with search algorithms)			
Year	**Author(s)**	**Scheduling Approach**	**Application Area**
1991	Shih & Sekiguchi	Transition-timed PNs and beam search approach	FMS
1994	Lee & DiCesare (a)	Timed–place PNs and *L1* Algorithm: *(A heuristic search approach based on the A* graph search algorithm with three heuristic functions)*	FMS
1994	Lee & DiCesare (b)	Timed –place PNs and *L1* Algorithm	FMS with two different AGV systems
1994	Sun *et al.*	Timed –place PNs and Limited-Expansion *A* algorithm	FMS that includes a multiple AGVs
1994	Chen *et al.*	PNs with a truncation technique and modified branch-and-bound technique	FMS
1995	Zhou & Xiong	PNs and branch-and-bound method with heuristic dispatching rules for scheduling and Timed Marked Graphs for cycle time analysis	FMS
1996	Xiong *et al.*	Timed–place PNs and a hybrid heuristic search methodology combining best-first and controlled backtracking strategy	FMS
1996	Chetty & Gnanasekaran	Coloured PNs and a controlled search algorithm	FAS

(Table 2.1 is continued)

1998	Wang & Wu	Colored Object Oriented PNs and modified *L1* algorithm described in Lee&DiCesare (1994)	FMS
1998	Xiong & Zhou	Timed-Place PNs with two hybrid strategies: *Best First -Back Tracking* (BF-BT) and *Back Tracking-Best First* (BT-BF)	Semiconductor test facility
1998	Jeng & Chen	Timed-place PNs and a heuristic search approach based on analytic theory of the PN state equations	FMS
1999	Jeng *et al.*	Timed-place PNs and a heuristic search approach based on analytic theory of the PN state equations which is more effective in different system configurations	FMS
2000	Kis *et al.*	A high-level object PNs (*Chameleon Systems*) and Greedy heuristic approach	Job-Shop (with 3-jobs and 2-machines)
2001	Tang *et al.*	Product PNs, Workstation PNs, Scheduling PNs and a heuristic search approach	Integrated Flexible Demanufacturing System
2001	Moore *et al.*	Petri nets and Reduced Reachability tree method	Disassembly process planning
2002	Fung *et al.*	Object-oriented PNs with changeable structure and a modified *A** search algorithm	Virtual production system

(Table 2.1 is continued)

2002	Abdallah *et al.*	PNs and a search algorithm based on the branch-and-bound and the depth-first-search strategy with a siphon truncation technique	A class of FMS called S^4R
2002	Moro *et al.*	PNs and a hybrid algorithm *(Dynamic Limited-Selection Limited Backtracking algorithm*)	FMS
2003	Yu *et al.*	*Buffer nets*, a class of PNs, and a heuristic search approach based on Resource Cost Reachability Matrix	FMS
2003	Elmekkawy & Elmaraghy	Petri nets and best-first search technique with backtracking	FMS
2003	Korbaa *et al.*	Petri nets and Constraint programming method	FMS (cyclic and acyclic scheduling)
2004	Lee & Korbaa	Unfolding PNs and an algorithm based on the transitive matrix	FMS cyclic scheduling problem
2005	Ghaeli *et al.*	Timed Place PNs and *A** search algorithm	A simple batch plant

(with mathematical approaches)

Year	**Author(s)**	**Scheduling Approach**	**Application Area**
1989	Hillion & Proth	Timed Event Graphs, 0-1 Linear programming and a heuristic algorithm	JSS
1998	Proth & Sauer	Controllable Output nets (CO- nets) and a mathematical programming approach	JSS

(Table 2.1 is continued)

1998	Song & Lee	Timed Marked Graphs and a mixed integer programming model	JSS with no buffer
(with meta heuristics)			
Year	**Author(s)**	**Scheduling Approach**	**Application Area**
2000	He *et al.*	PNs with decomposition method, and Simulated annealing with a deadlock recovery procedure	A flexible disassembly cell
2001	Chen *et al.*	Colored Petri nets and Genetic Algorithms	Wafer fabrication
2003	Jain *et al.*	Generalized Stochastic PNs and Simulated Annealing based methodology	Wafer fabrication
2004	Kumar *et al.*	Neuro-Expert Petri net (NEPN) model	Wafer fabrication
2004	Gang & Zhiming	Petri nets and Genetic Algorithms	FMS (deadlock-free scheduling)
2005	Cavory *et al.*	Petri nets and Genetic Algorithms	A cyclic job-shop

In this section, Petri net applications in the area of production scheduling were reviewed, and the historical progression in this field was emphasized. The outlines of the literature review can be summarized as follows:

- ✓ The first applications of PNs for scheduling of production systems were developing a scheduling system based on PNs with heuristic rule based systems to make decisions for on-line scheduling, and to evaluate the different scheduling policies. In dynamic scheduling problems, to obtain an acceptable solution in a shorter time is desired. Hence PN execution with rule-based scheduling system has been extensively used.

- ✓ However, rule-based scheduling and control systems are often specific to particular applications. Thus it is often difficult to generalize simulation models and their results. Recently the combination of PNs execution with the other search techniques such as AI search techniques, branch-and-bound technique and beam search has been paid great attention. Since the generation of the entire reachability graph and finding the optimal path from initial marking to final marking is very difficult even for simple PN models of small size systems, search algorithms including heuristic functions were employed to find good solution in a considerable time. By this way only a portion of the reachability graph is generated. Depending on the heuristic function used, the scheduling algorithm finds a global or near optimal solution. The heuristic function must be admissible in order to guarantee the optimum solution. It must provide a lower bound for the objective function. On the other hand, the search algorithm with an admissible heuristic function often makes the problem impractical for even a small size problem. Thus, the main problem is the difficulty in balancing the control of the search effort while maximizing the admissibility.

- ✓ Several new classes of PNs - *Buffer-nets (B-nets)*, *Chameleon Systems*, etc - were introduced, and search space was reduced by new heuristic functions

using the properties of these new classes or by new scheduling algorithms based on these nets. Although the proposed approaches provided promising improvements on reducing the search effort, it must be noted that both are very important; to have a proper heuristic function and to keep the modeling power of PNs.

- ✓ On the other hand, mathematical foundation of PN theory also allows the combination of PNs with some mathematical techniques. In literature, there exist successful studies combining PNs with some mathematical techniques. But, they present a limited modeling power as they need a special class of PNs, e.g. Timed Event Graphs, and Controllable Output PNs, to model a production system.

- ✓ Since ordinary PNs are limited to model complex nature of production systems, they have been extended to high-level PNs, namely Colored PNs, Evolution PNs (E-nets), predicate/transition PNs, and Extended High-level Evaluation Stochastic PNs. These extended classes of PNs were used with a rule-based expert system to describe for a real-time scheduling problem. They give, however, a good overview on a high level, the detailed behavior of the system may not be graphically represented, and the search process for generative scheduling may be slower than the equivalent standard PN model since the data handling associated to the firing of transitions is complex.

- ✓ Real-life FMSs consist of resources with limited capacities. The limited resource capacities can lead to deadlocks during resource allocation process. Therefore, to obtain a feasible solution, unlimited buffer capacities are usually assumed. However, these assumptions are not realistic for real-world scheduling problems. The studies on the PN based scheduling promote the use of PNs for dealing with deadlock free scheduling problems. Because, compared to mathematical programming approaches, deadlock states are explicitly defined in the PN framework, and no equation is needed to describe the deadlock avoidance constraints.

- ✓ High computation effort and memory requirement, and the rapidly growing search space problems in large size PNs are handled by some truncation or decomposition techniques. Although the decomposition techniques reduce the complexity of the scheduling problems and the computational time dramatically, they have some risk of going far from the real-system.

- ✓ Recently, there has been a growing interest in merging PNs and object-oriented approaches to provide the combination of graphical representation and mathematical foundation of PNs with the abstraction, encapsulation, and inheritance features of object-orientation.

Production scheduling based on PNs is a fruitful area for researchers. The modeling capabilities and formulating advantages make PN based methods attractive. PNs have high potential for many novel applications if future research has placed greater emphasis on the hybridization of PNs with evolutionary/meta-heuristics methods. Since real-world problems are large sized and complex, the future research issues have to handle the large-scale problems. If good heuristics can be used to generate efficient solutions, then enormous improvement on the total solution time problem can be achieved. The relevant literature needs new theory, practice and benchmarks.

CHAPTER THREE
A CONCEPTUAL FRAMEWORK OF A DECISION SUPPORT SYSTEM FOR REAL-TIME SCHEDULING OF FLEXIBLE MANUFACTURING SYSTEMS

Operations planning and scheduling (OPS) problems in FMSs, are composed of a set of interrelated problems, such as part-type batching, machine grouping, part routing, tool loading, part input sequencing and on-line scheduling (Mohamed, 1998). Prior studies in literature point out that the performance of an FMS is highly dependent on the efficient allocation of limited resources to tasks, and it is affected by the choice of scheduling rules. In this section, a conceptual framework of a high-level PN based Decision Support System using Object-oriented design approach, which integrates loading, part inputting, routing, and dispatching issues of the OPS is discussed.

3.1 Scheduling Problem

Scheduling concerns the allocation of the limited resources to tasks over time. It is a decision making process that exists in most manufacturing or service systems (Pinedo, 1995). In a manufacturing system, resources represent machines, operators, robots, tools, buffers etc., and activities are the processing of products on machines, the transportation of products among workstations, or loading/unloading the parts from/to machines by the operators.

Developing a schedule requires detailing a sequence of operations for each job on the given facilities, such that the task is completed. Scheduling problems are often complicated by large numbers of constraints relating activities to each other, resources to activities and to each other, and either resources or activities to events external to the system (Morton & Pentico, 1993). Product orders have to be released

and have to be translated into jobs with associated due dates. The jobs often have to be processed by the machines in a work center in a given order or sequence. Jobs may have to wait for processing on machines that are busy, and preemptions may occur when high-priority jobs arrive at machines.

A solution to a scheduling problem usually requires two sub problems to be solved. The *allocation problem* is to decide which resources should be allocated to perform the given tasks. The *sequencing problem* is to determine when each task will be performed. The scheduling function in an organization or system has to interface with many other functions (Pinedo, 1995). In Figure 3.1, information flow diagram in a manufacturing system is shown.

Scheduling problems can be divided into different categorizes. The first distinction is based on the degree of certainty in the problem; *deterministic* and *stochastic* scheduling problems. In the former class, all parameters in the problem are known without uncertainty, whereas this is not the case in the latter.

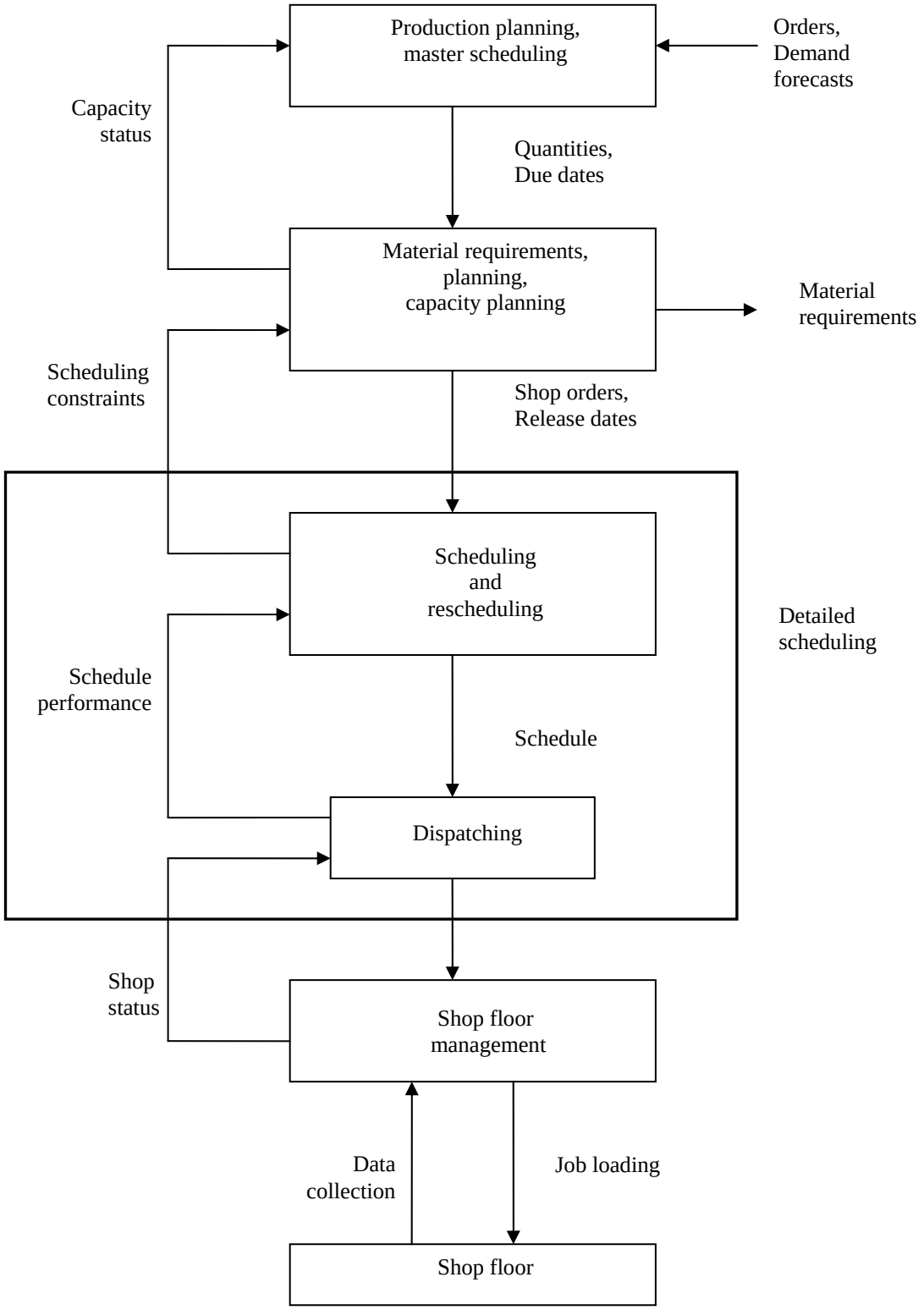


Figure 3.1 Information flow diagram in a manufacturing system (Pinedo, 1995)

Another classification is *static* (*off-line*) *vs. dynamic* (*on-line*) scheduling problems. In static scheduling problems, all jobs that are to be completed within the time horizon of the scheduling problem are known. When it is referred to the dynamic scheduling, it means that jobs arrive over time, and information about arriving jobs is not known in advance. In dynamic scheduling problems, new jobs can be added during the scheduling time period. Practical scheduling problems are dynamic, uncertain and often unpredictable due to the continuous arrival of new and unforeseen orders, and the occurrence of all kinds of disturbances (e.g. machine breakdowns, process variation, etc). However, from a traditional viewpoint, scheduling is performed as an off-line activity where operations that are known prior to production are scheduled before the production starts. The potential problem with generating off-line schedule subjects to inevitable changes on the shop floor owing to the uncertainties inherent in real life environments such as uncertain arrival of jobs, due date changes, shortage of materials, and so on. Due to these factors, rescheduling has to be made after any change in the system. From this perspective, the traditional off-line scheduling approaches cause increased waiting times, increased work-in-process inventory, low equipment utilization, and eventually degrade the system performance (Chiu & Yih, 1995; Wu & Wysk, 1989). Therefore, the assumptions made during the schedule generation make off-line schedules hard to implement as generated (Saygin *et al*., 2001). Consequently, reactive (and/or proactive) scheduling, which can react to dynamic and stochastic manufacturing environments is required for today's manufacturing scheduling problems (Chong, Sivakumar, & Gay, 2003).

Besides, scheduling problems are known to be NP-complete, that is the computation time required to obtain the global optimal schedule grows exponentially with the problem size. In recent years, the growth in the complexity of today's manufacturing environment causes the scheduling problems to become more difficult. Therefore, some of the existing scheduling techniques are improved, or new scheduling approaches are developed, since they lack in the formulation and/or the solution of the problems.

The simplest scheduling problems include only one resource and a set of activities (*single machine scheduling*) with a given processing times to be performed under some constraints. A natural generalization of single machine problems is to have a number of parallel machines (*parallel machine scheduling*) that can perform the activities. In this case all machines are identical and processing times are given for the activities, and some performance measures are tried to be optimized. This is equivalent to balancing the load on the machines as much as possible.

Furthermore, in *general shop scheduling*, we have a set of *m* different machines and a set of activities with precedence constraints between them given, and no further restrictions are imposed on the problem. If the order of operations within each job is uniquely specified, we have *the job-shop scheduling*. If there are no precedence relations between the operations in the jobs, the problem is called *open shop scheduling*. Once all jobs have the same processing order on the machines, we have the special case of *flow-shop scheduling.*

Another difficulty of scheduling is the conflicting objectives. Ideally, the objective function should consist of all cost in the system that depends on scheduling decisions. In practice, however, such costs are often difficult to measure, or even to identify completely. Nevertheless, three types of decision making goals seem to be prevalent in scheduling: efficient utilization of resources, rapid response to demands, and close conformance to pre-described deadlines (Gupta, Sivakumar, & Sarawgi, 2002). Some of the most common objectives of scheduling are (Baker, 1974):

1. Meet due dates
2. Minimize average flow time through the system
3. Minimize the total number of tardy jobs
4. Minimize the average tardiness of the jobs
5. Minimize the maximum tardiness of the jobs
6. Minimize work-in-process (WIP) inventory
7. Provide for high machine/worker time utilization

8. Minimize production costs

In these objectives, (1)-(5) are aimed primarily at providing a high level of customer service, and (6)-(8) are aimed mainly at providing a high level of plant efficiency.

3.2 FMS Scheduling Problem

The scheduling problems in FMSs, relate to the execution of production orders and include raw part input sequencing, machine, material handling device and operator scheduling, monitoring the system performance and taking the necessary corrective actions (Chan & Chan, 2001). Since FMSs contain very diverse properties and constraints (e.g. the availability of alternative machines to perform the same operation(s), multi-resource sharing, and product varieties), scheduling problems in FMSs are more complex and often very difficult to be solved by conventional optimization techniques. In studies on FMS scheduling problem demonstrate the importance of scheduling decisions for system performance (Saygin *et al*., 2001).

As O'Kane (2000) explains "The task of choosing a suitable schedule for an FMS can present immense problems due to the inherent intricacies of such systems and the scale of the scheduling problem within a flexible system can be a function of many factors which relate to the way in which the system is designed and planned and how it evolves". A General Flexible Job-Shop Scheduling Problem may be stated as (Yu *et al*., 2003):

- m resources are available $\{R_1, R_2, \ldots\ldots, R_m\}$,
- n jobs are to be processed $\{J_1, J_2, \ldots\ldots, J_n\}$,
- Each job J_i has p_i process plans,
- Each plan p_{ij} (indexes i and j represent the job and plan, respectively) has a sequence of q_{ij} temporarily related tasks $\{T_{ij1}, T_{ij2}, \ldots\ldots\ldots, T_{ijq}\}$ ordered by the technological constraints,

- Each task can be processed in several ways. C_{ijk} stands for the number of different possibilities (choices) of achieving task T_{ijk}, S_{ijkl} is the k th set of resources required for choice l of task T_{ijk}. h_{ijkl} is the processing time for task T_{ijk} using resource set S_{ijkl},
- Ready time and or due date for each operation,
- The processing of task T_{ijk} using resource set S_{ijkl} is called an operation, O_{ijkl}.
- Set-up times of each operation (st_i).

An FMS is designed to manufacture a variety of items simultaneously and to provide alternative processing routes for individual products. Routing flexibility, i.e. the ability to produce a part by alternate routes through the system, is an important property of FMSs that increases the decision domains and provides the potential for improved performance by better balancing machine workloads (Byrne & Chutima, 1997).

Many studies in real-time FMS scheduling and control area assume the operations of a job to follow a fixed sequence and do not consider the influence of routing flexibility (Saygin *et al*., 2001). Consequently, the influence of routing flexibility on the performance of FMSs is rarely addressed in literature. The control system of a random-type FMS is required to have the capability to adapt to the randomness in arrivals and other unexpected events in the system by effectively using operation and routing flexibility in real-time (Rachamadugu & Stecke, 1994).

Existing methodologies for alternate route/machine selection are primarily off-line methods based on mathematical programming, and they have focused on the development of analytical and algorithmic solutions for highly constrained problems. However, static schedules and strategies, developed ahead of time, quickly lose validity in a rapidly changing environment as some new constraints are encountered or even that the constraints used while generating schedule have already changed significantly (Byrne & Chutima, 1997). Thus, generated optimal process plans becomes suboptimal or infeasible, and they can not be directly applied over long

planning horizons. On the other hand, mathematical or analytical models rely heavily on the assumptions and simplifications upon which they are based. Therefore, in complex situations such as the dynamic scheduling of an FMS, the development of a single global function that accurately describes the relationship between input variables and the system output becomes impossible without risking significant loss of model integrity. Moreover, analytical methods usually require repetition as the status of the system changes and assume that all the parts are ready at time zero, which is rarely the case in practice. This makes them too computationally inefficient to be real-time decision criteria for alternate route selection (Peng & Chen, 1998). As a result, analytical method's ability is limited in handling real manufacturing problems, due to either their failure to handle many realistic and dynamic features of an FMS or lack of providing analytical solutions within reasonable time. To deal with the challenge, an effective and dynamic scheduling system that can maintain its performance while reacting to changes in a timely manner is essential (Chong *et al.*, 2003), as different parts arrive for processing in the system at random points in time. Although, the heuristics do not guarantee optimal results, they are usually more computationally efficient compared to analytical methods, and give the reasonable good solution in reasonable time. (Ozmutlu & Harmonosky, 2005). However, the heuristics proposed in the area of alternate route selection are also few. The most preferred approach to job shop scheduling in the industry is dispatching rules. Nevertheless, using single dispatching rule is not enough to consider all of the available resources at the same time, since the rigid structure of the dispatching rules exclude the use of other useful information that may be available for scheduling (Subramaniam, Lee, Hong, Wong, & Ramesh, 2000b). Second, there is no single universal dispatching rule which allow for the use of multiple criteria in the decision making process, as dispatching rules often influence only one performance measure, whereas scheduling in the industry usually requires the meeting of several objectives simultaneously (Subramaniam, Ramesh, Lee, Wong, & Hong, 2000a).

Since the operating environment of an FMS is dynamic, the nature of the scheduling will change over time and therefore the static rules based on having all information in advance are not appropriate, thus dispatching rule will also need to

change over time (Subramaniam *et al.*, 2000b). A new and simple alternate routing heuristic, which would be superior to conventional routing strategies in terms of various system performance measures, could prove to be very useful. The effective choice of dispatching rules depends on the scheduling criterion and existing job-shop conditions. Therefore, there is a certain need for a dynamic routing decision heuristics usable for real-time scheduling and decision-making, which would yield better results compared to single dispatching rules and be computationally efficient and easier to apply than optimization-based methods in more realistic situations (Ozmutlu & Harmonosky, 2005).

3.3 Problem Definition

In literature, in most of the applications on PN based scheduling, small size problems which do not include routing-flexibility, and setup operations, operator and vehicle constraints have often been solved ignoring material handling action, and limited intermediate storage. However, in real situations, subsystem optimization such as part-machine scheduling, isolated transporter or operator control can cause system under-performance. Although machines are major constraints of a scheduling problem, an operation can not be started if all need resources for that task are not available. Therefore availability and current status of other shared resources including transporters, operators and limited buffer sizes should also be taken into account in decision-making process, as they have also a great impact on the system performance (Lin, 1993). Thus scheduling rules should concern the allocation of the mixture of resources in a shared multiple resource environment (Sivagnanavelu, 2000).

Indeed, in most real shops, managers rely on flexible workers that can be moved to different workstations as the shop load varies. From this standpoint, the role of workers and decision rules that govern worker assignment to different workstations or departments has to be also considered in the context of part routing and sequencing (Kher, 2000). Moreover, part types in most of the FMS may be large size

and may require a lot of floor space and large sized buffers (Chan, 2003). Therefore, the cost of implementing buffer storage in FMS is very high, thus the capacity of local buffers is always an important constraint in the design of an FMS (Chan & Chan, 2004). In addition, there is always a possibility that a particular station can be blocked or the system can be locked due to the limited buffer spaces. Blocking occurs when a station can not move its product to a buffer if the buffer is full, whereas locking occurs when the system is totally prevented from functioning, i.e., no product movement can be achieved in the system. Thus the limited buffer capacities must be taken into account in scheduling of an FMS. Finally, the procedures for scheduling of operations on machines and scheduling of transportation operations on vehicles in a FMS are closely interrelated. Thus the transportation operations are also considered with the scheduling of machining operations as well as operator and limited buffer constraints. Through the literature, there is not much importance given to the simultaneous allocation of additional resources required for operating an FMS, by the way to incorporate different aspects of the scheduling problem.

The aim of this study is to show, how a dynamic scheduling problem of an FMS can be formulated in a timed Petri nets framework, and how this model can easily be extended to cover realistic scheduling applications in practice. For this purpose, a DSS which is coupled with high-level PNs and Object-oriented design approach is proposed. Proposed DSS provides computerized support for decision making through tracking of raw parts, work-in process, and resources in the system, maintaining current status information of the system, and properly generating the required data. A heuristic rule based approach is used to solve resource contention problems and to determine the best route(s) of the parts having alternative routes. The problem of resource contention reveals by the concurrent flow of competing parts for sharing limited resources such as machines, buffers, AGVs, and operators in a manufacturing system. In order to cope with the problem of resource contention in real-time, the system controller must be capable of effectively resolving the conflicts caused by the resource sharing problem (Jain, 2001). Resource sharing may also increase the

scheduling complexity of a PN model because transition firing conflicts caused by shared resources need also to be resolved (He *et al*., 2000).

We are thus considering four types of extension to the standard scheduling problem: multi-resource, resource flexibility, nonlinear routing, and sequence independent machine setup operations.

- *Multi-resource*: an operation may need several resources at the same time to be performed.

- *Resource flexibility*: a resource (e.g. machine, operator, transpallet etc.) may be selected in a given set.

- *Nonlinear routing*: in a standard job-shop, the routing is linear. Here, the number of predecessors and the number of successors of an operation in the routing may be larger than one.

- *Machine set-up operations*: The setup operations have to be considered while taking the scheduling decisions.

Consequently, the problem is thus to both make an assignment and a sequence of the operations on the resources that minimize a performance criterion. Thus, the scheduling decisions in such environments must consider workstation capacity, operator and material handling device availability, and machine set-up status at the same time.

3.4 A Decision Support System (DSS) for Real-time Scheduling of FMSs

A Decision Support System (DSS) is an interactive computer-based system intended to help managers to take more effective and efficient decisions, and it can be defined as a set of procedures in a model format which are intended to process

data and judgments in assisting managers in their decisions. Thus, the decision can be considered as a consequence of dynamic interaction between three overlapping circles of information, preferences, and alternative solutions (Boose, Bradshaw, Koszarek, & Shema, 1993).

In recent years, as a result of increasing levels of automation and implementing concepts such as Flexible Manufacturing systems, and Computer Integrated Manufacturing (CIM), manufacturing facilities and their control systems increase in complexity. In this thesis, we propose an adaptive, robust, integrated flow control framework that is capable of handling the complexity of the real-world manufacturing systems. The proposed DSS exploits a set of technologies and theories, such as databases, graphical interface, high-level PNs, expert systems, the object-oriented modeling approach, and client/server paradigm.

A specific DSS provides support on the strategic level, on the operational level, and problem solving processes. On the strategic level, it supports the selection process of the most efficient design alternative that meets the user requirements. On the operational level, it assists the production managers in analyzing the performance of their existing system and in determining possible ways for modifying the production system to improve the overall performance. A DSS with its ability of helping decision-makers utilize data and models in a semi-structured or unstructured environment is an ideal candidate that can be easily implemented to solve such a scheduling problem (Schniederjans, Carpenter, 1996).

3.4.1 General Characteristics of the Proposed DSS

In the shop floor level planning, the objective of DSS for scheduling is not only to obtain a short term schedule of the operations, but also to make the system interactively response to any dynamic changes by the scheduler. The proposed DSS fulfills the general requirements stated in Abu Hammad (2001), chap.2 and Wu

(1999), chap.1. The main purposes and features of the proposed DSS can be stated as follows:

1. Be capable of handling the complexity of the real-world manufacturing systems to meet the shop-floor control criteria: High-level PNs allow microscopic analysis of complex system dynamics giving the detailed understanding required to maximize the efficiency of such systems. As well as being used to check behavioral and structural properties of system, and explain the operation of complex systems, PN models are also used in real-time control systems to provide decision support of automated (intelligent) decision maker, or to support a decision making process at a long term strategic level.

2. Adopt a structure and architecture for heterarchical control of manufacturing systems focusing on distributed information and distributed decision making paradigm. The system described here employs an object-oriented modeling of the environment and adopts distributed decision making paradigm. In this framework, a manufacturing system operates through the cooperative behavior of many interacting subsystems which may have their own independent data/attributes, interests, values, and manners of operations/methods (Rumbaugh, Blaha, Premerlani, Eddy, & Lorensen, 1991). Thus, control can be distributed throughout the system by flexible and efficient interactions among all entities of the system, and conflicts can be solved by defining some well-formed models, and simple rules.

3. Support all decisions phases: intelligence, design, choice, and implementation: For this purpose, a knowledge representation technique which comprises rule-based representation (Production rules) is employed to incorporate scheduling knowledge and control strategies in control logic of PNs. The use of distributed computing provides the opportunity for simultaneous decision making and speeds up the response time (Lin, 1993). Thus, it can react well to

changing environment and becomes a real-time control system for distinct activities undergoing concurrent execution.

4. Adapt to changes over time by adding, deleting, and rearranging basic elements in the flexible system structure. The system can be considered as a population of intelligent entities operating in cooperation to succeed in attaining individual and global goals. As Lin (1993) explains, object-based architecture is used to keep the modeling framework separate from system details so that any change can be made without affecting the functionality of other components and the analysis can be performed quickly at a moderate cost. Therefore stability of the concept can be kept, and so the framework remains constant even when some parts of the system changes. Flexibility feature of the proposed DSS is thus attained.

5. Modularity: Modeling in the object-oriented paradigm consists of describing the system as a collection of objects that interact with each other by sending or receiving messages to carry out the desired behavior. Internally, an object may perform its function in ways which are concealed from other objects in the system. Thus, each entity has its own control communicating with other entities in the system through message passing. These properties ensure that objects will be portable, modular, and maintainable (Lin, 1993). The system is capable of handling new features easily and is able to function, so any variety of the number of the components has minimal effects.

6. Support, but not replace, the decision maker in semi-structured and unstructured cases: It provides a promising direction to support decisions required to generate production schedules in dynamic environment of FMS. By using Object-oriented description and high-level PNs, a complex FMS model can be constructed and managed easily (Ku, Huang & Yeh, 1998). The resulting behavior of the entity system collectively determined.

7. Allow intelligence design and manage knowledge: Proposed DSS allows conducting many different experiments to quantify the effects of the added flexibility in various fields and explore methodologies to utilize this flexibility effectively. The existing system environment and global knowledge can be captured graphically in a concisely and systematically manner through the use of high-level PNs and object-oriented modeling approaches, and incorporating relevant resource allocation and part routing control algorithms. Through this visualization, understanding of the system model and communicating becomes easier.

8. Deadlock prevention: Occurrence of possible deadlocks is eliminated by incorporating some predetermined rules and structures into the class models in advance. By this way, we prevent probable blocking conditions and thus none of the system entities will be stuck in the system indefinitely in the well-formed models.

9. Improve effectiveness of decisions. The system needs to satisfy customer requirements. System models are utilized for analyzing and experimenting decision-situations, with different strategies under different configurations. Thus, various combinations of manufacturing system components can be tested by changing the system parameters.

The proposed DSS consists of main components given in (Turban & Aronson, 2001). The fundamental components of the DSS are as follows: a data base system, knowledge system, model-base system, and a user interface system

- ***Data base system***: the data base system includes information and data obtained internally or externally. It stores product types, process plans which contains the set of operations to be performed on the part, the order

constraints among operations, and resource requirements, local variables, object parameters and collects system performance measures,

- ***Quantitative model or system***: a model (e.g., high-level Petri net model, Object modeling technique diagram) that processes the data and performs certain functions;

- ***Knowledge - base system***: provides intelligence; It contains policy knowledge, heuristic rule base for the dispatch decision making, state descriptions, and evaluation of system performance,

- ***User interface system***: a control and dialog subsystem through which the user can communicate with the system. It includes a graphical interface, a natural language interface, and an interactive dialogue interface.

The proposed DSS has a general framework given in Abu Hammad (2001) and Zopounidis & Doumpos (2000). The general structure of the proposed DSS is depicted in Figure 3.2.

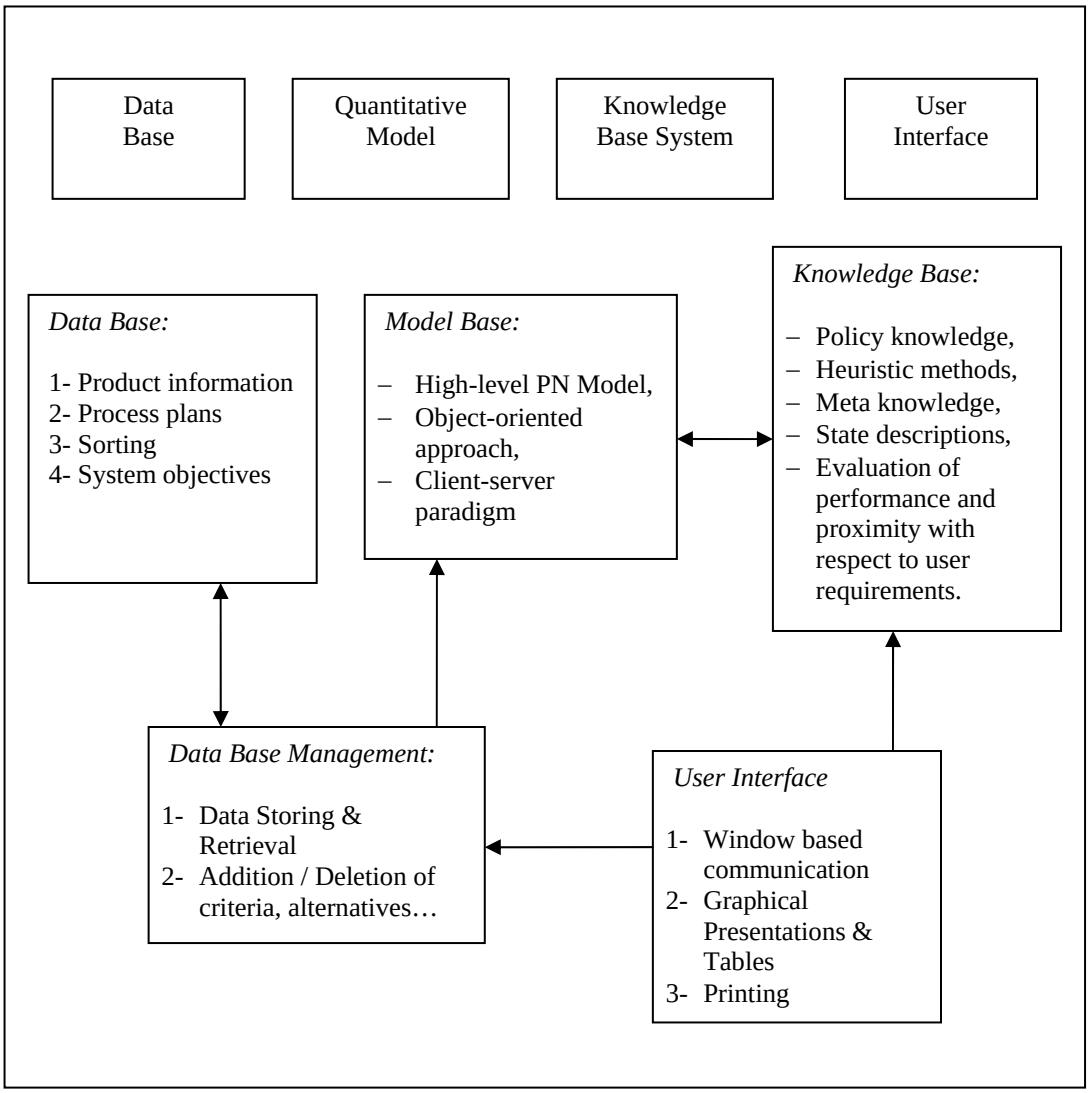


Figure 3.2 The basic framework of the proposed DSS for scheduling

The decision alternatives at any point depend upon the current system state. An integrated control paradigm which keeps track the system status and resolves the conflicts is developed by providing a powerful communication and efficient interactions among the system entities. The flow of material and information in the system under consideration is quite complex as it involves multiple product types, machine setup operations, alternative part routings, limited buffer sizes, and operators which have the same skill level and are able to work for any workstations, thus the system has a variety of scheduling and operational choices. Therefore, the control system involves a set of cooperative control processes: part routing and

sequencing controls, machine controls, operator controls, and transporter controls. A scheduling module which forms the core of the real-time control system is introduced to determine feasible operational sequences for each manufacturing request (i.e. a manufacturing order or production batch) according to the process requirements and current shop floor conditions, in providing sufficiently comprehensive information about the system at any point of time, in finding the tardy / late orders, and in achieving better resource utilization. For this purpose, a centralized routing method, in which route decision is made when a job waits to be transported to the next workstation, is proposed to solve the routing problem. The scheduling rules, which employ various priority attributes and relevant information concerning the availability status of resources are used in decision making process. This module communicates with other objects in the system through a communication network to control the entity flow and multi-stage resource sharing. By this way, a global knowledge and structure can be captured in a concise manner through the use of part routing control algorithms. This results that system can adjust itself smoothly and quickly based on the flow and load of the system. Moreover, deadlock prevention can be provided by bidding communication protocols between the system objects. Since the structure of DSS is highly modular, miscellaneous control strategies and rules can be easily embedded, and system can be tested with respect to various performance measures under assorted experimental conditions.

3.4.2 System Assumptions

The scheduling problem of an FMS is related to the execution of production orders and includes raw part input sequencing, machine, vehicle and operator scheduling, monitoring system performance and taking the necessary corrective actions (Chan & Chan, 2001). The assumptions made for modeling and analysis of an FMS are as follows:

- Each machine can process only one part at a time.
- Some machines are alternative to each other.
- Operation sequences for the jobs are fixed for a particular part type.

- There are no tooling constraints. Tools are available whenever required.
- Processing times on each machine for the predetermined operations on each part type under consideration is known in advance
- Material handling time for each movement is same for each movement (i.e. between any pair of workstations and from/to the loading/unloading station of the system)
- System input and output buffers have infinite capacity, however local buffers of the workstations and loading / unloading station have finite capacity.
- Machines needs for setup operation in order to process a different part type. Machine setup times are sequence independent.
- Job preemption is not allowed (i.e. a job, once taken up for processing, should be completed before another job can be taken up).
- Material handling device (transpallet) can only transport one part at a time.
- There is no material handling device breakdowns. Idle transpallets park in a waiting area.
- Loading and unloading of parts from/to machines and pallets are performed by operators.
- The operators in the system are cross-trained, (they have the same skill level), so that they can work for any operation in different workstations.

3.4.3 Modeling Methodology

FMSs are characterized by concurrency, resource sharing, routing flexibility, limited buffer sizes, and variable lot sizes. For interaction activities, and the coordination of individual units, a descriptive and dynamic modeling tool is required to model in detail the concurrency, and synchronization in the system with respect to time (Lin & Lee, 1997).

Owing to the complexity of these real-world manufacturing systems, it is very difficult to contain all the information necessary for scheduling in a unique model.

According to Lin, Fan, and Loiacono (2004), three views are required to be introduced to compose real-life dynamic scheduling model as the process view, the resource view, and the job view. Process view is constituted multi processes, each of which defines activities and their process constraints required to produce one type of product. Resource view involves the individual resource and resource pool. The resource pool is a classification of individual resources according to their functions or physical positions so that the individual resources in the same resource pool can be substituted for each other. This feature makes the model flexible in dealing with multi-resource sharing problem. Job view describes the properties of the jobs to be produced, which includes the product type, order arrival time, due date, order size, process plan, priority, and so on (Lin *et al*., 2004). Finally, control flow diagrams are employed to represent control issues such as decisions, dispatching and sequences. The views and the relationship among them present various constraints and conditions of the scheduling problem as shown in Figure 3.3.

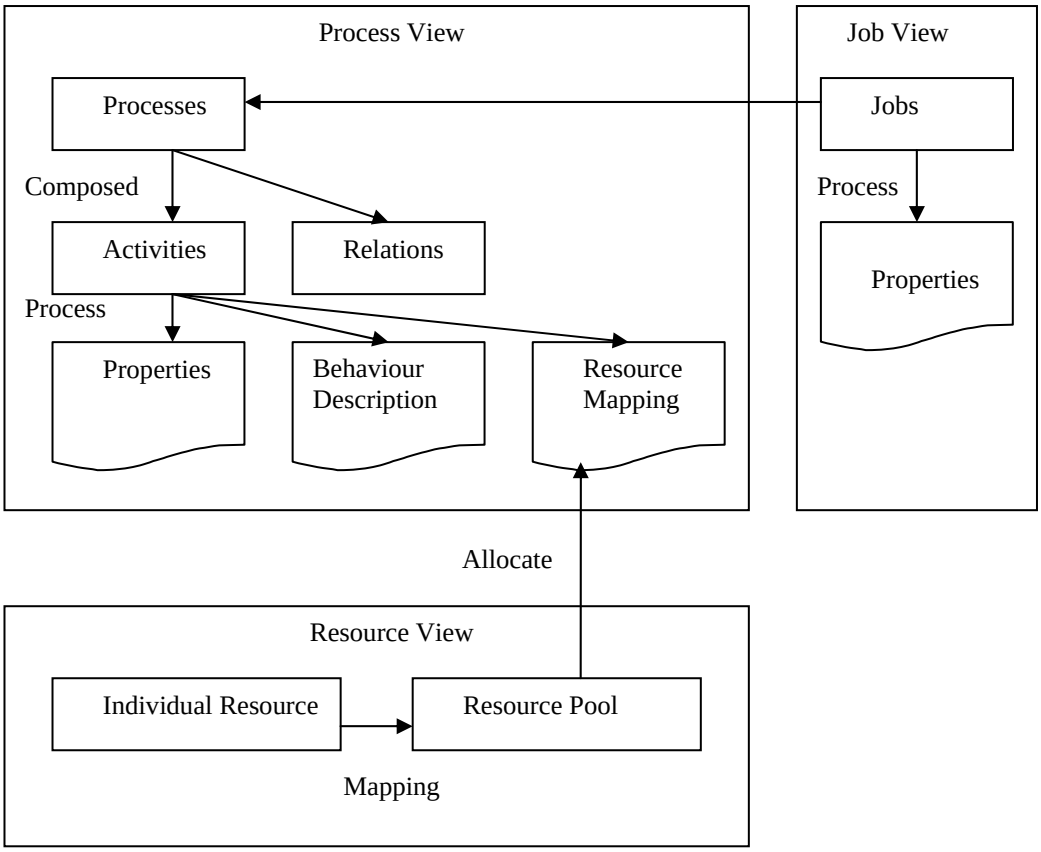


Figure 3.3 Fundamental structure of real-world scheduling problem (Lin *et al.*, 2004)

A manufacturing system can be viewed as a collection of objects with rules that govern their dynamics and interactions to generate desired objects (products). The objects can be represented graphically as simplified images, icons, and stored in a data base as members of a class of similar objects, sharing common properties (Zhou & Venkatesh, 1998). Therefore, Object Oriented Design (OOD) concepts can be employed to design of shop floor scheduling and control system in FMSs, and provide a way of defining manufacturing system using models organized around real-world concepts (Booth, 1998). OOD methodologies can offer the reusability, extendibility, and modifiability of software design.

By using OOD concepts, the properties and behavior of the real-world object are modeled by the data/attributes and methods/operations of the corresponding system object. It should be noted that real-world objects include not only physical objects, such as a machine, material handling device, robot, operator, and so on, but also logical objects, such as a scheduling process, routing process, product data, orders, bill of materials, and so on (Venkatesh & Zhou, 1998).

The fundamental building block of OOD is an “object” that contains both a data structure and a collection of related procedures. Procedures are also called operations or methods. Objects interact with each other by sending messages or by calls to their interfaces. Objects with identical data structure (attributes) and behavior (operations) can be grouped into a “class”. Each object is an instance of its class.

The most widely used OOD methodology is the object modeling technique methodology, known as OMT methodology, which is gaining increasing recognition as a powerful and robust system development methodology. The OMT methodology consists in developing three orthogonal models, as follows (Booth, 1998; Venkatesh & Zhou, 1998):

1. *Object model:* It divides the application into object classes and shows the static, structural aspects of the system in which objects, their identity and

their attributes, relationships, and operations are described in detail. The relationships (interfaces) among the classes are described by the class structure, and the behavior of the objects is defined by operations associated with the object class.

2. *Dynamic model:* It represents the temporal, behavioral aspects of a system and shows the way the system behaves with internal and external events by capturing the time-dependent behavior of the system.

3. *Functional model:* It represents the transformational aspects of a system and shows how to process the data flow in the system during each event or action.

PNs and OOD concepts are complementary to achieve the goal of system development at incremental stage, and enable us to model the complex manufacturing systems in detail, so that the strategies to operate these systems effectively can be applied in a more realistic environment. In this thesis, OOD concepts are employed to design of shop floor scheduling and control system in FMSs, and high-level PNs are used to model dynamic behavior of the system. The properties and behavior of the objects are modeled by the data/attribute and methods /operations. OOD methodology used for development of the DSS is summarized in the following steps (Chen & Chen, 2003; Venkatesh & Zhou, 1998):

1. Apply the OOD concepts to find objects in an FMS and to model the static relations among them by developing an object modeling technique diagram (OMT). The OMT diagram is used to represent explicitly different kinds of static relations such as generalization, aggregation, and association among the objects in FMS.

2. Use PNs to model the dynamic behavior of the FMS based on the static relations of the OMT diagram. Part, operator, and transpallet flows in the OMT diagram are transformed into token color classes in CPN model.

And FMS objects WSs, Loading/Unloading station, and part routing control object are modeled as CPN classes.

3. According to the system configuration, (i.e. number of workstations, number of transportation vehicles between workstations, capacity of load/unload station and buffers of workstations, number of operators, process plans, and batch sizes), FMS objects are derived from the corresponding CPN classes.

4. Integrate all the instance of the CPN classes into an integrated system model by linking the input and output places of each CPN object.

CHAPTER FOUR
DESIGN AND IMPLEMENTATION OF THE PROPOSED DECISION SUPPORT SYSTEM BASED ON HIGH-LEVEL PETRI NETS AND OBJECT-ORIENTED APPROACH

The conceptual framework of proposed decision support system for real-time scheduling of FMSs has been discussed in Chapter 3. Its modeling methodology using high level PNs and Object-oriented design approaches has also been presented. This chapter is devoted to the illustration of the modeling methodology discussed in the third chapter. Firstly, object modeling diagram of the system is constructed and a heuristic rule base is proposed to solve the resource contention problem, then the dynamic behavior of the system is formulated by high-level PNs. Subsequently, the proposed rule-based system is compared with common dispatching rules with respect to part flow time and tardiness related performance measures, then the simulation experiments were carried out under different experimental conditions to analyze the impact of the different levels of independent variables on performance measures.

4.1 System Description

Due to the unavailability of sufficiently large problem sets, the researchers studying on flexible manufacturing systems mostly generate their specific systems (Nayak & Acharya, 1997). In this section, an FMS with alternative operations and setup times is employed to explain the modeling methodology of the proposed decision support system for real-time shop floor scheduling and control problem (Tuncel & Bayhan, 2005).

The system consists of a loading and unloading station, nine workstations (CNC1, CNC2, Multiplex1, Multiplex 2, HMC1, HMC2, HMC3, Drill 1, and Drill 2), a

material handling device called transpallet that can load one pallet at a time and transport it among workstations or between load/unload station and workstations, and operators which have the same skill level and capable of operating at any workstation. Parts enter and leave the system through the loading/unloading station. Each workstation (WS) has one machine with a limited buffer capacity. Each machine can only process a part at a time, and needs setup time to change from one setup configuration to another for processing different types of operations. A pallet is unloaded from the transpallet in the input buffer of a WS, and transpallet becomes available for other transport operations. Then the part is unloaded from the pallet, and loaded to the machine to be processed. Once the machining operation of a part on a pallet is completed, the pallet is transferred to the WIP storage of the workstation until it is sent to the next destination. The time taken to transfer a part to the WIP storage is very small compared to the processing time. While machining buffer of each workstation is limited with one pallet, the WIP storage buffer, where the processed parts waiting to be transferred their next destination are stored, is allowed to keep at most 10 pallets at a time. When a job is completed for all the machining processing, it must exit the system through the unloading station. Then, the system's performance criteria (mean flow-time, mean tardiness, proportion of tardy jobs, system throughput rate) are updated. The manufacturing system is flexible to produce a variety of products which has alternative processing routes for some operations (i.e. two or more machines have ability to perform the same operations). Three types of products are produced.

The FMS operational policy is under push paradigm that machines process parts whenever they are available. The layout of the system under consideration is shown in Figure 4.1.

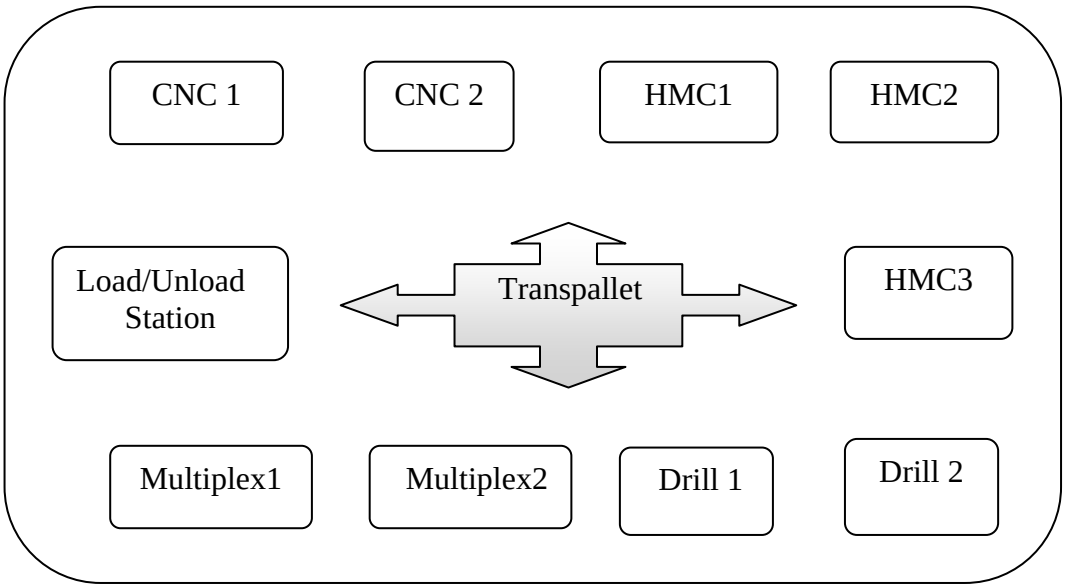


Figure 4.1 The layout of the FMS

For each product type, Tables 4.1, 4.2, and 4.3 show the operational sequences with required resources and processing times without setup times.

Table 4.1 Process plan of product type 1

Operatio	**Machine required – Processing time**
O1	CNC 1 (4.5 min.) / Multiplex 1 (M-1) (3.2 min.)
O2	CNC 2 (4.5 min.) / Multiplex 2 (M-2) (3.2 min.)
O3	HMC 1 (9 min.) / HMC 2 (5.2 min.) / HMC 3 (6 min.) / Multiplex 1 (M-1) (3.6 min.)
O4	Drill 1 (2 min.)
O5	Drill 2 (1.5 min.)

Table 4.2 Process plan of product type 2

Operatio	Machine required – Processing time
O1	Multiplex 1 (M-1) (3.7 min)
O2	Multiplex 2 (M-2) (5.3 min.)
O3	HMC 1 (13 min.) / HMC 2 (7.5 min.) / Multiplex 1 (M-1) (5.25 min.)
O4	Drill 1 (2 min.)
O5	Drill 2 (1.5 min.)

Table 4.3 Process plan of product type 3

Operation	Machine required – Processing time
O1	Multiplex 1 (M-1) (3.7 min.)
O2	Multiplex 2 (M-2) (5.3 min.)
O3	HMC 1 (13 min.) / HMC 2 (8.5 min.) / Multiplex 1 (M-1) (5.25 min.)
O4	Drill 1 (2 min.)
O5	Drill 2 (1.5 min.)

Processing routes of product types are shown in Figures 4.2 and 4.3. Product types 2 and 3 involve the same processing routes, but they differ in the processing time of third operation.

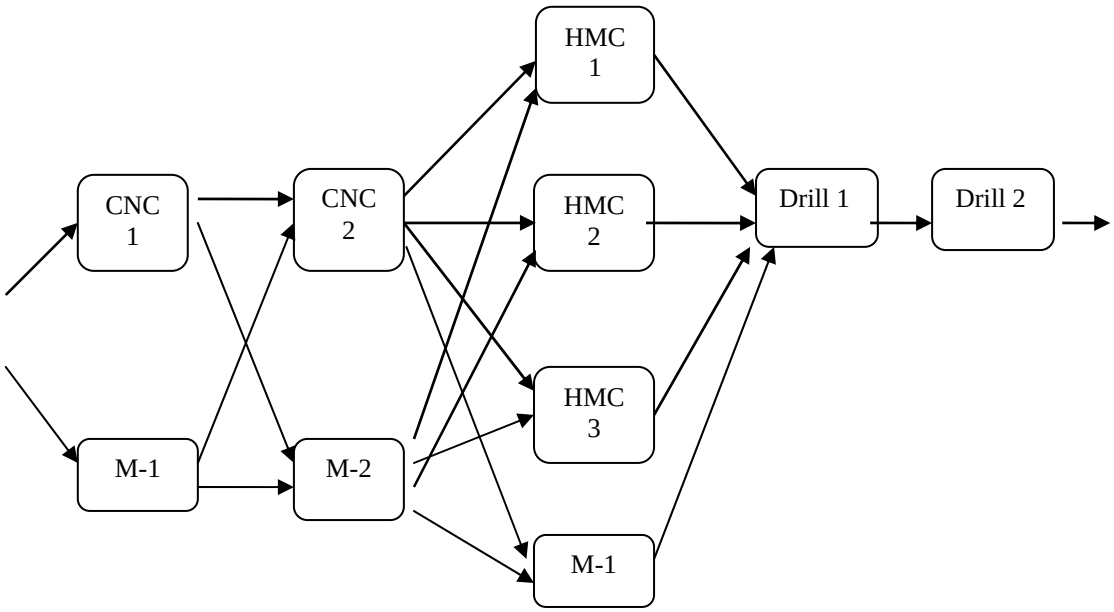


Figure 4.2 Process diagram of product type 1

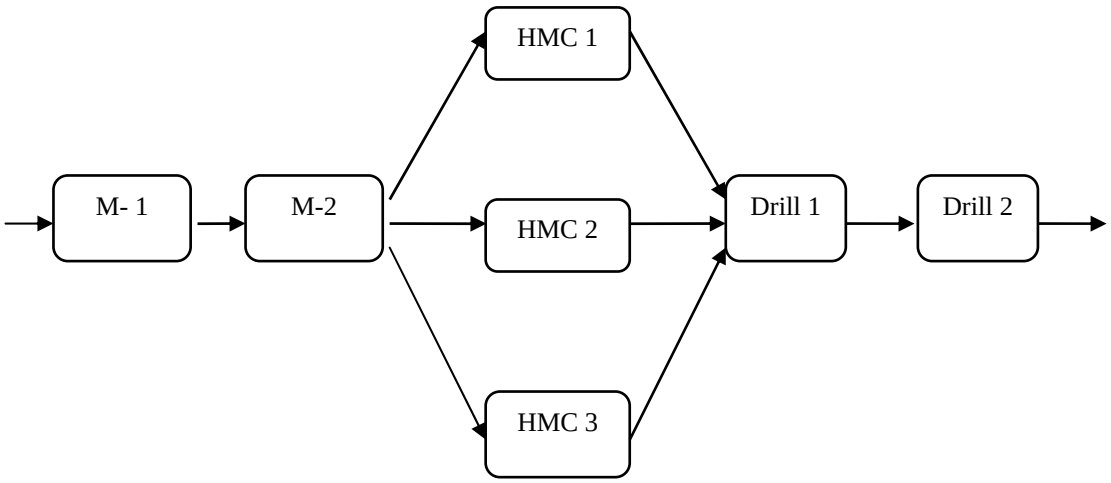


Figure 4.3 Process diagram of product types 2 and 3

4.2 Problem Statement

As mentioned before, some operations of the product types can be performed by alternative machines. From the point of part flow view three different types of products are processed simultaneously in the system, and each product is allowed to have flexible routing (i.e. two or more machines have the ability to perform the same

operation of a part). Employing flexible routing requires a two-level hierarchical decision making process: assignment and sequencing. In the first level, the next destination of a job is determined; in the second level the sequencing decisions of a part waiting for the limited resources is taken.

For scheduling of automated manufacturing systems, explicit recognition should be given to auxiliary resources such as material handling system, operators and buffer spaces. However, this will increase the problem complexity, since deadlocks may arise from explicit recognition of these shared resources. In the system considered, when the part type is changed, a setup operation is needed too. So, we also have to introduce setup requirements of the machines into the PN model of the system.

Each job has a best operation sequence that determines the order in which resources must be assigned to the job, but the efficient utilization of the resources on real time basis requires a real time resource allocation policy to assign resources to jobs as they advance through the system. Since each machine is capable of processing a variety of part types, and also since each part has to visit a number of machines, there is often a conflict when more than one part is contending for the same resources such as machine, material handling device, and operator. The problem is the allocation of resources to a set of tasks, that is determination of the best route of each task in the system according to the current shop-floor condition (due dates, release dates, order quantities, tardiness penalties, inventory levels, urgency situations, and setup times). However, formulation of real-life scheduling problem using traditional methods becomes very complex when routing flexibility, setup operations, operator and material handling system constraints are also considered. PNs are well suited for representing FMS characteristics such as precedence relations, concurrency, conflicts, and synchronization.

In the following section, the scheduling problem mentioned above will be solved by using the proposed DSS, and the effect of the different system configurations (i.e. the number of material handling device, operator, and buffer size) on performance

measures will be investigated. The performance of the heuristic rule base will be compared with some commonly used scheduling rules: FIFO, EDD, SPT, LNRO, SNRO and CR.

4.3 Object Modeling of the System

In developing an object-oriented control model, it is needed to define all the object classes required for the FMS, then the statistical relationships among these object classes are described. The object modeling technique (OMT) method that is the most widely used OOD methodology is employed to describe and analyze the object classes. It divides the system considered into object classes and it is used to model the static relations among FMS objects by the class structure. The behavior of the objects is described by operations associated with object classes. In Figure 4.4, generalization and aggregation of the FMS is presented. Aggregation can be described the “part-whole” or “a-part-of” relationship in which objects representing the components of a structure are associated with an object representing the entire assembly (Booth, 1998). Generalization or inheritance describes the relationship between a class and its refined version, and is used to share the similarities among classes while preserving their diversities (Kuo *et al*., 1998). Aggregation is drawn with a small diamond indicates the assembly end of the relationship, and the notation for inheritance is a triangle connecting a superclass to its subclasses.

Figure 4.5 shows the OMT diagram corresponding to the decision support system for shop floor scheduling problem of the FMS. This diagram captures the relevant properties of the FMS objects and their functions in the DSS. The links between objects with black dots at each end represent the associations, which can have attributes in dashed rectangles. For example, L/UL station loads parts with attributes such as job number (Job No), due date, processing time, source, and destination station. The flow chart of the FMS operation is presented in Figure 4.6.

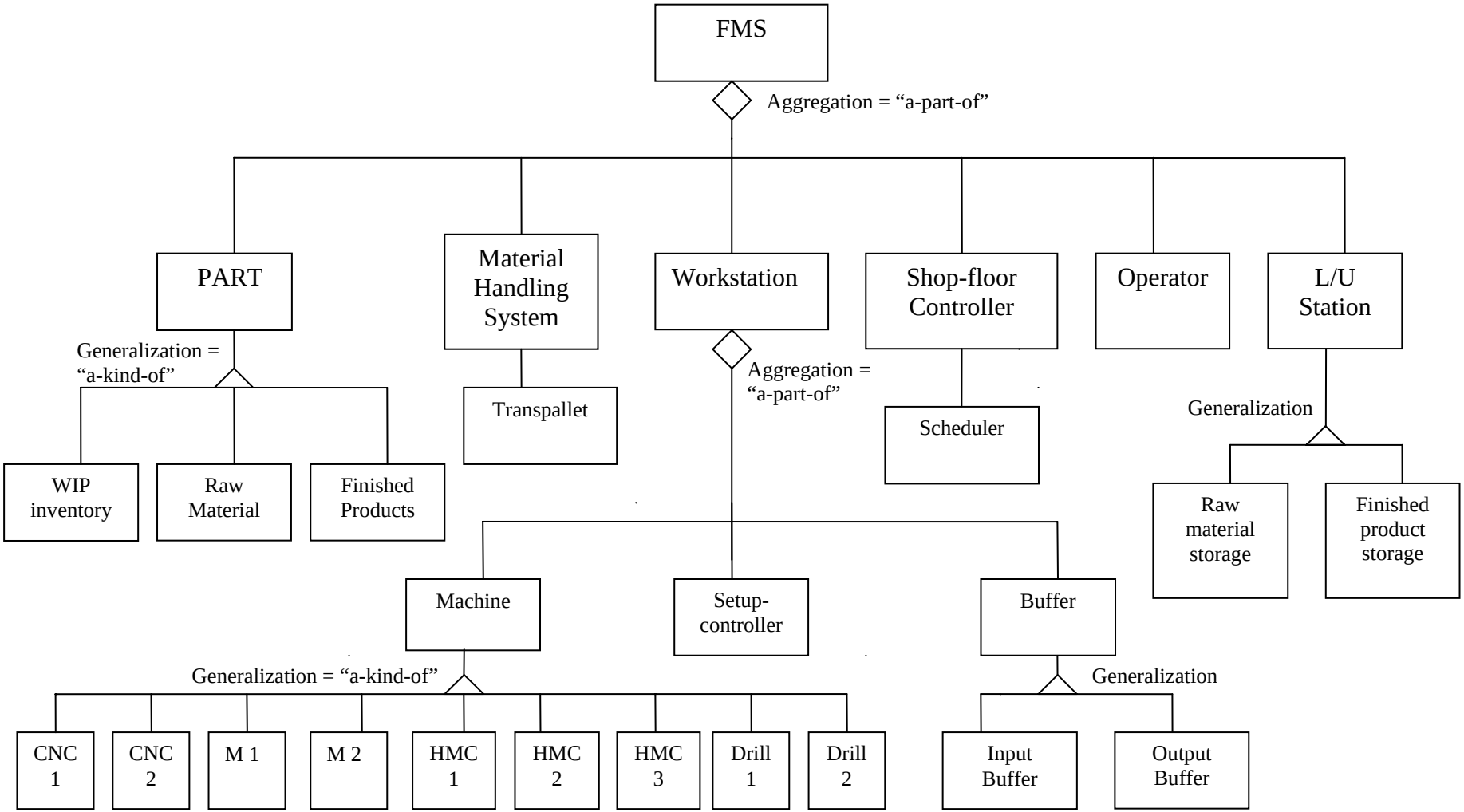


Figure 4.4 Aggregation and generalization of the FMS

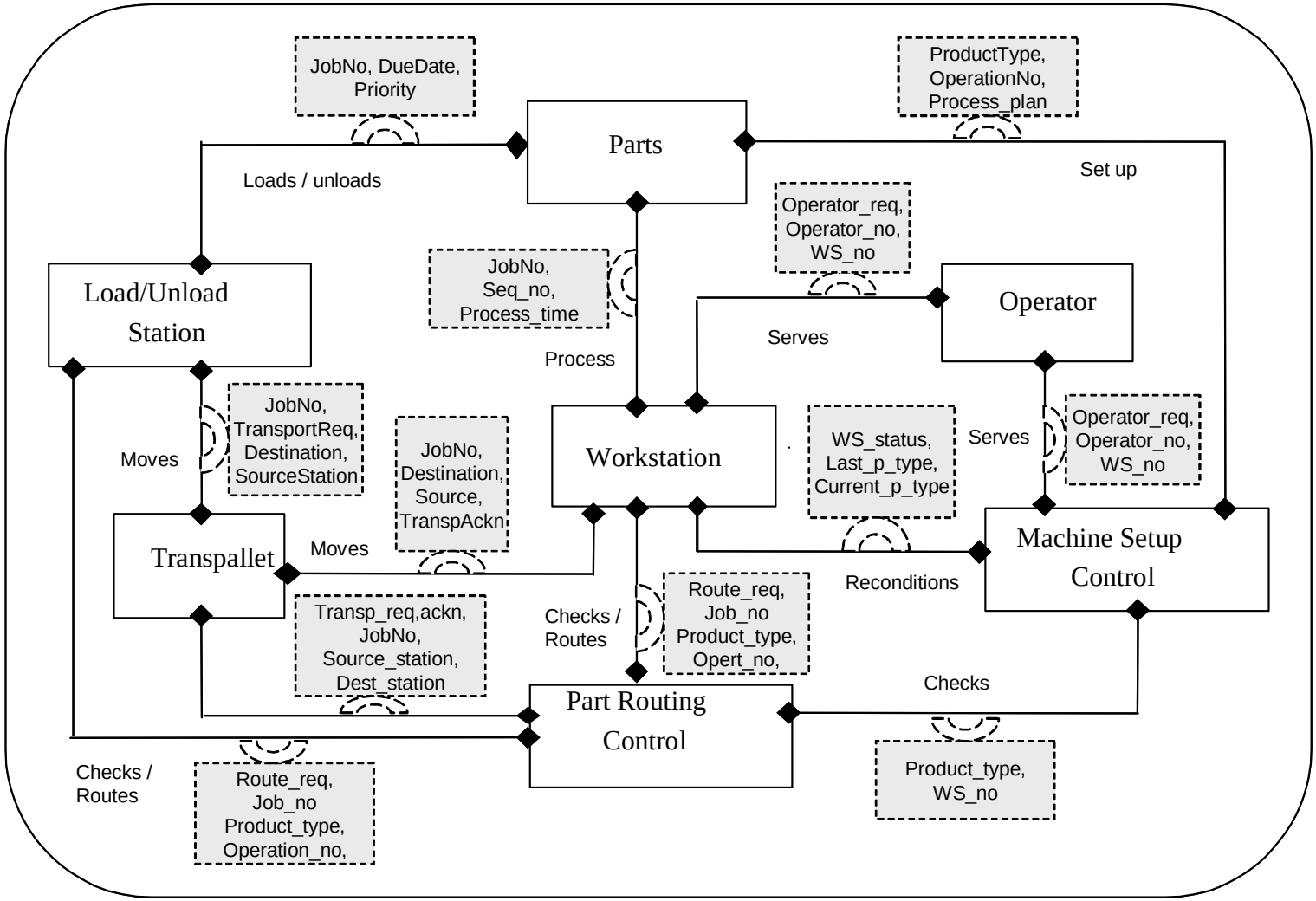


Figure 4.5 OMT Diagram

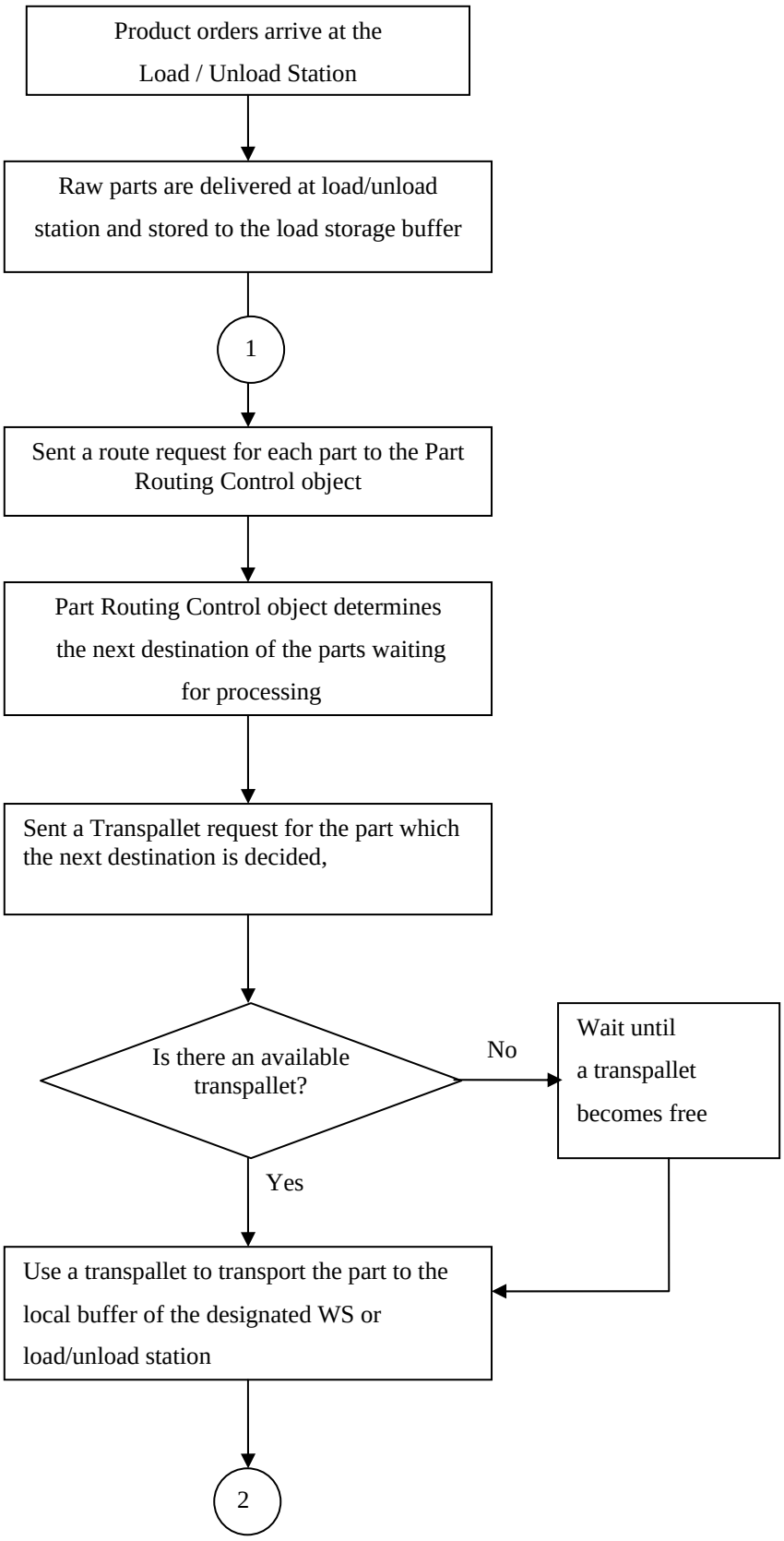


Figure 4.6 Flow chart of the FMS operations

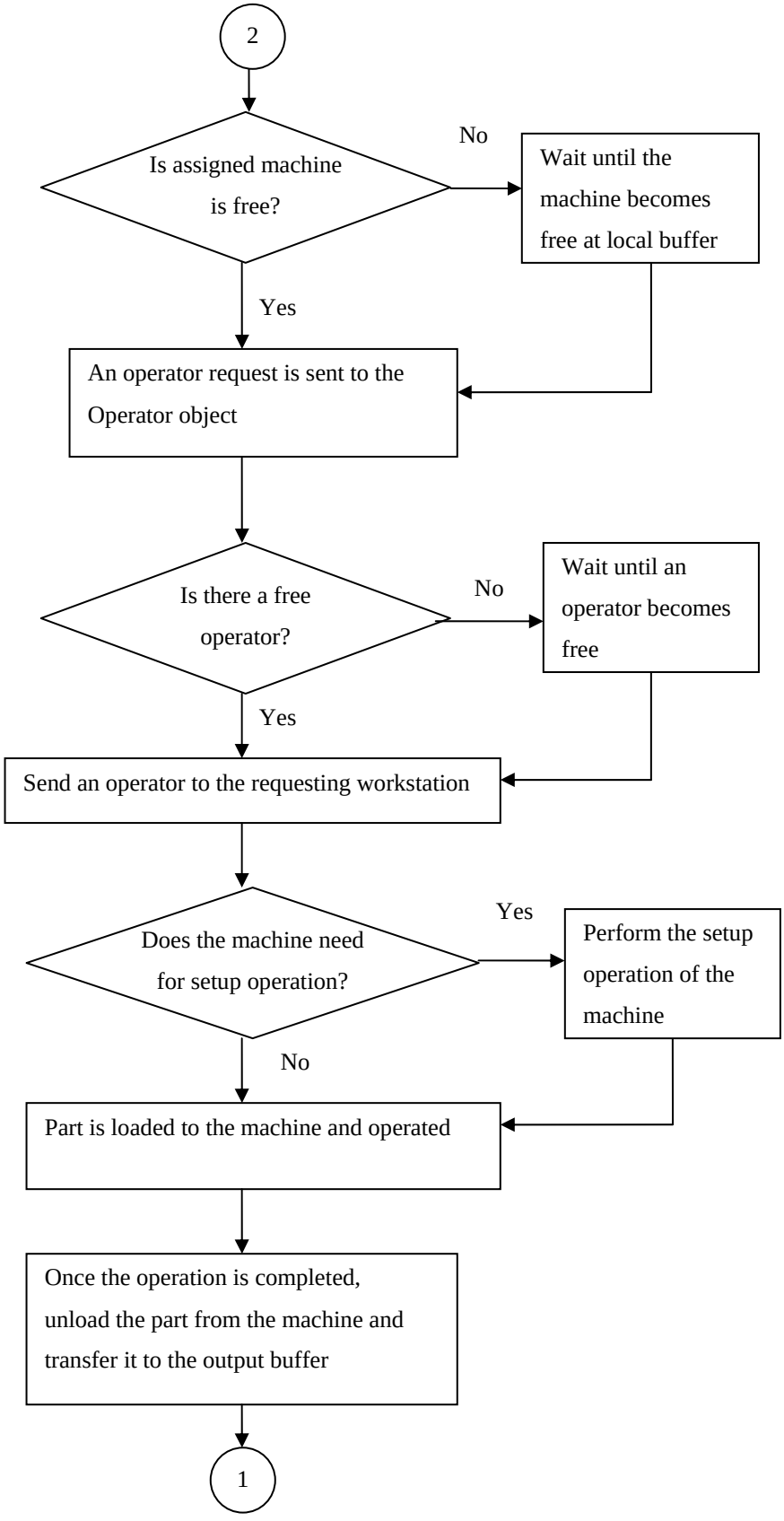


Figure 4.6 Flow chart of the FMS operations (continued)

4.4 Object-Oriented Modeling of the FMS

The OMT diagram of the system embodies the distributed decision making environment. After the OMT diagram is developed, dynamic behavior and control logic of the FMS is formulated by constructing high-level PN model of the system based on the static relations of the OMT model. For this purpose, CPN class models are first constructed to capture concurrency and synchronization of the system.

Individual classes of FMS objects are defined as follows:

- Load/unload station class
- Workstation class
- Part transport Class
- Operator class
- Scheduler Class

Since the standard PNs can be insufficient to model the complex and large size systems, they have been extended to the High-level PNs which allow arbitrary complex data types for tokens and to let expressions for places and transitions be constructed by using some special programming languages. In this study, Artifex PN software tool (RSoft Design Group, Inc., 2002) is employed to model the system. Artifex exploits a graphical language, which is based on extended Petri Nets, to define detailed models of a system or process. The Artifex language is integrated with standard programming languages like C and C++. This allows developers to use existing software components within their models, and to use C and C++ programming beside the Artifex language where more appropriate, as an example to implement algorithms and to perform complex operations on data.

Graphical elements of the Artifex related to the high-level PNs are as follows (RSoft Design Group, Inc., 2002):

Places are the elements of a class that may represent either one of the possible states of a class or a data-store containing units of information called tokens. A place

is represented graphically by a circle, which is labeled with its name followed by the type of tokens to be contained in it. Optionally, the label may contain the number of initial tokens present in the place at the beginning of the execution of the model and the queue type. There are several kinds of places;

- *Internal* or *normal places* are the basic ones and they are represented by a circle.

- *Input places* are class interface places where an object can receive tokens that come from other objects. Input places are represented graphically by two concentric circles.

- *Output places* are class interface places that an object can use to send tokens to other objects. Output places are represented graphically by circles with a triangle inscribed.

The type of input and output places is the type of tokens exchanged with other objects and they are called communication token types which are defined in special communication units.

Tokens are structured data whose type is defined as a record type. Tokens with a void data structure are given the standard type *NUL*. Each place can contain several tokens at a time, but they must all be the same type.

Places are basically queues of tokens, and the queue type may be FIFO (first in first out), LIFO (last in first out), token delay (i.e. tokens put into this kind of place are ordered in time units on the basis of their token “delay”) or any other priority rule. Queue type of places can be changed by adding some C/C++ code to the transition action which produces tokens.

Transitions are the processing units of the model. They act on their input and output places. A transition models a token-driven computation, where the activation

of the computation and the propagation of the information are established by transition's graphic context. For an ordinary transition, the necessary condition for a transition to fire is that each of its input places contains at least one token. And when it fires, it removes one token from each of input places and adds one token to each of its output places. However, it is often necessary to specify a more flexible policy to allow tokens that a particular condition to be selected from the input palaces of the transition. For this purpose, an explicit condition called *predicate* can be specified to the transition and a priority level can be established to the produced tokens. Moreover, an action can be defined that the transition will perform on the data stored in the tokens.

Predicate is an optional Boolean condition that we can associate with a transition. It is evaluated on the tokens contained in a specified subset of the transition's input places. If the predicate is *true* then the transition fires; otherwise a new tuple of tokens will be selected and the predicate re-evaluated until all tuples have been tested or the transition has fired.

Priority is an optional non-negative integer that can be associated with a transition so as to give it precedence over all transitions with a lower priority. By combining predicated and priorities, it is possible to describe common behaviors like if-then-else, conditional loops, and switches.

Action is an optional block of C/C++ statements that can be associated with a transition. Such statements are executed when the transition fires. The action has access to the data stored in the tokens fetched from the transition's input places and can write data onto the tokens that will be released to the transition's output places. This activity may be as simple as a state transition or also very complex, if the transition is associated with a large piece of C / C++ code.

A transition is represented graphically by a rectangle (vertical or horizontal) which is labeled with its name and optionally priority number. A transition with a predicate defined has a small diamond in its icon. A transition with an action defined

has a thick line on the side of its icon. A transition has two further timing attributes: the firing delay and the release delay. Firing delay is defined as the delay from the instant at which the transition is enabled to the instant when it fetches the tokens from its input places. Release delay is defined as the delay from the instant at which the transition fetches the tokens from its input places to the instant when it releases the tokens in its output places.

An ***arc*** defines a token path between two graphical elements and is represented by an oriented line. A place which is a both input and output place of the same transition are often connected with a double arrow arc that represents simultaneously the incoming and the outgoing arc.

In the predicate section of some transitions, a statement which is employed to represent the inhibitor arc of PNs is used. These transitions can only be fired, when the input places which are connected with inhibitor arc to the relevant transition has no token. Since this property of inhibitor arcs are represented by predicates, these arcs are not graphically displayed on the class models.

A ***link*** is a connection between two objects. *Links* are the communication channels which can be used to have the objects exchange information. Each *link* defines a path for tokens that are moved from on output place of an object to an input place of another object.

A ***linkset*** is a set of *links* that connect two *portsets* in two different objects, and it is symbolized by a line that joins two object icons. A *linkset* represents the logical connection between two objects regardless the number of links that are necessary to establish the communication. A ***portset*** is an ordered set of input places and output places from the interface of a class. These places are used together to manage a related set of messages sent or received from another object. *Portsets* can be thought as to connectors which are used to bind many “cables” at once. Each cable is a link, and the whole set of links that connects two objects is called *linkset*. Figure 4.7 shows the graphical representations of the Artifex elements.

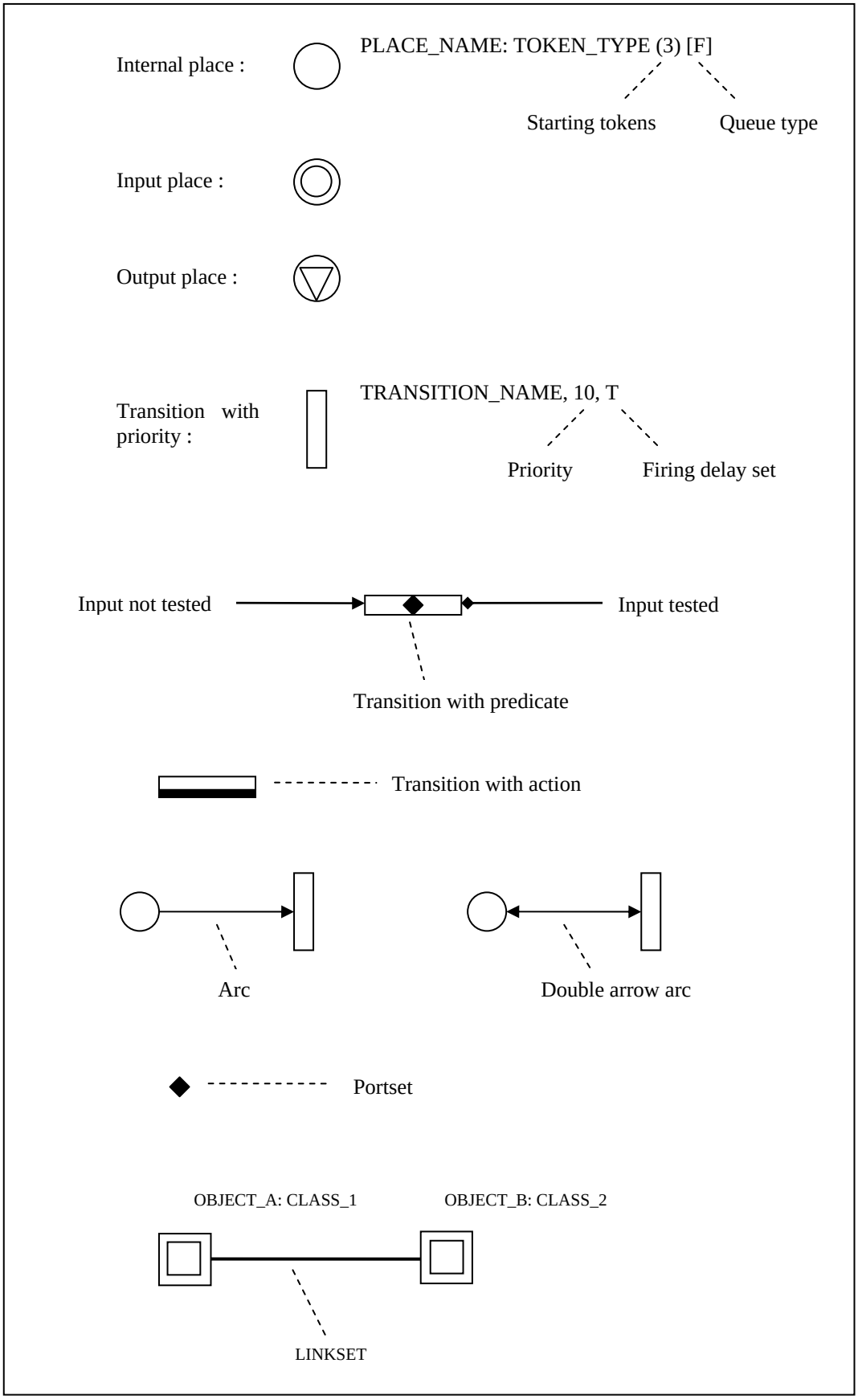


Figure 4.7 Graphical representations of the Artifex elements

4.4.1 CPN Token Color Definitions

As mentioned before, tokens of classic Petri Nets are mere markers used to highlights active states of a system, and standard Petri nets have been extended to the *High-Level Petri nets*. Thus, a PN allows arbitrary complex data types for tokens and let expressions for places, arcs and transitions be constructed with some special programming languages belong to the class of *High-Level Petri nets*. In particular, the Artifex language associates a data type with tokens, and transitions can read and write data contained in tokens. Each place of an Artifex model is given a type, which defines which type of tokens can be queued into that place. Thus places are data warehouses that store information until some transition will need it. For this purpose, some C/C++ code is added to transitions as an action of the transition, and to places as some extended properties.

In this manner, tokens are typed objects that represent the data and information managed by the system. Part, pallet, transpallet, operator flows in OMT models can be transformed into token color classes in CPN models. And based on the associations between each individual FMS object in OMT diagram, control messages and information flow are modeled by colored tokens. Token type definitions are as follows:

- **PART**: the physical flow of parts in the system is described by a token type PART. All part data such as Job No, Part Type, Number of Operation, Due Date, and Priority can be defined as attributes of the token type PART.

- **T_CONTROL**: part transport control messages between load/unload station and workstations by transpallet are described by a token type T_CONTROL. Such as transpallet request and acknowledgment are represented by a token type T_CONTROL.

- **O_CONTROL**: represents operator control messages between workstations and operator class such as operator request and acknowledgment.

- **WORKSTATION**: represents physical entities of machines at workstations. All machine data such as machine status and setup information can be defined as the attributes of the token type WORKSTATION.

- **R_CONTROL**: is used to represent part flow control messages within the system such as route request and route reply messages.

- **PART_TYPE:** is used to represent the control messages of the machine setup status within a workstation or between workstations and scheduler class.

- **TRANSFER:** is used to represent part routing control requests which could not be responded before in scheduler class.

- **L_U_STATION:** represents the available unload buffer capacity in Load/unload station class.

Token type definitions used for material and information flow are presented in Figure 4.8.

Token PART
{
int Order_ID;
int Order_size;
double Due_time;
double Order_Arrival_Time;
int Pallet_ID;
int Product_type;
int Operation_no;
int Route_no;
xx_address addrs;
xx_address destn;
double Total_process_time;
 	
}
	Token R_CONTROL{
	int Pallet_ID;
	double Due_time;
	int Product_type;
	int Operation_no;
	int Route_no;
	int Control_no;
	double Critical_ratio;
	xx_address source;
	xx_address destn;
	int Process_plan_info;
	int Alternative_route_num;
	int Total_operation_num;
	double Total_process_time;
	}
		Token T_CONTROL
		{
		int T_Pallet_ID;
		xx_address addrs;
		int Source_Stn;
		int Destination_Stn;
		double waiting_time;
		double transfer_time;
		
		}

Token O_CONTROL
{
int source_WS;
int destination_WS;
double waiting_time;
xx_address source;
xx_address destn;
}
	Token P_TYPE
	{
	int Current_P_Type;
	int Last_P_Type;
	xx_address addrs;
	int ST_index;
	}
		Token WORKSTATION
		{
		int index;
		int Current_P_Type;
		xx_address addrs;
		}

Token L_U_STATION
{
xx_address addrs;
xx_address source;
}
	Token TRANSFER
	{
	int S_Pallet_ID;
	int Product_type;
	int Operation_no;
	int Route_no;
	int Control_no;
	}	

Figure 4.8 Token type definitions for material and Information flow

4.4.2 CPN Models of the FMS's Objects

The properties and behavior of FMS objects are modeled by internal and external data/attributes and operations/methods of their deriving classes. Internal places and transitions are used to model the operations and dynamic behaviors inside the object class. Input and output places are used to model the external data/attributes which will serve as an interface to parts flow and information and control flow between FMS objects. Then all the related PN class models are connected through the input and output places to obtain the complete model and the control logic of the system.

4.4.2.1 Load and Unload Station Class

In Table 4.4, data/attributes and operations/methods of the load/unload station class is presented. When a production order arrives at the load/unload station (i.e. the transition ORDERS_ARRIVE fires), raw parts (PART tokens) with the attributes such as their arrival time to the system, products type, due date, process plan, and processing time on each machine are delivered to the system by the transition DELIVER_RAW_PARTS. ORDERS and RAW_PARTS places have infinite capacity. Then the raw parts are set up on pallets by the LOAD_P_PALLET transition, if there is an available pallet and free operator. After the parts are loaded to the pallets, the pallets are transferred to the load storage buffer, and a route request for each pallet stored is sent to the Routing_Control Object of the Scheduler Class by the transition MOVE_P_STORAGE. Load storage buffer and the buffer where raw parts are loaded on pallet have finite capacity with one pallet. Each pallet is stored until its destination workstation is informed by the token incoming through the input place ROUTE_REPLY which is the output place of the Scheduler class, and a transpallet reply is received from the Transportation class by the input place TRANSP_REPLY. If there is an available active buffer, the pallet which has its destination station is moved from the load storage buffer to the active buffer, then is sent to its next station through the output place READY_P_PALLET. Once all operations of the part on a pallet are completed, the pallet is sent back to the load/unload (L/UL) station to be unloaded from the pallet and stored the output

buffer of the system. The finished products arrive through the input place P_TRANSP_ARRIVE and received by the RECEIVE_F_P transition, if there is available active buffer. Then they are transferred to unload storage buffer by the transition STORE_P. The capacity of unload buffer is 50 pallets. Then the finished products are unloaded from the pallet by the operators, thus pallets become available and are sent to the free pallet buffer to be used again. Two operators work in Load/ unload station. Finally finished products are moved to the finished product storage area (FINISHED_PRODUCTS place) which has infinite capacity, and the relevant statistical performance measures are collected for each product by the transition SEND_F_PARTS. Figure 4.9 shows PN Model of Load and Unload Station Class.

Table 4.4 Data/attributes and operations/methods of the load/unload station class

Data / attributes	**Operations /methods**
Internal data / attributes - Number of pallets in the system - Number of operators in the Load/Unload Station - Load storage capacity - Unload storage capacity - Active buffer capacity for raw parts - Active buffer capacity for finished products ***External data / attributes*** - Number of product types - Process plan data of each product type - Order batch size - Order inter arrival time - Probability distribution of order due dates - Probability distribution of the product types	- Generating product orders, - Delivering raw parts into the system, - Loading raw parts to the pallets, - Transfer the pallet to the load storage buffer, - Send a route request for each part to the Scheduler Class, - Receive the route reply, - Receive the transpallet reply, - Receive finished products from the workstation class, - Send free transpallet information to the part Transport object, - Transfer the pallet to the unload storage buffer, - Unload the finished product from the pallet, and release free pallet, - Collect the statistical performance measures for each finished product.

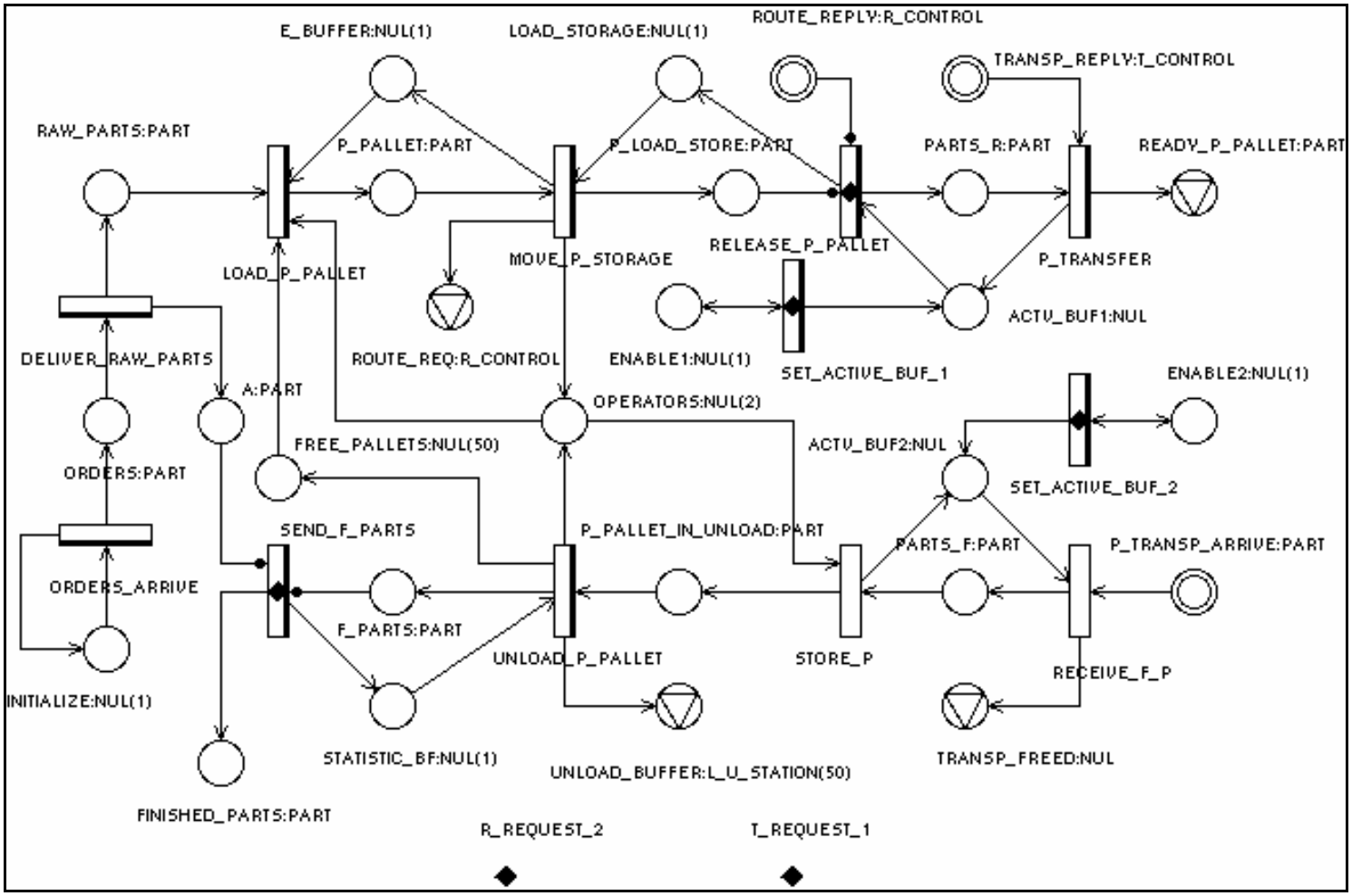


Figure 4.9 PN model of Load/Unload Class

4.4.2.2 Workstation Class

Table 4.5 shows a summary of data/attributes and operations/methods of the workstation class. Parts are processed at machining stations. Once a transpallet arrives at a workstation through the input place P_TRANSP_ARRIVE, the pallet is moved to the input buffer of the workstation by the P_PALLET_RECEIVE transition and the transpallet becomes free by adding a token to the output place TRANSP_FREED. If machining station is idle, the part is transferred from the input buffer to the machining buffer, and one operator request is sent to the Operator Object by the P_PARTS_UNLOAD transition. Since the machines in the system need setup operation to process different types of product, setup status of the machine and the product types of both new and previous parts should also be controlled in the Workstation class. Hence, an attribute is attached to the tokens which represent machines and this attribute can save the information which type of product was processed by the machine last. So a major data abstraction is included to the tokens in the PN model. Once a part is received to the machining buffer of a workstation, the current product type information attached to the arriving token in the PARTS_TYPE place is compared with the last product type information on the token in the place LAST_P_TYPE. If they are the same type of products, the release time of the transition MCHN_SETUP is taken as zero, otherwise the machine is set up by the operator, and the release time of the transition becomes the setup time of the machine. After the machine is ready for the operation, the new part is loaded to the machine by the P_LOAD transition. The timed transition P_PROCESSING represents the operation of the parts on the machine. Once the processing of the part finishes, it is unloaded from the machine by the P_UNLOAD transition, then transferred from the machining buffer to WIP storage buffer (i.e. the output buffer) of the workstation (P_PALLET_OB), and a routing request is sent to the Part Routing Control object through the output place ROUTE_REQ. While the machining buffer capacity is one pallet, the capacity of WIP storage buffer is ten pallets. When a routing request is replied for a pallet (ROUTE_REPLY), the ready pallet is transferred to the active buffer (P_PALLET) which has the capacity of one pallet,

and it is sent to the destination station by a transpallet through the output place READY_P_PALLET. In Figure 4.10, PN Model of Workstation Class is presented.

Table 4.5 Data/attributes and operations/methods of the workstation class

Data / attributes	**Operations /methods**
Internal data/attributes - Machine status (idle/busy) - Machine setup status - Machining buffer capacity - Buffer stand status - WIP storage (output) buffer capacity - Part status - Operator availability - Operator request - Route request - Number of machines ***External data/attributes*** - Parts arriving at workstation - Operator acknowledge - Operator reply - Route reply - Transpallet reply	- Receive a pallet to the input buffer, - Check the availability of the machine, - Check the setup status of the machine, - Check the operator availability, - Send an operator request, - Setup the machine, - Shuttle a job between the machining table and buffer stand, - Process an operation sequence, - Update the setup status of the machine, - Move the job to the output buffer - Release the operator - Send a route request to the scheduler class - Transfer the job to its next destination

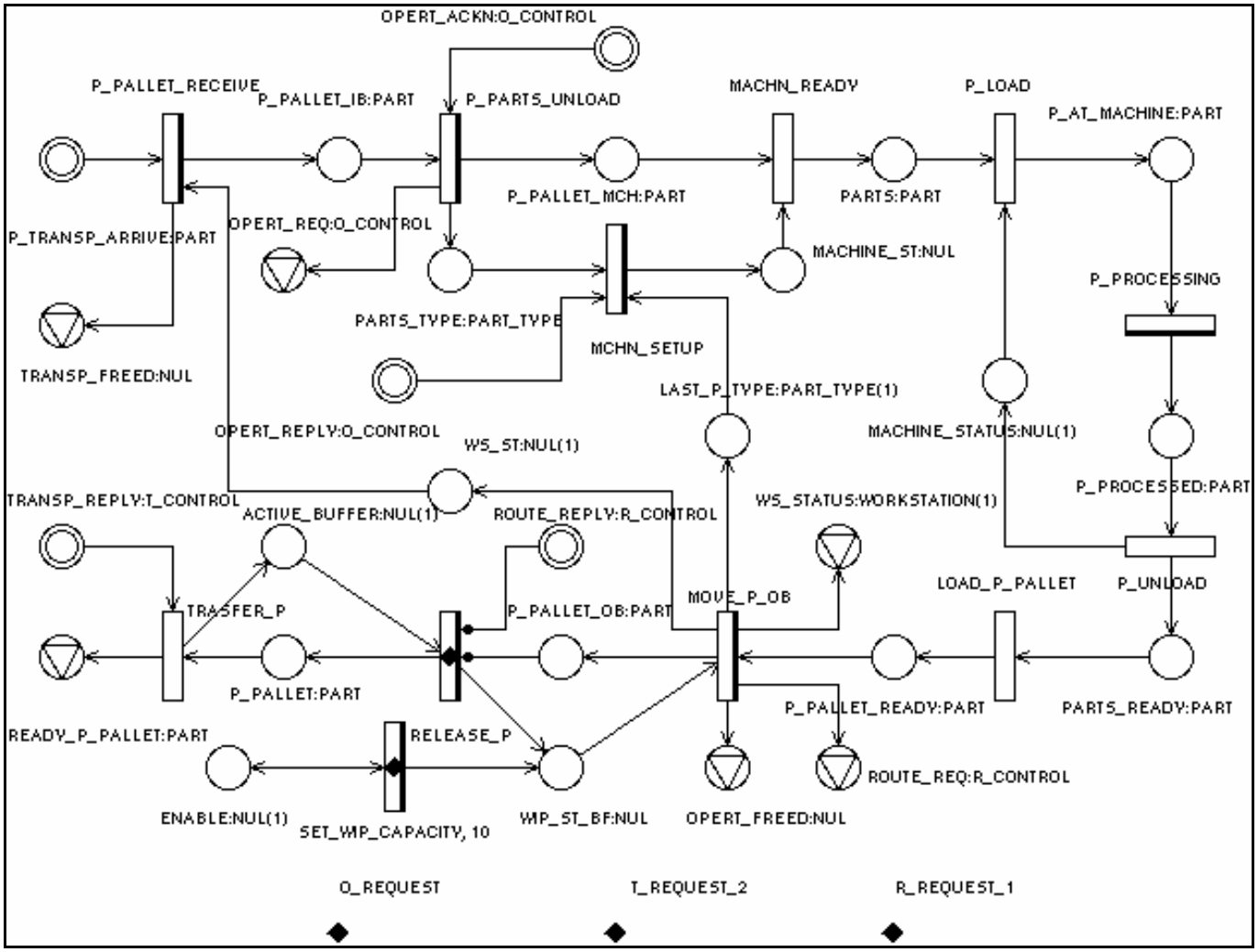


Figure 4.10 PN model of Workstation Class

4.4.2.3 Transportation Class

In Table 4.6, a summary of data/attributes and operations/methods of the Transportation Class is presented. Parts are transported by the transpallet among workstations and between load/unload station and workstations. Input place TRANSP_REQ represents a part transport request sent by the Scheduler class. The Part Transport Object accepts a transport request by the transition ACCEPT_T_R, and a transpallet is sent to the Load_unload station class or Workstation class where the ready pallet is waiting to be routed its next destination by the output place TRANSP_REPLY, if the internal place FREE_TRANSP has at least one token (i.e. there is a free transpallet in the system). Then, the ready pallet is loaded to the transpallet, and transferred to the destination station by the transition LOAD_P which releases a token to the output place P_TRANSP_ARRIVE. A token in this place is sent to the input place of the destination object which is connected by a link. It is made possible by having a unique address associated with any object of the model, and token routing mechanism. Once a transpallet arrives to a workstation, part is unloaded and free transpallet is sent back to the Transportation Class, it is then allowed to reply another part transport request of Scheduler class. Number of transpallet parameter in the system is represented by the number of tokens in the FREE_TRANSP place and is set by the transition SET_TRANSP_CAPACITY and ENABLE place. The transition SET_TRANSP_ CAPACITY fires and puts tokens as much as the number of transpallet parameter. Transport object has links between machines, load/unload object, and part routing control object through portsets to provide them with the transpallet availability. Figure 4.11 illustrates the PN Model of Transportation Class.

Table 4.6 Data/attributes and operations/methods of the Transportation class

Data / attributes	**Operations / methods**
Internal data / attributes - Number of transpallet in the system - Transpallet availability ***External data / attributes*** - Transpallet request - Source station - Destination station - Ready part for transportation - Transpallet returned (free transp.)	- Receive transpallet requests from the Scheduler Class, - Accept a transpallet request if there is an available transpallet (FIFO), - Send the transpallet to the source station, - Load the part to the transpallet, - Send a transpallet acknowledgement to the Scheduler class, - Transfer the part to the destination station, - Receive the free transpallet information.

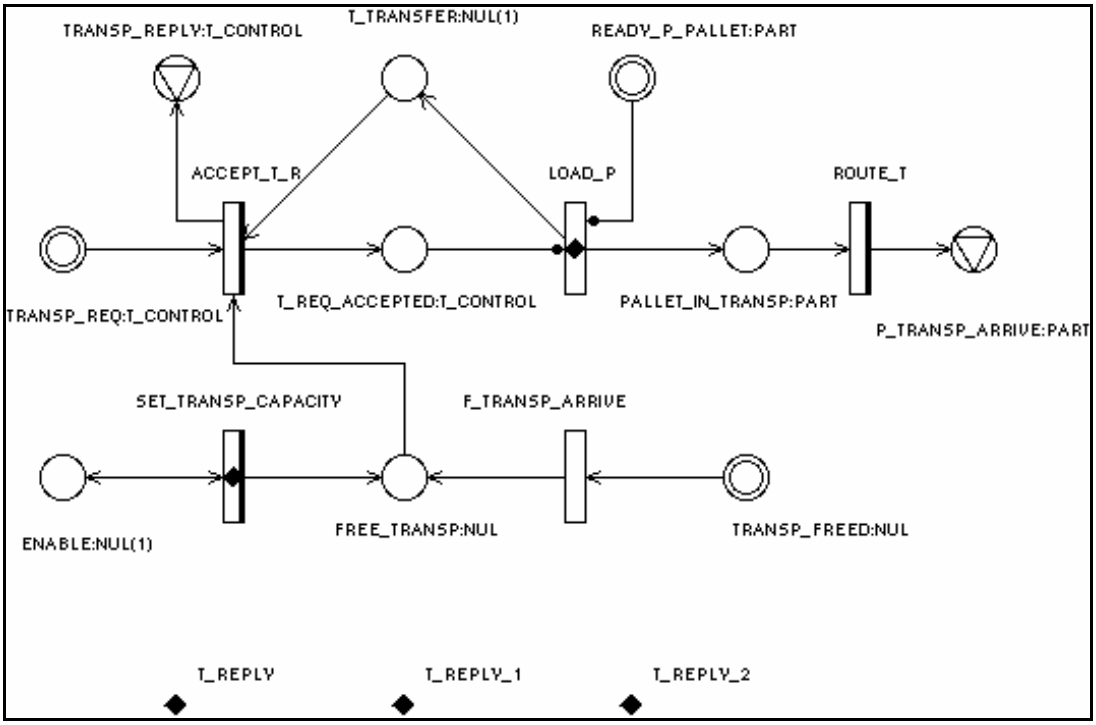


Figure 4.11 PN model of Transportation Class

4.4.2.4 Operator Class

Table 4.7 summarizes the data/attributes and operations/methods of the Operator Class. Machine setup operations, loading and unloading the parts to/from the machines are performed by the operators in the system. The operators which work for workstations are cross-trained thus have the same skill level. So they are assumed highly flexible and capable of processing jobs in any workstations without loss of efficiency. Input place OPERT_REQ represents an operator request sent by any object of the Workstation Class. Operator object accepts an operator request by the transition ACCEPT_OPERT_REQ, and releases a token to the ACCEPTED_O_R place, if there is a free operator, i.e. the internal place FREE_OPERT has at least one token. Then, operator goes to the requesting station by the transition SEND_OPERT which adds a token to the output place OPERT_AT_WS, and an operator acknowledgement is also sent to Workstation class by the output place OPERT_ACKN which is used to inform the requesting object of the Workstation class, that its previous operator request has been responded, and it is then allowed to send another operator request to the Operator class. Number of Operator parameter is modeled by the number of tokens in the place FREE_OPERT and is set by the transition SET_OPERT_ CAPACITY. Once the system starts its operation, transition SET_OPERT_ CAPACITY inserts tokens as much as the numbers of operators in the system. The operators which have finished the operation and becomes free, is informed to the Operator class by the input place OPERT_FREED. Operator object has a link between machines through the portset O_REPLY to provide them with the operator availability. Figure 4.12 shows the PN Model of Operator Class.

Table 4.7 Data/attributes and operations/methods of the Operator Class

Data / attributes	**Operations /methods**
Internal data / attributes - Number of operator in the system - Operator availability - Operator skill level ***External data / attributes*** - Operator request - Source station - Free operator information	- Receive operator requests from the Workstation Class, - Accept an operator request if there is an available operator (FIFO), - Transfer an operator to the source station, - Send an operator acknowledgement to the Workstation class, - Receive the free operator information.

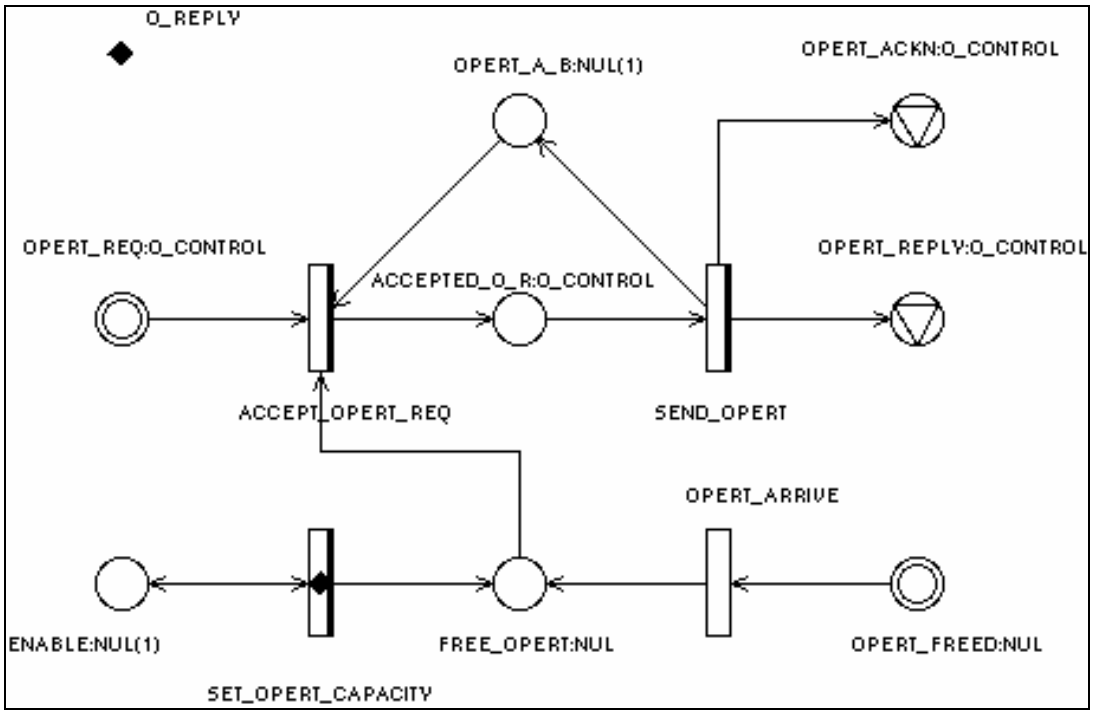


Figure 4.12 PN Model of Operator Class

4.4.2.5 Scheduler Class

An important issue during the execution process of a manufacturing system is to ensure that all needed resources for a task are available at the right place and the right time. In a manufacturing system, concurrent flow of parts competing for sharing limited resources causes resource contention problem. Resource sharing in any manufacturing system often leads to conflicts. The system controller must be capable of resolving these conflicts effectively. From PNs view of point, resource sharing increases the complexity in scheduling. The system control and scheduling approach in proposed DSS use a heuristic rule based approach to solve resource contention problems and to determine the best route(s) of the parts, which have routing flexibility. The system aims to solve the part routing control problem which includes material handling, operator and machine setup operation constraints, and thus also to solve resource sharing problem effectively at the same time. The proposed approach is explained with the help of the PN model of the system by the sequence of execution of transitions.

Part flow between stations in the system are controlled and managed by the Routing_Control Object of the Scheduler Class. All the information of each part which is ready to be transported in the system is sent to the scheduler class. In the scheduler class, the next destination station of a part is determined dynamically based on the existing shop-floor conditions. It also makes arrangements for transporter. Since the production environment is dynamic, the nature of the scheduling decision will change over time and therefore, the dispatching or sequencing rule will also need to change over time. Thus scheduling rules should be combined and consider the prevailing conditions of the shop floor. Since no iterative decision-making is required, scheduling decisions can be made in real-time, thus it would likely generate a quick response in industry.

The Scheduler class examines the state of the system at every discrete event which there was a change in the status of the system and makes a decision applying scheduling rules in the knowledge base then passes the decision to the other classes.

Thus it provides a support to the user so that the decision is taken after comparing the different available options of scheduling which are better in respect to the different aspects such as due dates, overhead costs, minimum tardiness, and flow time. A set of production rules in the form of "IF….. THEN….." statements have been constructed based on the heuristics developed for this work to assign the resources to the parts, and to determine the best route(s) of the parts thus to solve the resource contention problem. A production rule which is a means of expressing reasoning links between facts expresses the behavior of objects in the system of interest.

IF $(c_1, c_2, \ldots..., c_m)$ THEN $(r_1, r_2, \ldots\ldots..., r_{rn})$

where

the c_j (j = 1, 2, ……., m) are predicates known as conditions, and

the r_k (k = 1, 2,…….., n) are to as consequences.

A predicate checks the state of the system, such as the process plan of the parts, the number of conflicting part types, their remaining number of operations, their remaining process times, and the setup status of the machines, and selects the next part to be processed form a set of parts awaiting service according to some priority. When the IF portion of a production rule (predicate) is satisfied by the conditions, the action specified by the THEN portion is performed. When this happens, the rule is said to fire. In scheduler class, a rule interpreter compares the IF portion of each rule with the facts and executes the rules whose IF portions match the facts, as shown in Figure 4.13 (Kusiak, 2000). Each rule in this scheme corresponds to the use of a routing control strategy subject to the existence of certain conditions.

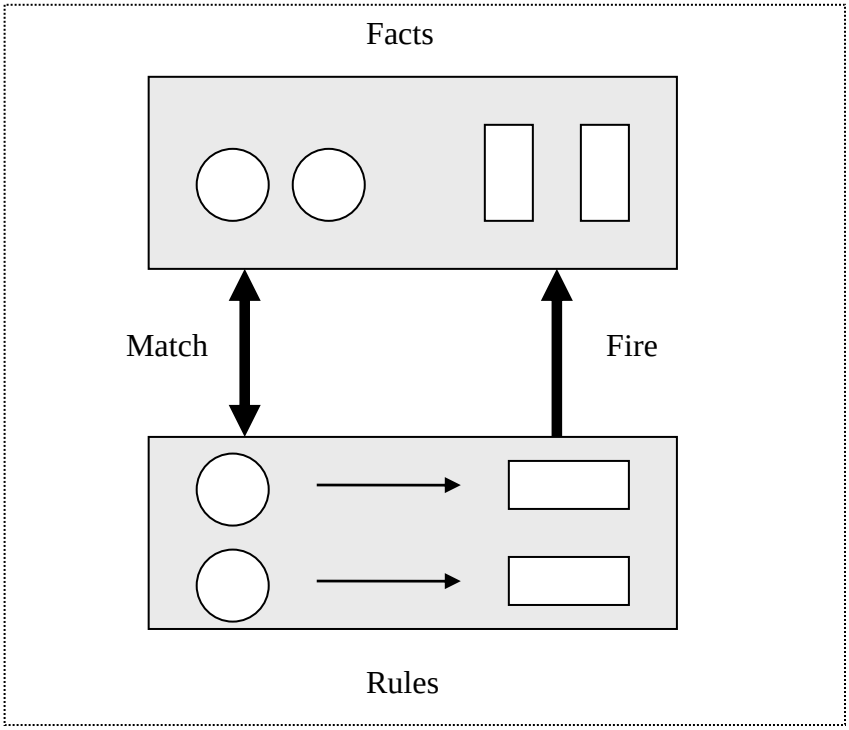


Figure 4.13 Execution of rules through a match-fire sequence (Kusiak, 2000)

In the IF…-THEN… statements, the following dispatching procedural rules are used in decision making process.

- ***First come, first served, (FCFS or FIFO),*** the sequencing of jobs is done according to the arrival order of the jobs. So, the job that has entered the queue at the earliest is chosen for loading. This rule is effective for minimizing the maximum flow time and variance of flow time.

- ***Earliest due date (EDD)*** sequences the jobs in an increasing order of their due dates. This rule is often used in industries for its simplicity of implementation in the shop floor. This rule performs well with respect to minimizing maximum tardiness and variance of tardiness in the case of single

machine scheduling problem (Rajendran & Holthaus, 1999). The priority index is used as follows:

$P_i = DT_i$

where P_i = Priority

DT_i = Due time of the task

- ***Shortest total processing time (STPT),*** the jobs are sequenced in an order of increasing total processing time of jobs.

- ***Smallest number of remaining operations (SNRO)***; the job sequence according to the increasing order of the remaining number of operations.

- ***Largest number of remaining operations (LNRO):*** the job sequence according to the decreasing order of the remaining number of operations.

- ***Shortest processing time, (SPT),*** this is a conventional sequencing rule that selects the job with the smallest processing time. Processing time does not include setup time. This rule is included in the heuristic rules, since it is effective for flow time related performance measures.

- ***JIS (Job of identical setup):*** This heuristic is a setup-conscious rule. It scans the queue for a job identical to that which has just been completed on the first alternate machine. When there is no identical job for all alternate machines, it computes the critical ratio defined as in CR. This rule intends to minimize setup time.

- ***Critical Ratio scheduling:*** This is a conventional sequencing rule, and requires forming the ratio of the processing of a job divided by the remaining time until the due date, and scheduling the job with the largest ratio next (Kim & Bobrowski, 1997). The idea behind the CR scheduling is to provide a balance between SPT, which considers only processing time, and EDD,

which considers due dates (Gupta, Sivakumar, and Sarawgi, 2002). The ratio will grow smaller as the current time approaches the due date and more priority will be given to those jobs with longer remaining processing times. This sequencing rule is effective for due date related performance measures.

$$CR = (DD - CT) / (RPT + RNO * QAPO)$$

where,

DD = due date,
CT = current time,
RPT = remaining processing time,
RNO = remaining number of operations,
QAPO = queue allowance per operation.

QAPO is the expected waiting time in each queue at the time of release, and is calculated as follows:

$$QAPO = (DD - RD - TPT) / TNO$$

where,

RD = release date of the job,
TPT = total processing time of a job,
TNO = total number of operations in a job.

The procedure for scheduling operations on the machines is executed whenever a raw part for a new order is delivered to the load storage buffer at Load/unload station or a workstation completes performing of its current operation and becomes available for the next task assignment. Part Routing Control object has links between load/unload station, machines and transporter through corresponding portsets. Thus, it will be announced the resource availability information from the integrated model, when they become idle. When a part is processed at a workstation, it is transferred to WIP storage area of the workstation which has a limited capacity and a route request

with the product information such as product type, due date, and process plan information of the part, is sent to the scheduler class for this part. Scheduler class is also informed about all system changes, such as the setup status of the machines and availability of the workstations. In scheduler class, all route requests that are sent from the workstations and load/unload station are put in order, and replied by considering the prevailing system conditions and the rule based system. Once a route request is replied, the destination of the pallet is informed to the station which sent that route request and a transpallet request is sent to the Transport class, by this way, if there is an available transpallet, it can be directed to the workstation which the part waiting to be transferred its next destination. The route requests which are not satisfied join a waiting queue, and once a new route request arrives to the Routing control object, all the route request including newly arriving one are retested in the new system status. This procedure is repeated until all route requests are replied. Figure 4.14 shows the PN Model of Scheduler Class.

There are several decision points in the FMS at which decision making process and dispatching rules can be applied. These decision points are as follows:

i) routing of the raw parts or semi finished products in the system
ii) inventory sorting in the local buffers
iii) raw material sorting in the incoming storage; and
iv) AGV selection of parts.
v) operator selection of workstations

According to the given heuristic rules, the part in the local buffers or storage location with the highest priority will be placed in the first position of the queue. Hence it will be the first one to be served for machine processing, or inputting to the system, or transportation.

The rule-based system for the manufacturing system under consideration is constructed using the following heuristics:

- Route requests are received according to the Largest Number of Remaining Operations (LNRO) and Earliest Due Date (EDD) sequencing rule. A highest priority is given to the product with largest number of remaining operations or the earliest due-date, depending on the scheduling criterion employed.

- Finished products waiting to be transferred to the unload buffer of the Load/unload station have the highest priority for transportation (Smallest Number of Remaining Operations rule, SNRO).

- Semi-finished or raw parts are sorted according to their remaining processing time with the Smallest Remaining Processing Time (SRPT) first, or their due dates with the Earliest Due Date is first (EDD) for the parts with the remaining processing time.

- Alternative machines are ranked according to their operation time with the Shortest Processing Time (SPT) first, and the pallet is routed to the available machine among the alternative machines which is ranked first and doesn't require setup operation (JIS).

- If there is no available machine, which doesn't need setup operation, the route request is accepted regarding the order's Critical Ratio (CR).

- After all the route requests are checked under the current system status, the route requests, which are not critical, are reevaluated to be routed.

The above–defined conditions are incorporated into the PN model like the following rule:

Rule 1:

The transition reads the product type, process plan, operation no and alternative route number of the part which sent the route request, and controls the

availability and setup status of the machines and load/unload station in the system, then the following predicate is checked;

IF {There is an available machine which can process the part AND If this machine does not need setup operation} AND {The previous part transport request has been responded, THEN {Route the part to the available workstation which have the shortest processing time} AND {Send a transpallet request to the Transport object}.

This rule can be presented in the model as follows:

IF

ROUTE_REQUEST->Process_plan_info == WS_STATUS->index) **AND**
ROUTE_REQUEST->Process_plan_info == SETUP_ST->ST_index) **AND**
ROUTE_REQUEST->Product_type == SETUP_ST->Last_P_Type)

THEN

xx_address_cpy(&(ROUTE_REPLY->destn),&(WS_STATUS->addrs));
xx_address_cpy(&(TRANSP_REQ->addrs),&(ROUTE_REQUEST->source));
xx_setaddress(XL_ROUTE_REPLY,&(ROUTE_REQUEST->source));
TRANSP_REQ->Pallet_ID=ROUTE_REQUEST->Pallet_ID;

Figure 4.15 illustrates the flow diagram of part routing control process. This algorithm attempts to further reduce the mean flow times of the jobs by reducing the setup times incurred while the product types change.

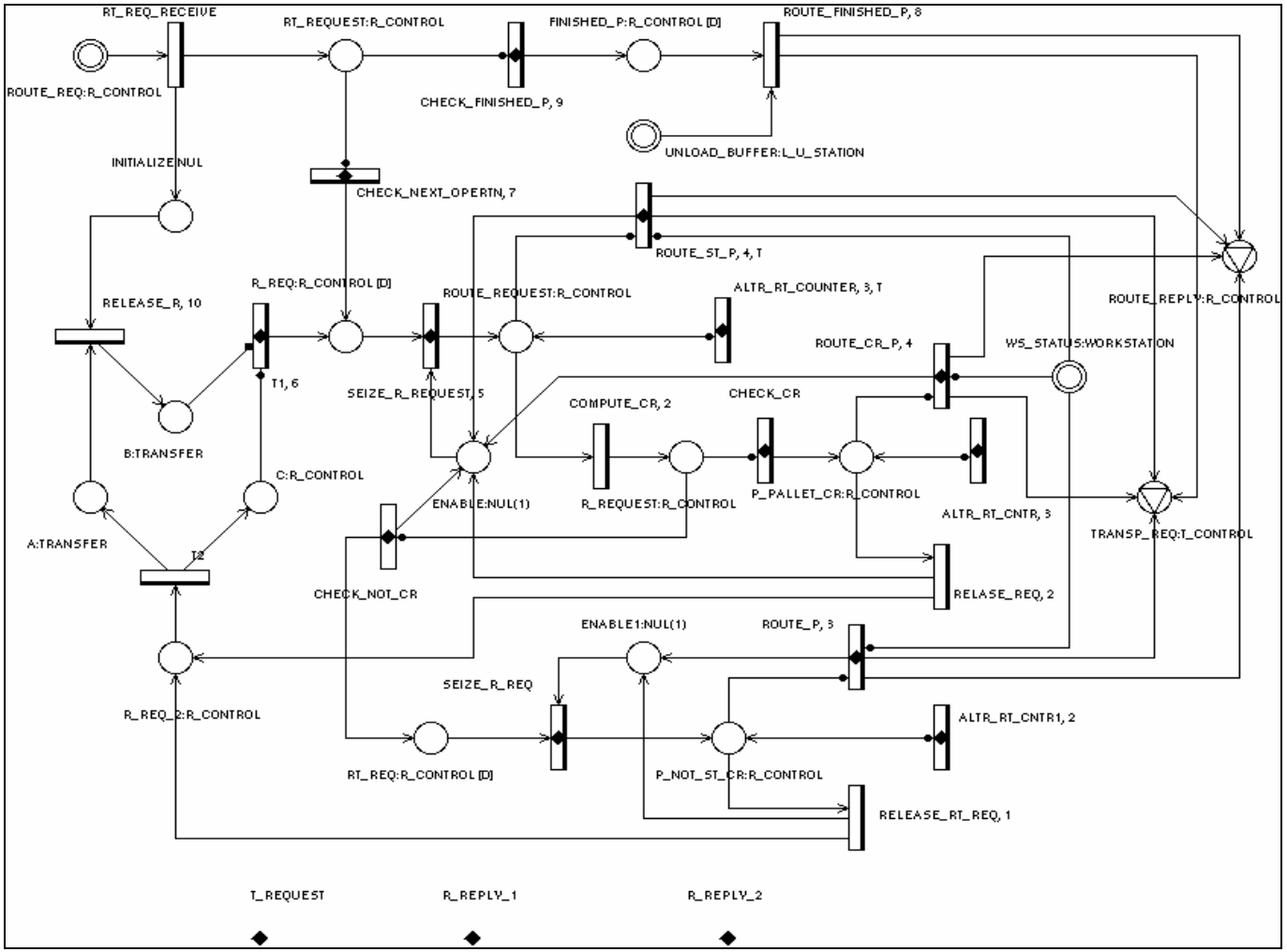


Figure 4.14 PN model of Scheduler Class

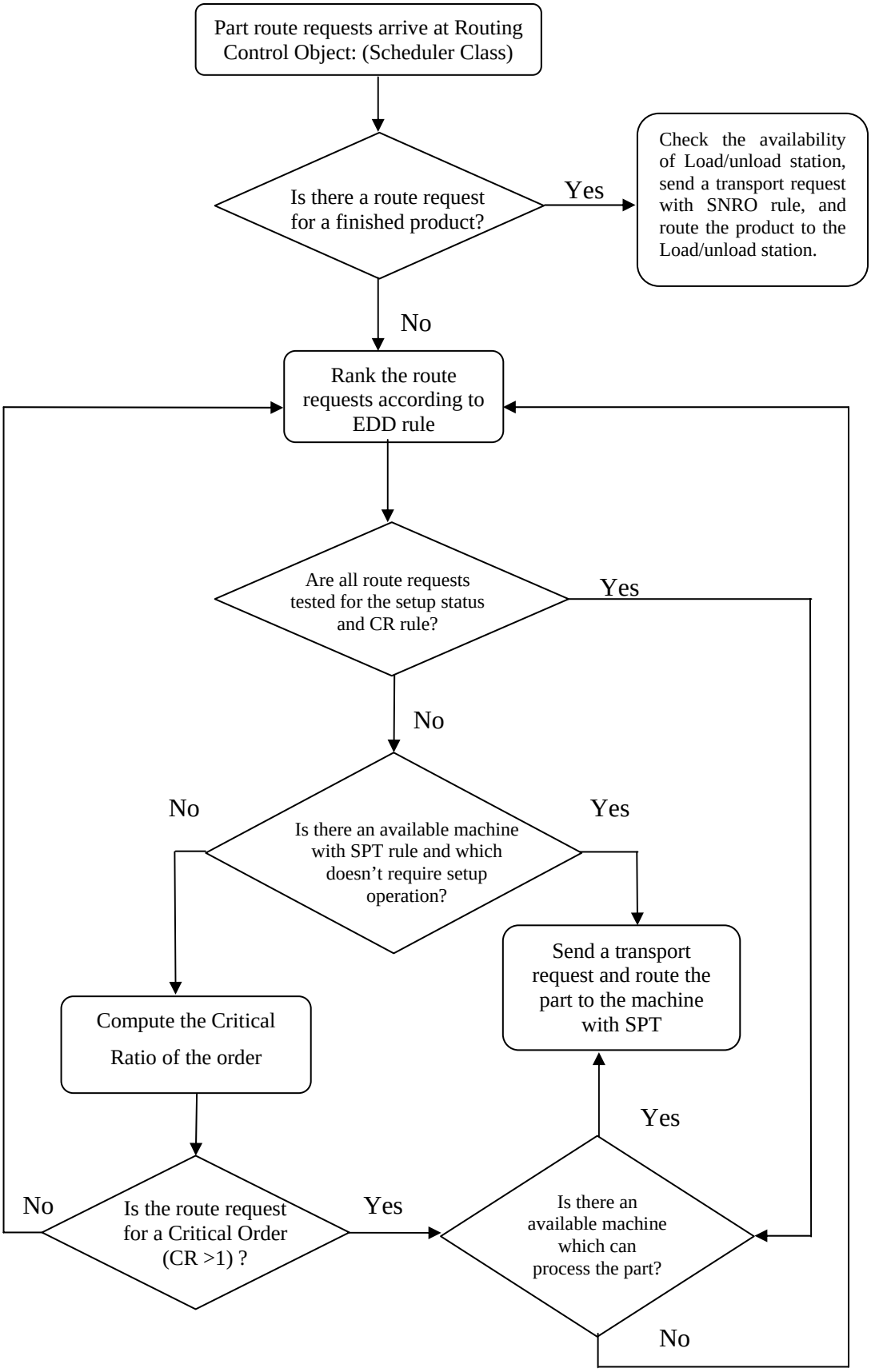


Figure 4.15 Flow chart of the Routing Control Process

4.4.3 Integrating the Objects of the Manufacturing System

As stated above, the objects are the instance of their corresponding classes defined by developing the OMT diagram, and each object has some input and output places to represent interface of these objects which provide their communication with other objects by sending or receiving tokens. All the FMS objects are integrated to obtain a complete model of the system by linking the corresponding input and output places of each object. A set of input or output places is ordered together to manage a related set of token types. Interacting objects in FMS are connected through creating “link set” which involves bidding protocols between objects. The interactions between the system entities can be considered as a client-server paradigm. The marking condition of the integrated PN model indicates the current system status. The obtained model can provide a thorough understanding of the dynamic behavior of the system as well as assisting the evaluation of various system operational strategies. We can manipulate the different input parameters such as number of operators, transpallet, arrival rate, or distribution, and so on to study their effects on the system and to evaluate performance of the rule based system and different scheduling rules.

Figure 4.16 shows the integrated model of the complete FMS. Each line represents the link set between the FMS objects.

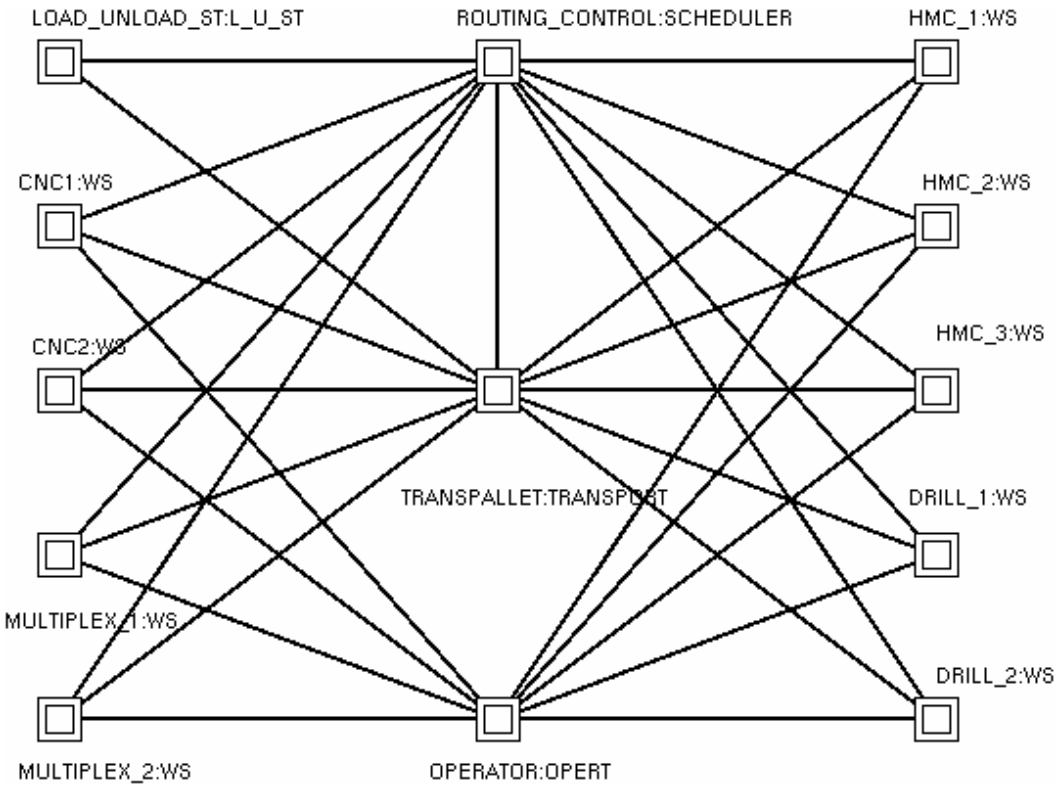


Figure 4.16 Integrated model of the proposed DSS

4.5 Performance Evaluation of the Dynamic Scheduling

After the CPN model of the FMS is completed, a high-level Petri net based simulation analysis is performed for the performance evaluation of the part routing, and resource allocation strategies under different levels of system parameters such as the number of operators and transpallet in the system. The input data to the PN models consists of part types to be processed, machines, load/unload station, material handling device, and operators.

Input data for the performance analysis of the example FMS are as follows:

PARTS:

Parameter	***Value***
Number of product types	3
Order batch size	1
Product type arrival ratio	% 30, Product type = 1 % 40, Product type = 2 % 30, Product type = 3
Order inter arrival time	Exponential with mean 30 min.
Process plan	Totomak Main Bearing Production Department
Due date	Arrival time + (100+ uniform (0;2)* max.total processing time)

MACHINES:

Parameter	***Value***
Number of machines	9
Machine Setup Time	20 min.
Loading/unloading parts to/from machines	0.2 min.

LOAD/ UNLOAD PROCESS:

Parameter	***Value***
Loading / unloading parts to/from pallets	0.3 min.
Moving pallets to load / unload storage buffers	0.2 min.

TRANSPALLET:

Parameter	***Value***
Number of transpallet	1
Transfer time between the stations	1.5 min.

OPERATORS:

Parameter	***Value***
Number of Operator	8
Operator transfer time between workstations	0.5 min

4.5.1 Performance Measures

There are numerous objectives in scheduling. Therefore, there are dozens of sensible performance measures. As the objective function of scheduling problems in practice, due date related measures such as mean tardiness or the proportion of jobs tardy is more critical as they reflects the customer satisfaction. Besides, the most commonly used performance measure of turnaround in a manufacturing system has been mean flow time of parts, as it is closely related to other common performance measures (Baker, 1974). Indeed, minimizing mean flow time decreases mean number of jobs in the system (i.e. work-in-process inventory) and waiting times of the parts in the system which result an increase in throughput rate and the productivity of the system. Consequently, for evaluating the performance of different generated schedules, the following performance parameters are taken into consideration; mean job flow time through the system, number of tardy jobs, average tardiness of the jobs, proportion of tardy jobs, and number of machine setup operations.

A task T_j can be characterized by the following data:

p_{ij} = processing time of operation j on machine M_i,

r_j = the arrival or release date of task j. This is the time at which task T_j is ready for processing,

w_j = the weight of task T_j. The weight expresses the relative urgency of task T_j,

d_j = due date , that is the promised delivery time of task T_j.

Mean flow time: Mean flow time of each part is calculated as the sum of flow times of all parts divided by the total number parts. The flow time of the part j is the

time elapsed from the initiation of the part *j* on the first machine to the completion time of job *j;* equivalently it is the amount of time that job *j* spends in the system. The rules that perform well in terms of flow time criteria will also show improvements in average waiting time, work-in process inventory measures.

The weighted flow-time of a task T_j (F_{wt}) is the between the release time and the completion of T_j, ie.

$$F_j = C_j - r_j$$
$$F_{wt} = w_j(C_j - r_j) = w_jC_j - w_jr_j$$

where

C_j is the completion time at which task *j* is completely processed,

Mean Flow time is $\overline{F} = 1/n \sum_{j=1}^{n} F_j$ or

Mean weighted flow time is $\overline{F}_{wt} = \sum_{j=1}^{n} w_j F_j \Big/ \sum_{j=1}^{n} w_j$

n = the total number of jobs.

Lateness is the difference (positive or negative) between the completion time and the due date of a job. *The due-date* d_j of a task T_j, is the time at which task T_j should be completed.

$$L_j = C_j - d_j$$

A weighted lateness is;

$$L_{wt} = \Sigma\, w_j(C_j - d_j) = \Sigma\, w_jC_j - \Sigma w_jd_j = C_{wt} - d_{wt}$$

where

d_{wt} is the weighted combination of due dates.

Maximal lateness $L_{\max} = \max \{ L_j \}$

Tardiness is the positive difference between the completion time and the due date of a job. A tardy job is one that is completed after its due date.

The tardiness T_j measure only considers "tardy" tasks, i.e. $T_j = max \{C_j - d_j, \ 0\}$.

Mean Weighted Tardiness is calculated by summing average tardiness of jobs that are charged by the weights.

Mean Tardiness is $\overline{T} = (1/n) . \sum_{j=1}^{n} T_j$

Mean Weighted Tardiness is $\overline{T}_{wt} = \sum_{j=1}^{n} w_j T_j \Big/ \sum_{j=1}^{n} w_j$

Percentage of Tardy Jobs defines the ratio of number of tardy jobs, to the total number of jobs.

The number of tardy jobs, $N_T = \sum_{j=1}^{n} N_j$

where

$$N_j = \begin{cases} 1 & if \ C_j > d_j \\ 0 & otherwise \end{cases}$$

Proportion of Tardy Jobs (%) $= 100 \cdot (N_T / n)$

n = the total number of jobs completed.

4.5.2 Performance Analysis Results

Ten independent replications for each dispatching strategies were run for 180.000 operating minutes (125 days with three 8 hr shifts) during the simulation of the CPN model. In each run, the shop is continuously loaded with job-orders that are numbered on arrival, and each run produced one observation for each performance measure. Different random number seeds were used to prevent correlation between the parallel runs of the factorial experiment. In order to ascertain when the system reaches a steady state, the shop parameters, such as mean flow time of jobs and utilization level of machines, were observed, and it has been found that the system became stable after a warm-up period of 43.200 simulated minutes (30 days with three 8 hr shifts). Thus, for each replication, the first 43.200 minutes was discarded to remove the effect of the start-up condition, which was an idle and empty state.

Performance of the proposed rule-based system was compared with some dispatching rules such as earliest due date (EDD), first in first out (FIFO), largest number of remaining operations (LNRO), smallest number of remaining operations (SNRO), and critical ratio (CR) with respect to mean flow time of jobs, mean tardiness, percentage of tardy jobs, number of tardy jobs, and number of total setup operations of the machines. In these cases, the jobs waiting for the next operations in the central buffer of Routing Control Object are ranked by only considering the selected dispatching rule, and the job which is ranked first is routed to the available machine which can process part with the smallest processing time (SPT) among the alternative process plans. Therefore, we only make modification on Scheduler Class model for comparison, and other object models are used in the same way.

Table 4.8 summarizes the performance analysis results of the proposed rule-based system and different dispatching rules. The results presented are obtained by taking the mean over 10 replications for each procedure. We compare the obtained results in Figures 4.17 through 4.21 in terms of the performance criteria.

Table 4.8 Performance analysis results of the different dispatching rules and proposed rule-based system

	EDD	**FIFO**	**SNRO**	**LNRO**	**CR**	**Proposed Rule–based System**
Mean job flow time (min)	133.12	134.13	141.71	138.38	133.92	115.04
Mean Job tardiness (min)	28.58	26.11	34.76	32.19	27.49	19.74
Proportion of Tardy Jobs (%)	48.6	55.6	53.3	51.8	50.9	32.3
Number of tardy jobs	2225	2577	2460	2366	2333	1499
Total number of machine setup operations	11186	11385	11552	11366	12573	9650

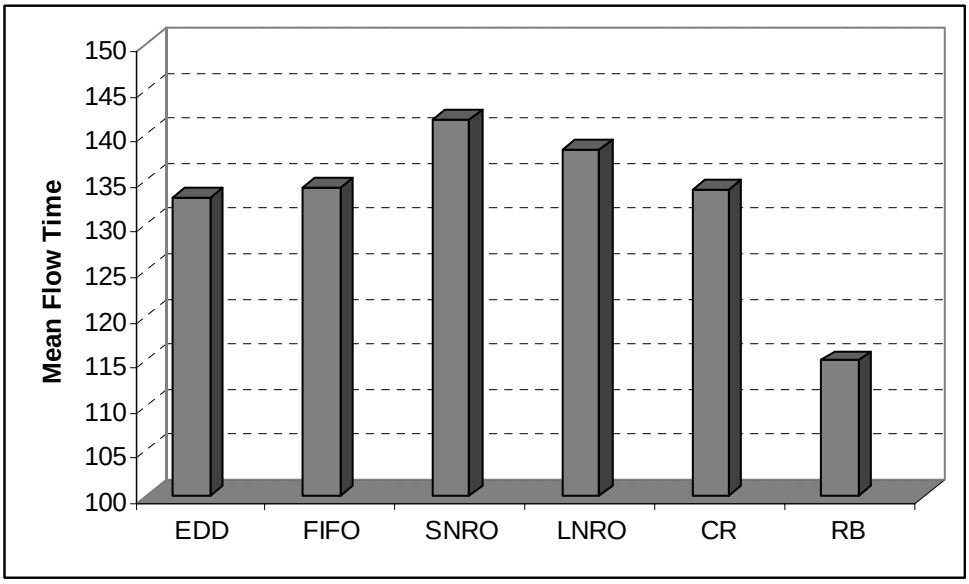


Figure 4.17 Mean job flow time comparison

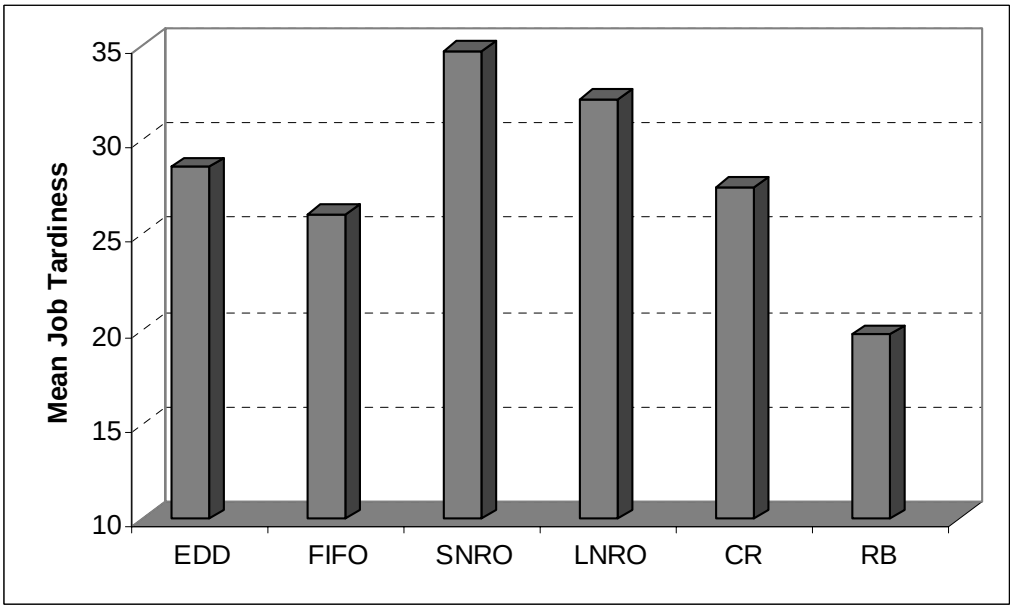


Figure 4.18 Mean job tardiness comparison

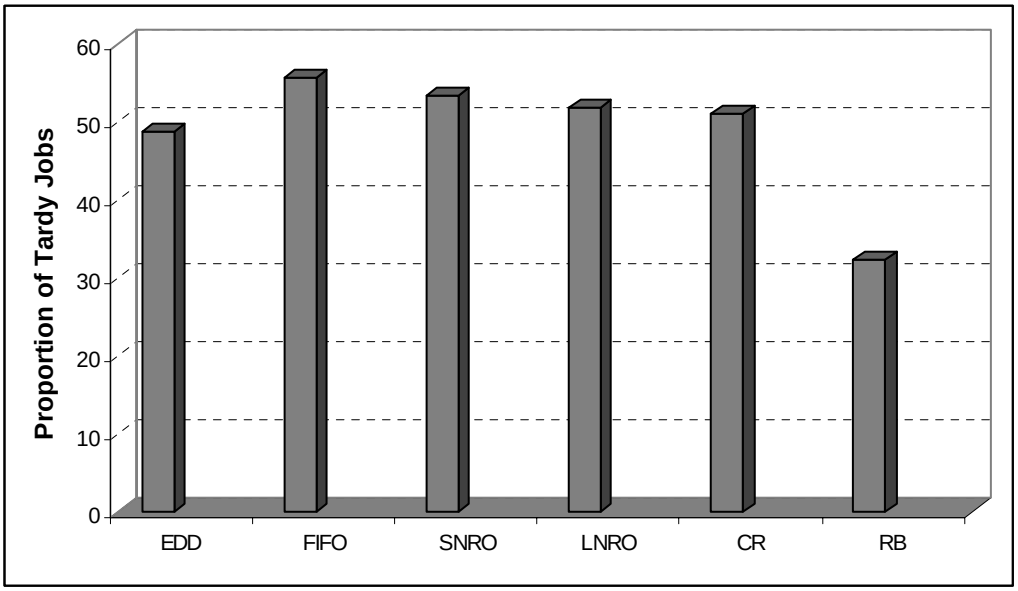


Figure 4.19 Proportion of tardy jobs comparison

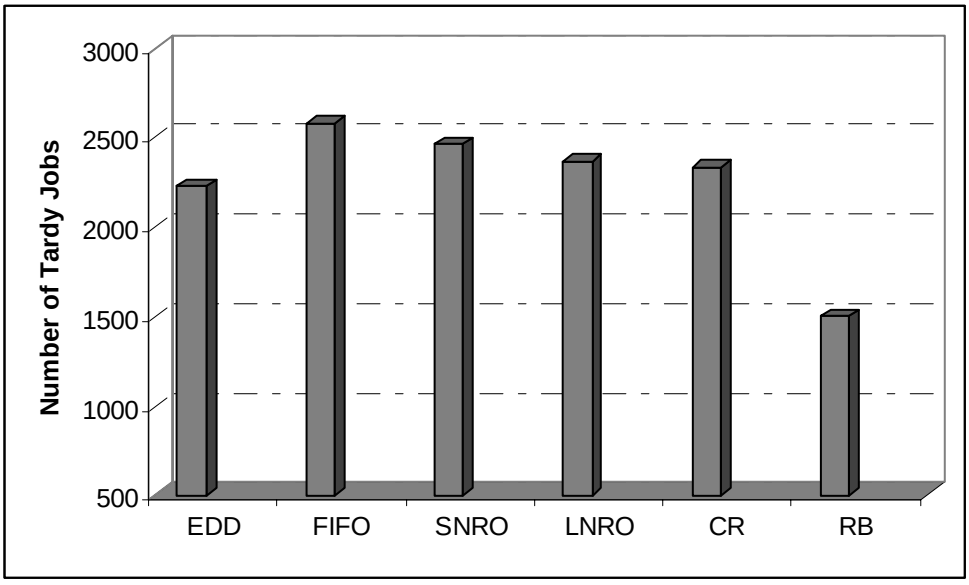


Figure 4.20 Total number of tardy jobs comparison

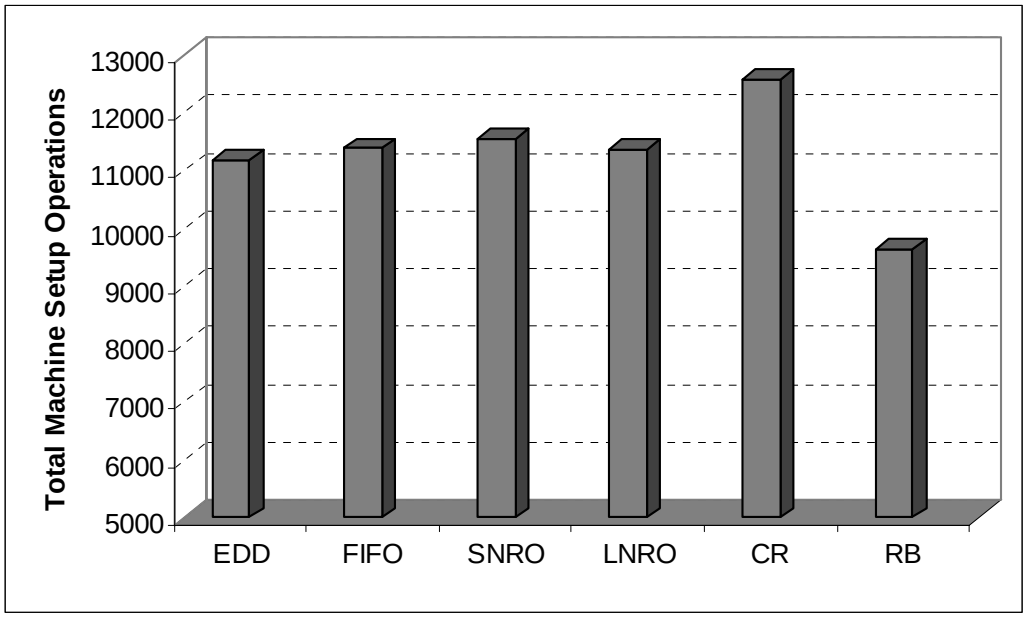


Figure 4.21 Total number of setup operations for each system configuration

The results show that routing and dispatching policies have a large impact on the performance measures of the system. The longer flow times imply a lower throughput rate and lower shop productivity, which cause a slow delivery of the products to the customers. The effect of longer flow times are reflected in inventory levels and resource capacity utilization rates. From the above results, a general conclusion is that the proposed rule-based system performs significantly better than the conventional sequencing rules for all of the performance measures, as it considers setup-condition of the machines, the alternative routes and due date requirements of the jobs at the same time, and the jobs are chosen with respect to the current state of the system.

The results also display that setup time and alternative routes have a negative impact on the performance of conventional sequencing rules, as they sort the jobs without any consideration of machine setup operations.

4.6 Analysis of the Rule-based System under Different System Configurations

In addition to giving the user insight into how complex systems function and how variables interact with each other, the proposed DSS provides the user with an informative approximation of “what-if” scenarios to assist in supporting long-term strategic decisions or real-time decision-making. For this purpose, the system performance is examined under different system configurations based on the proposed decision support system in order to find out the best combination of operating policy of the system. The system simulation has been carried out by varying the levels of the following system parameters:

- number of transpallet in the system (1, 2, 3, 4, 5)
- number of operators in the system (4, 5, 6, 7, 8)

In Table 4.9, different combinations of system configurations are presented.

Table 4.9 Different system configurations for performance analysis

Configuration No	Number of transpallet	Number of operators
1	1	4
2	1	5
3	1	6
4	1	7
5	1	8
6	2	4
7	2	5
8	2	6
9	2	7
10	2	8
11	3	4
12	3	5
13	3	6
14	3	7
15	3	8
16	4	4
17	4	5
18	4	6
19	4	7
20	4	8
21	5	4
22	5	5
23	5	6
24	5	7
25	5	8

In the following table, the performance measures for different system configurations for one replication are presented.

Table 4.10 Performance measures versus different system configurations

Configuration No	Mean Flow Time	Mean Tardiness	Proportion of Tardy Jobs (%)	Number of Tardy Jobs
1	115,67	20,28	33	1507
2	115,38	19,61	33	1495
3	114,67	18,67	32	1469
4	112,50	17,40	32	1440
5	112,50	17,40	32	1440
6	110,55	16,61	29	1324
7	104,85	13,20	27	1248
8	105,93	14,19	27	1248
9	103,59	12,44	26	1201
10	105,88	13,44	27	1242
11	107,70	15,03	29	1305
12	108,35	15,92	28	1277
13	107,14	15,00	29	1381
14	106,47	14,24	28	1270
15	106,47	14,24	28	1270
16	105,89	14,00	27	1208
17	107,05	14,34	29	1347
18	105,09	14,16	27	1269
19	108,53	15,81	28	1321
20	108,53	15,81	28	1321
21	105,89	14,00	27	1208
22	107,05	14,34	29	1347
23	105,09	14,16	27	1269
24	108,53	15,81	28	1321
25	108,53	15,81	28	1321

As can be seen in this table, the fourth and fifth groups of configurations (i.e. transpallet number 4 and 5, respectively) perform the same for the same operator numbers. For instance, performance results for configurations 16 and 21 are the same, and configurations 17 and 22, and so on. It means that, if we continue to increase the number of transpallet after 3, it does not affect the system performance. In the same way, operator numbers 7 and 8 give the same results for all cases except the configurations with two transpallets. Thus, it can be concluded that even we keep on increasing the number of operators in the system it does not affect the system performance significantly after seven operators.

Briefly, implementation results indicate that configurations 9 give better results than the other configurations. This configuration represents two and seven operators in the system, respectively. The current system has one transpallet and eight operators. Consequently, the system needs to increase the number of transpallets, while decreasing the number of operators in the system to achieve the best results on system performance.

Moreover, we can observe from the experimental results that, the higher numbers of transpallet and operators in the system will not always give better results on system performance. One possible reason of this outcome is that the number of machine setup operations also increases when the system start to response to requests quickly. In other words, the difference on the system configurations also affects the setup operations of the machines shared for different product types. Therefore, system performance does not improve significantly after a certain level, although we continue to increase the number of transpallet and operator in the system. Table 4.11 shows the number of setup operations of each machine for the system configurations.

Table 4.11 Number of setup operations of the machines

Conf.	CNC1	CNC1	HMC1	HMC2	HMC3	Multip.1	Multip.2	DRILL1	DRILL2	**Total Setup**
1	1	1	103	255	1	1746	1652	2872	2924	**9555**
2	1	1	95	261	1	1789	1693	2887	2926	**9654**
3	1	1	98	281	1	1780	1694	2881	2924	**9661**
4	1	1	76	266	1	1756	1641	2827	2879	**9448**
5	1	1	76	266	1	1756	1641	2827	2879	**9448**
6	1	1	84	226	1	1776	1690	2850	2859	**9488**
7	1	1	105	245	1	1756	1668	2843	2861	**9481**
8	1	1	90	285	1	1717	1626	2825	2845	**9391**
9	1	1	78	224	1	1711	1620	2826	2842	**9304**
10	1	1	101	282	1	1760	1681	2840	2856	**9523**
11	1	1	96	268	1	1751	1640	2829	2848	**9435**
12	1	1	102	271	1	1771	1678	2892	2903	**9620**
13	1	1	113	274	1	1776	1653	2952	2971	**9742**
14	1	1	100	265	1	1775	1710	2916	2928	**9697**
15	1	1	100	265	1	1775	1710	2916	2928	**9697**
16	1	1	86	226	1	1687	1578	2757	2766	**9103**
17	1	1	96	253	1	1813	1723	2914	2930	**9732**
18	1	1	102	246	1	1712	1651	2832	2845	**9391**
19	1	1	104	283	1	1823	1702	2964	2983	**9862**
20	1	1	104	283	1	1823	1702	2964	2983	**9862**
21	1	1	86	226	1	1687	1578	2757	2766	**9103**
22	1	1	96	253	1	1813	1723	2914	2930	**9732**
23	1	1	102	246	1	1712	1651	2832	2845	**9391**
24	1	1	104	283	1	1823	1702	2964	2983	**9862**
25	1	1	104	283	1	1823	1702	2964	2983	**9862**

The impact of resource levels on system performance is closely related to the other internal and external factors from manufacturing environment such as product types arriving rates, order sizes, so on, hence higher levels of resources may not always yield good results on the system performance. Therefore, the system parameters and operational control rules should be tested under different environmental conditions in order to find out the best implementing policy. However, the best combination of part routing control policy and dispatching rule is highly system dependent and each system needs for individual study. Since the proposed DSS and modeling methodology has generic, flexible and modular structure, it can be easily adapted to changing conditions by modifying only some parameters of the object models, and employed to analyze the system performance. Considering the high investment costs of FMSs, it is certainly worth choosing the best operating policy and system configuration by analyzing the developed system model, and adapting to changes over time.

CHAPTER FIVE
CONCLUSION AND FUTURE RESEARCH

In this thesis, a PN based DSS for shop floor scheduling and control of FMSs is presented. This study shows that high-level PNs and OOD approaches can be used effectively to model dynamic and stochastic nature of FMSs, to provide a thorough understanding of the dynamic behavior of a system, and to assist the evaluation of various system operational strategies. The methodology presented in this study can be applied for designing, analyzing, specification, and evaluation of FMSs as well as for solving their part routings and resource assignment problems using a real-time decision making tool.

In Chapter 3, the brief description and general characteristics of the proposed DSS has been provided. The developed rule-based DSS aims to solve the resource contention problem and to determine the best routes of the parts that have routing flexibility. The modeling methodology presented in Chapter 3 has been applied to a flexible manufacturing system, and an attempt has been made to develop a heuristic rule base for the dynamic scheduling of an FMS when alternative machines are included in Chapter 4. In the developed architecture, decisions are taken on real-time basis by considering product types, due dates, alternative process plans, next possible destination resource, machine setup status, and resource utilization rates.

The proposed DSS takes a global view of the system state before making decision about resource assignment, and proposes a dynamic approach to solve the conflict problems. It has adaptive and autonomous ability for obtaining intelligent control, and can be used for different production systems by only changing some system parameters (i.e. number of operators and stations, types of products, and process plan information) in Object Classes. Changes in an object or in the knowledge base can be

made easily without having to modify other parts or disturbing the operations of the system.

The implementation study presented in Chapter 4 puts forward that the proposed rule based system has performed satisfactorily for minimizing mean flow time, mean tardiness, number of tardy jobs, and proportion of tardy jobs. The reason why the DSS yielded better performance compared to several dispatching rules such as FIFO, EDD, LNRO, SNRO, and CR is that a single dispatching rule is not efficient to solve the conflicts due to resource contention situations, and may not be enough to succeed in meeting several objectives simultaneously. Using the developed DSS, a system's performance can be analyzed under different system configurations, and proposed heuristic rule based approach can be compared with other dispatching rules

In real production environments, there are many operation conditions that can be varied so that the results obtained may be different. For instance, a variation in inter arrival time, operation times, routing of the parts, volume of production, production mix, level of buffer capacities, and machine failure rates can lead to different results and have different effects. Therefore, it is not practical to find all the optimal levels of factors, which cover the ever-changing operation conditions. Indeed, it is much more fruitful to develop a methodology that provides the procedures for the system designers, so that they can carry out the evaluation by themselves under their own operation conditions. The modeling methodology and DSS presented here can be easily modified to apply for different systems, and can be extended continuously to incorporate additional functionality. It can be further improved by extending the heuristic rule base to include supplementary constraints such as dynamic tool allocation and sequence dependent setup times For practical implementation of the proposed DSS, additional research in the area of human interfaces could be useful to develop more user friendly system which automatically constructs PN models of object classes from the knowledge base of a flexible manufacturing system

REFERENCES

Abdallah, B., ElMaraghy, H. A., & ElMekkawy, T. (2002). Deadlock-free scheduling in flexible manufacturing systems. *International Journal of Production Research, 40* (12), 2733-2756.

Abu Hammad, A.A. (2003). *A decision support system for manufactured housing production process planning and facility layout*. Ph.D. Thesis, University of Cincinnati.

Baker, K.R. (1974). *Introduction to Sequencing and Scheduling*. New York: Wiley.

Boose, J., Bradshaw, J., Koszarek, J., & Shema, D. (1993) Knowledge acquisition techniques for group decision support, *Knowledge Acquisition*, 5, 405-448.

Booth, A.W. (1998). Object-oriented modeling for flexible manufacturing systems. *International Journal of Flexible Manufacturing Systems, 10*, 301-314.

Byrne, M.D., & Chutima, P. (1997). Real-time operational control of an FMS with full routing flexibility. *International Journal of Production Economics, 51*, 109-113.

Camurri, A., Franchi, P., Gandolfo, F., & Zaccaria, R. (1993). Petri net based process scheduling: A model of the control system of flexible manufacturing systems. *Journal of Intelligent and Robotic Systems, 8*, 99-123.

Cavory, G., Dupas, R., & Goncalves, G. (2005). A genetic approach to solving the problem of cyclic job shop scheduling with linear constraints. *European Journal of Operational Research, 161* (1), 73-85.

Chan, F. T. S., & Chan, H. K. (2001). Dynamic scheduling for a flexible manufacturing system–the pre-emptive approach. *International Journal of Advanced Manufacturing Technology*, *17*, 760-768.

Chan, F.T.S. (2003). Effects of dispatching and routeing decisions on the performance of a flexible manufacturing system. *International Journal of Advanced Manufacturing Technology, 21,* 328-338.

Chan, F.T.S., & Chan, H.K. (2004). Analysis of dynamic control strategies of an FMS under different scenarios, *Robotics and Computer-Integrated Manufacturing, 20*, 423-437.

Chen, J., & Chen, F.F. (2003). Performance modelling and evaluation of dynamic tool allocation in flexible manufacturing systems using coloured Petri nets: An object-oriented approach. *International Journal of Advanced Manufacturing Technology, 21* (2), 98-109.

Chen, J.H., Fu, L.C., Lin, M.H., & Huang, A.C. (2001). Petri-net and GA-based approach to modelling, scheduling, and performance evaluation for wafer fabrication. *IEEE Transactions on Robotics and Automation, 17* (5), 619-636.

Chen, Q., Luh, J.Y.S., & Shen, L. (1994). Complexity reduction for optimization of deterministic timed Petri-net scheduling by truncation. *Cybernetics and Systems, 25* (5), 643-695.

Chetty, O.V.K., & Gnanasekaran, O.C. (1996). Modelling, simulation and scheduling of flexible assembly systems with coloured petri nets. *International Journal of Advanced Manufacturing Technology*, *11* (6), 430-438.

Chincholkar, A.K., & Chetty, O.V.K. (1996). Stochastic coloured Petri nets for modelling and evaluation, and Heuristic rule base for scheduling of FMS. *International Journal of Advanced Manufacturing Technology, 12* (5), 339-348.

Chiu, C., & Yih, Y. (1995). A learning-based methodology for dynamic scheduling in distributed manufacturing systems, *International Journal of Production Research, 33* (11), 3217-3232.

Chong, C.S., Sivakumar, A.I., & Gay, R. (2003). Simulation-based scheduling for dynamic discrete manufacturing. *Proceedings of the 2003 Winter Simulation Conference (IEEE Cat. No.03CH7499), 2*, 1465-1473.

Desrochers, A.A., & Al-Jaar, R.Y. (1995). *Applications of Petri nets in Manufacturing Systems*. New York, NY: Institute of Electrical and Electronics Engineers Press.

DiCesare, F., Harhalakis, G., Proth, J.M., Silva, M., & Vernadat, F.B. (1993). *Practice of Petri Nets in Manufacturing*. London: Chapman & Hall.

Elmekkawy, T., & Elmaraghy, H.A. (2003). Efficient search of Petri Nets for deadlock-free scheduling in FMSs using heuristic functions. *International Journal of Computer Integrated Manufacturing, 16* (1), 14-24.

Fung, R.Y.K., Jiang, Z., Zuo, M.J., & Tu, P.Y.L. (2002). Adaptive production scheduling of virtual production systems using object-oriented Petri nets with changeable structure. *International Journal of Production Research, 40* (8), 1759-1785.

Gang, X., & Zhiming, W. (2004). Deadlock-free scheduling strategy for automated production cell. *IEEE Transactions on Systems, Man and Cybernetics, Part C, 34* (1), 113-122.

Ghaeli, M., Bahri, P.A., Lee, P., & Gu, T. (2005). Petri-net based formulation and algorithm for short-term scheduling of batch plants. *Computers & Chemical Engineering, 29* (2), 249-259.

Gupta, A.K.; Sivakumar, A.I., & Sarawgi, S. (2002). Shop floor scheduling with simulation based proactive decision support. *Winter Simulation Conference Proceedings, 2*, 1897-1902.

Hack, M.H.T. (1972). *Analysis of Production Schemata by Petri Nets*. Massachusetts Inst of Tech Cambridge Project Mac., MAC-TR-94.

Hatano, I., Yamagata, K., & Tamura, H. (1991). Modeling and on-line scheduling of flexible manufacturing systems using stochastic Petri nets. *IEEE Trans. Software Eng., 17* (2), 126-133.

He, D.W., Strege, B., Tolle, H., & Kusiak, A. (2000). Decomposition in automatic generation of Petri nets for manufacturing system control and scheduling. *International Journal of Production Research, 38* (6), 1437-1457.

Hillion, H.P., Proth, J.M., & Xie, X.L. (1987). A heuristic algorithm for the periodic scheduling and sequencing job-shop problem. *26th IEEE Conference on Decision and Control, 1*, 612-617.

Hillion, H.P., & Proth, J.M. (1989). Performance evaluation of job-shop systems using timed event graphs. *IEEE Trans. Autom. Control., 34* (1), 3-9.

Hillion, H.P., & Proth, J.M. (1989). Using timed Petri nets for the scheduling of job-shop systems. *Engineering Costs and Production Economics, 17*, 149-154.

Holthaus, O., & Rajendran, C. (2000). Efficient job shop dispatching rules: further developments. *Production planning & Control, 11* (2), 171-178.

Hu, G.H., Wong, Y.S., & Loh, H.T. (1995). An FMS scheduling and control decision support system based on generalised stochastic Petri nets. *International Journal of Advanced Manufacturing Technology, 10*, 52-58.

Jain, P.K. (2001). Solving resource contention problem in FMS using Petri nets and a rule-based approach. *International Journal of Production Research, 39* (4), 785-808.

Jain, V., Swarnkar, R., & Tiwari, M.K. (2003). Modelling and analysis of wafer fabrication scheduling via generalized stochastic Petri nets and simulated annealing. *International Journal of Production Research, 41* (15), 3501-3527.

Jeng, M.D., & DiCesare, F. (1993). A Review of Synthesis Techniques for Petri Nets with Applications to Automated Manufacturing Systems. *IEEE Transactions on Systems, Man, and Cybernetics, 23*, 301-312.

Jeng, M.D., & Chen, S.C. (1998). Heuristic search approach using approximate solutions to Petri net state equations for scheduling flexible manufacturing systems. *International Journal of Flexible Manufacturing Systems, 10* (2), 139-162.

Jeng, M.D., Lin, C.S., & Huang, Y.S. (1999). Petri net dynamics-based scheduling of flexible manufacturing systems with assembly. *Journal of Intelligent Manufacturing, 10* (6), 541-555.

Jiang, Z.B., Liu, M.Z., & Zhao, H. (2000). Virtual production systems and their modeling by temporized object-oriented Petri net changeable structure (TOPNs-CS). *Proceedings of ICME 2000 November.*

Jones, A., & Rabelo, J.C. (1998). Survey of Job Shop Techniques. *NISTIR National Institute of Standards and Technology*. Gaithersbur, MD.

Kher, V.H. (2000). Examination of worker assignment and dispatching rules for vital customer priorities in dual resource constrained job shop environments. *Computers & Operations Research, 27*, 525-537.

Kim, S.C., & Bobrowski, P.M. (1997). Scheduling jobs with uncertain setup times and sequence dependency. *Omega, International Journal of Management Science, 25* (4), 437-447.

Kis, T., Kiritsis, D., Xirouchakis, P., & Neuendorf, K.P. (2000). Petri net model for integrated process and job shop production planning. *Journal of Intelligent Manufacturing, 11* (2), 191-207.

Korbaa, O., Benasser, A., & Yim, P. (2003). Two FMS scheduling methods based on Petri nets: a global and a local approach. *International Journal of Production Research, 41* (7), 1349-1371.

Koriem, S.M., & Patnaik, L.M. (1997). A generalized stochastic high-level Petri net model for performance analysis. *Journal of Systems Software, 36*, 247-265.

Kumar, R., Tiwari, M.K., & Allada, V. (2004). Modelling and rescheduling of a re-entrant wafer fabrication line involving machine unreliability. *International Journal of Production Research, 42* (21), 4431-4455.

Kuo, C.H., Huang, H.P., & Yeh, M.C. (1998). Object-oriented approach of MCTPN for modeling flexible manufacturing system. *International Journal of Advanced Manufacturing Technology, 14*, 737-749.

Kusiak, A. (2000). *Computational Intelligence in Design and Manufacturing*. New York: John Wiley & Sons Inc.

Larson, N., & Kusiak, A. (1996). Managing design processes: a risk assessment. *IEEE Trans. Syst. Man, Cybern. Part A: Systems Humans, 26* (6), 749-759.

Lee, D.Y., & DiCesare, F. (1994a). Scheduling flexible manufacturing systems using Petri nets and heuristic search. *IEEE Transactions on Robotics and Automation, 10* (2), 123-133.

Lee, D.Y., & DiCesare, F. (1994b). Integrated scheduling of flexible manufacturing systems employing automated guided vehicles. *IEEE Transactions on Industrial Electronics, 41* (6), 602-610.

Lee, J., & Korbaa, O. (2004). Modeling and scheduling of ratio-driven FMS using unfolding time Petri nets. *Computers & Industrial Engineering, 46* (4), 639-653.

Lin, C., & Marinescu, D. C. (1988). Stochastic High-Level Petri Nets and Applications, *IEEE Trans. Comp., 37*, 815-823.

Lin, G.Y.-J. (1993). *A distributed production control for intelligent manufacturing systems*. Ph.D. Dissertation, Purdue University.

Lin, J.T., & Lee, C.C. (1997). Petri net-based integrated control and scheduling scheme for flexible manufacturing cells. *Computer Integrated Manufacturing Systems, 10* (2), 109-122.

Lin, H., Fan, Y., & Loiacono, E.T. (2004). A practical scheduling method based on workflow management technology. *International Journal of Advanced Manufacturing Technology, 24*, 919-924.

Marsan, M.A., Balbo, G., Conte, G., Donatelli, S., & Franceschinis, G. (1995). *Modelling with Generalized Stochastic Petri Nets*. West Sussex: John Wiley & Sons.

Mohamed, N.S. (1998). Operation planning and scheduling problems in an FMS: An integrated approach. *Comp. Ind. Eng., 35* (3-4), 443-446.

Moore, K.E., & Gupta, S.M. (1996). Petri Net Models of Flexible and Automated Manufacturing Systems: a survey. *International Journal of Production Research, 34* (11), 3001-3035.

Moore, K.E., Gungor, A., & Gupta, S.M. (2001). Petri net approach to disassembly process planning for products with complex AND/OR precedence relationships. *European Journal of Operational Research, 135* (2), 428–449.

Moro, A.R., Yu, H., & Kelleher, G. (2002). Hybrid heuristic search for the scheduling of flexible manufacturing systems using Petri nets. *IEEE Transactions on Robotics and Automation, 18* (2), 240-245.

Morton, T.E., & Pentico, D.W. (1993). *Heuristic Scheduling Systems*. John Wiley and Sons Inc, New York, USA.

Murata, T. (1989). Petri Nets: Properties, Analysis and Applications. *Proceedings of IEEE, 77* (4), 541-580.

Nayak, G. K., & Acharya, D. (1997). Generation and validation of FMS test prototypes. *International Journal of Production Research, 35 (*3), 805-826.

O'Kane, J.F. (2000). A knowledge-based system for reactive scheduling decision-making in FMS. *Journal of Intelligent Manufacturing, 11* (5), 461-474.

Ozmutlu, S., & Harmonosky, C.M. (2005). A real-time methodology for minimizing mean flowtime in FMSs with routing flexibility: Threshold-based alternative routing. *European Journal of Operational Research, 166*, 369-384.

Peng, P., & Chen, F.F. (1998). Real-time control and scheduling of flexible manufacturing systems: an ordinal optimisation based approach. International Journal of Advanced Manufacturing Technology, 14 (10), 775-786.

Pinedo, M. (1995). *Scheduling Theory, Algorithms, and Systems*. New Jersey, USA: Prentice Hall.

Proth, J.M., & Xie, X. (1996). *Petri nets: A tool for design and management of manufacturing systems.* Chichester, UK: John Wiley & Sons Inc.

Proth, J.M., & Sauer, N. (1998). Scheduling of piecewise constant product flows: A Petri net approach. *European Journal of Operational Research, 106* (1), 45-56.

Rachamadugu, R., & Stecke, K.E. (1994). Classification and review of FMS scheduling procedures. *Production planning and Control, 5*, 2-20.

Rajendran, C., & Holhaus, O. (1999). A comparative study of dispatching rules in dynamic flowshops and jobshops. *European Journal of Operational Research, 116*, 156-170.

Raju, K.R., & Chetty O.V.K. (1993). Priority nets for scheduling flexible manufacturing systems. *Journal of Manufacturing Systems, 12* (4), 326-340.

RSoft Design Group, Inc. (2002). *Full spectrum photonic and network design automation*. Retrieved November 29, 2005, from http://www.rsoftdesign.com/products/network_modeling/Artifex/

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., & Lorensen, W. (1991). Object-Oriented Modeling and Design, Englewood Cliffs, NJ: Prentice-Hall.

Russo, M.F., & Sasso, A. (2005). Modeling, analysis, simulation and control of laboratory automation systems using Petri nets: Part 1. Modeling. *JALA Tutorial, 10*, The Association for Laboratory Automation doi:10.1016/j.jala.2005, 172-181.

Saygin, C., Chen, F.F., & Singh, J. (2001). Real-time manipulation of alternative routeings in flexible manufacturing systems: a simulation study. *International Journal of Advanced Manufacturing Technology*, 18, 755-763.

Schniederjans, M.J., & Carpenter, D.A. (1996). A heuristic job shop scheduling decision support system: a case study. *Decision Support Systems, 18*, 159-166.

Shih, H., & Sekiguchi, T. (1991). A timed Petri net and beam search based on-line FMS scheduling system with routing flexibility. *Proceedings of the 1991 IEEE Int. Conf. On Robotics and Automation*, Sacramento, Ca, pp.2548-2553.

Sivagnanavelu, D.G. (2000). *Dynamic scheduling of flexible manufacturing systems: a study of machine and material handling control strategies*. M.Sc. Thesis, Concordia University, Montreal, Canada.

Song, J.S., & Lee, T.E. (1998). Petri net modeling and scheduling for cyclic job shops with blocking. *Computers and Industrial Engineering, 34* (2), 281-295.

Subramaniam, V., Ramesh, T., Lee, G.K., Wong, Y.S., & Hong, G.S. (2000a). Job shop scheduling with dynamic fuzzy selection of dispatching rules. *International Journal of Advanced Manufacturing Technology, 16*, 759-764.

Subramaniam, V., Lee, G.K., Hong, G.S., Wong, Y.S., & Ramesh, T. (2000b). Dynamic selection of dispatching rules for job shop scheduling. *Production Planning & Control, 11* (1), 73-81.

Sun, T.H., Cheng, C.W., & Fu, L.C. (1994). Petri net based approach to modeling and scheduling for an FMS and a case study. *IEEE Transactions on Industrial Electronics, 41* (6), 593-601.

Tang, Y., Zhou, M.C., & Caudill, R.J. (2001). An integrated approach to disassembly planning and demanufacturing operation. *IEEE Transactions on Robotics and Automation, 17* (6), 773-784.

Tricas, F., & Martinez, J. (1998). Distributed control systems simulation using high level Petri nets. *Mathematics and Computers in Simulation, 46*, 47-55.

Tuncel, G., & Bayhan, G.M. (2003). Survey of scheduling in manufacturing systems with Petri nets. *32nd International Conference on Computers and Industrial Engineering (ICC&IE)*, Limerick, Ireland, 803-808.

Tuncel, G., & Bayhan, G.M. (2005). A high-level Petri net based decision support system for real-time scheduling and control of flexible manufacturing systems: an object-oriented approach. *Lecture Notes in Computer Science, LNCS 3514*: Sunderam, V., van Albada, G.D., Sloot, P.M.A., Dongarra, J.J. (Eds.), 843-851.

Turban, E., & Aronson, J. (2001). *Decision Support Systems and Intelligent Systems.* Sixth Edition, Upper Saddle River, N.J.: Prentice Hall.

Venkatesh, K., & Zhou, M.C. (1998). Object-oriented design of FMS control software based on object modeling techniques diagrams and Petri nets. *Journal of Manufacturing Systems, 17* (2), 118-136.

Wang, L., & Xie, X. L. (1996). Modular Modelling Using Petri Nets. *IEEE Transactions on Robotics and Automation, 12*, 800-809.

Wang, L. (1996). Object-oriented Petri nets for modelling and analysis of automated manufacturing systems. *Computer Integrated Manufacturing Systems, 26* (2), 111-125.

Wang, L., & Wu, S.Y. (1998). Modeling with colored timed object-oriented Petri nets for automated manufacturing systems, *Computers and Industrial Engineering, 34* (2), 463-480.

Wu, S.Y.D., & Wysk, R.A. (1989). An application of discrete event simulation to on-line control and scheduling in flexible manufacturing systems. *International Journal of Production Research, 27*, 603-1623.

Wu, X. (1999). *DSS user interface design method with application to shop floor scheduling*. M.Sc. Thesis, Department of Mechanical Engineering, University of Alberta, Edmonton.

Xiong, H.H. (1996). *Scheduling and discrete event control of flexible manufacturing systems based on Petri nets.* Ph.D. Dissertation, New Jersey Institute of Technology, USA.

Xiong, H.H., Zhou, M.C., & Caudill, R.J. (1996). A hybrid heuristic search algorithm for scheduling flexible manufacturing systems. *Proceedings of IEEE International Conference on Robotics and Automation, 3*, 2793-2797.

Xiong, H.H., & Zhou, M.C. (1998). Scheduling of semiconductor test facility via Petri nets and hybrid heuristic search. *IEEE Transactions on Semiconductor Manufacturing, 11* (3), 384-393.

Yan, H.S., Wang, N.S., Zhang, J.G., & Cui, X.Y. (1998). Modelling, scheduling and simulation of flexible manufacturing systems using extended stochastic high-level evaluation Petri nets. *Robotics and Computer-Integrated Manufacturing, 14* (2), 121-140.

Yan, H.S., Wang, N.S., Cui, X.Y., & Zhang, J.G. (1997). Modeling, scheduling and control of flexible manufacturing systems by extended high-level evaluation Petri nets. *IIE Transactions, 29* (2), 147-158.

Yim, D.S., & Linn, R.J. (1993). Push and pull rules for dispatching automated guided vehicles in a flexible manufacturing system. *International Journal of Production Research, 31* (1), 43-57.

Yu, H., Reyes, A., Cang, S., & Lloyd, S. (2003). Combined Petri nets modelling and AI-based heuristic hybrid search for flexible manufacturing systems-part II. Heuristic hybrid search. *Computers & Industrial Engineering, 44* (4), 545-566.

Zhou, M.C., & DiCesare, F. (1993). *Petri Net Synthesis for Discrete Event Control of Manufacturing Systems*. Norwell, Massachusetts: Kluwer Academic Publishers.

Zhou, M.C., & Jeng, M.D. (1998). Modeling, Analysis, Simulation, Scheduling, and Control of Semiconductor Manufacturing Systems. *IEEE Transactions on Semiconductor Manufacturing, 11* (3), 333-357.

Zhou, M.C., & Venkatesh, K. (1998). Modeling, simulation, and control of flexible manufacturing systems: A Petri net approach. *Intelligent Control and Intelligent Automation, 6,* World Scientific, Singapore.

Zhou, M.C., & Xiong, H.H. (1995). Petri net scheduling of FMS using branch-and-bound method. *Proceedings of IEEE Industrial Electronics, Control, and Instrumentation, 1* (6-10 November), 211-216.

Zhou, M.C., & Xiong, H.H. (2001). *Petri net modeling and scheduling of automated Manufacturing systems*. in Computer Aided Design, Engineering, and Manufacturing: Systems Techniques and Applications, Vol. IV, Optimization Methods for Manufacturing, C.T. Leondes (Ed.), CRC Press, Chapter 8, 1-23: Taylor & Francis Group, LLC.

Zopounidis, C., & Doumpos, M. (2000). PREFDIS: a multi criteria decision support system for sorting decision problems. *Computer & Operations Research, 27*, 779-797.

Zurawski, R., & Zhou, M.C. (1994). Petri Nets and Industrial Applications: A Tutorial. *IEEE Transactions on Industrial Electronics, 41* (6), 567-582.