Journal of the Korean Institute
of Industrial Engineers
Vol.21, No.1, March . 1995

Petri-Net을 이용한 FMS 제어 및 모니터링

FMS Control and Monitoring using Petri Net

김고중*, 정무영**, 조현보**

Go-Joong Kim*, Mooyoung Jung**, Hyunbo Cho**

Abstract

A difficult problem in operating Flexible Manufacturing Systems (FMS) is to control the system in real-time by coordinating heterogeneous machines and integrating distributed information. The objective of the paper is to present the models and methodologies useful to resolve the difficult problem. The detailed objectives can be described in three folds. First, a hierarchical Colored and Timed Petri-Net (CTPN) is designed to control an FMS in real-time. The concerned FMS consists of a loading station, several machining cells, a material handling system, and an unloading station. Timed-transitions are used to represent the timed-events such as AGV movements between stations and cells, part machining activities in the cells. Signal places are also used to represent communication status between the host and the cell controllers. To resolve the event conflicts and scheduling problems, dispatching rules are introduced and applied. Second, an implementation methodology used to monitor and diagnose the errors occurring on the machines during system operation is proposed. Third, a Petri-Net simulator is developed to experiment with the designed control logic.

Key Words : FMS, Control and monitoring, Colored and timed Petri-Net

1. 서론

FMS (Flexible Manufacturing System)의 두드러진 특징으로는 사건이 비동기적이고 이산적으로 발생하는 Discrete Event Dynamic System이라는 것과 예측할 수 없는 장비의 가

* 삼성전자(주), 정보통신 시스템 본부 SI 사업부

** 포항공과대학교 산업공학과

동중단 사태에 대비해야 한다는 것이다. 또한, 구성요소들의 상호작용이 동시발생적으로 반복되는 현상을 지니고 있다는 것도 특징 중의 하나이다. 이와 같은 시스템의 제어에는 최적제어 이론이 존재하는 연속공정 산업과는 달리 최적화 기법이 거의 존재하지 않는다. 이러한 시스템을 미분방정식이나 대기이론(queueing theory) 등으로 모델링 한다는 것은 너무 복잡하다[5]. 따라서 현존하는 생산 시스템들에서는 특정 FMS 구조(configuration)나 특정 제품만을 위하여 설계된 소프트웨어를 이용하여 이러한 어려운 문제들을 해결하기 위해 노력하고 있다. 그러나 이와 같은 해결책도 구현하기가 일반적으로 어렵다는 것 외에도 제어시스템의 수정 및 보완이 쉽지 않다는 문제점을 안고 있다. 또한, 주어진 설계, 계획, 또는 제어 방법들의 효과를 측정할 수 있는 방법론이 거의 존재하지 않는다는 현실적인 어려움도 내포하고 있다.

제어시스템의 설계 및 분석을 위한 하나의 가능한 방법으로서 시뮬레이션 기법을 이용할 수 있다. 그러나 시뮬레이션 모델이 시스템의 구조, 자원, 버퍼 등의 정적인 특성 및 시스템의 운영 시 발생하는 동시성, 비동기성과 같은 동적인 특성을 분석함에는 장점을 갖고 있으나 실제의 제어 문제에 있어서는 적합하지 않다는 단점도 동시에 안고 있다. 또한, 가동이 중단되었을 때에는 공작기계 오퍼레이터가 경험이나 직관에 의하여 작업순서, 절삭공구 및 공작기계 등으로 인한 문제를 관측하고 발견하여 수정할 수 있었으나 자동화된 환경에서는 이러한 기능들이 모니터링 및 제어 시스템에 의하여 제공되어야 한다는 실제적인 어려움도 내포하고 있다[19].

Petri-Net 모델링 패러다임은 이론적인 분석을 위한 기능을 제공할 뿐만 아니라 FMS에 존재하는 concurrency 와 resource conflict 등에 효과적인 모델을 제시하여 준다. 또한, 제어와 시뮬레이션을 동일한 framework 내에서 수행할 수 있다는 장점도 제공해 준다. 이에 따라 Petri-Net 모델링 패러다임은 정보시스템 모델링 및 분석, 컴퓨터 하드웨어 시뮬레이션 및 분석, 통신 프로토콜 검증, FMS의 분석 및 시뮬레이션 등의 분야에 광범위하게 이용되어 왔다[14]. Menon *et al.*[10]은 FMS 제어를 위한 coordination 제어 구축 방안을 제안하였다. 공정계획 모듈과 스케줄링 모듈을 인터페이스 시킴으로써 작업 수행을 위한 CPN (Colored Petri-Net)을 설계하였고, 대상 시스템에 대한 이산 사건 시뮬레이션을 수행할 수 있는 Petri-Net 시뮬레이터의 개발 방법을 제안하였다. 그러나, 이들의 방법은 부품의 공정 개수에 따라 Petri-Net의 크기가 증가한다는 단점을 갖고 있다. 한편, Murata [12]는 제어 소프트웨어의 설계와 더불어 사건행렬 (incidence matrix)을 이용하여 Petri-Net 모델에 오류가 발생했는가를 결정하는 방법을 제시하였다. 그러나 사건행렬이 도달 가능한 스페이스를 결정하기 위한 충분조건을 제시하지 않았다. 또한, 적절한 Marking상태에서는 복수 에러가 발생되었을 수도 있어 에러 감지가 항상 가능한 것은 아니다.

실시간 제어에 적용할 수 있는 실질적인 Petri-Net를 이용한 고장 감지 방법이 Sahra-

oui *et al.*[16]에 의하여 처음으로 제시되었다. 이를 위하여 place를 사건의 진행상태로 모델링하고, 토큰이 place로 투입된 후, 일정 시간이 지날 때까지 place에 토큰이 존재하면 이상 상황 발생신호를 내게 하는 시스템을 제안하였다. 프로그램의 오류나 기계설비의 고장 등으로 시스템이 루프를 끝없이 수행되는 것을 방지하기 위하여 고전적인 watchdog을 사용하였다. Watchdog에 의하여 이상 상황 발생신호가 나면 전문가 시스템을 이용하여 진단을 수행하였다.

기존의 연구들을 분석하여 보면, 제어에 있어서는 하위 수준의 컨트롤러의 구현 방법이나 특정 수준의 Coordination에 관한 연구가 주종을 이루어 왔다. 모니터링에 있어서는 하드웨어(watchdog)에 의존한 고장 감지 방법을 연구했음을 알 수 있다. 본 논문에서는 FMS의 계층적 제어구조를 CTPN(Colored Timed Petri-Nets)으로 설계하여 고장 감지 및 진단 기능을 수행할 수 있는 통합된 시스템의 설계 방안을 제시하였다. 또한 Petri-Net 시뮬레이터를 개발하여 설계된 제어로직의 수행 상황을 실험하였고 그 결과를 제시하였다.

2. Petri-Net 모델링 패러다임

Petri-Net는 일반적으로 합리적인 결정 능력을 제공하면서 다양한 시스템을 효과적으로 모델링할 수 있게 하는 것으로 알려져 있다. 특히, 동시적, 비동기적, 비확정적, 확률적인 특성을 가진 Discrete Event Dynamic System을 표현하고 해석하는 데에 유용하게 사용될 수 있다. 그러나 초기에 제안된 Petri-Net는 모델링 능력에 한계가 있기 때문에, 두 개의 새로운 Petri-Net, 즉 Colored Petri-Net [6]와 Timed Petri-Net[15]가 탄생되었다.

FMS의 상태는 시간에 따라 변하는 동적인 특성을 가지고 있으므로 이를 모델링하기 위해서는 고전적인 Petri-Net에 시간 매개변수를 도입해야 한다. 시간 매개변수를 도입하여 공정의 주기, 자원의 이용 현황, 가공되는 부품의 평균 개수, 그리고 throughput rate를 계산함으로써 시스템의 성능 분석을 도모할 수 있다[15]. 또한, 고전적인 Petri-Net의 단점 중의 하나인 방대한 크기를 줄이기 위해서 토큰이 표현하는 자원(resource)의 종류에 따라 각 토큰에 다른 색을 부여한다. 이러한 방법을 사용하면 복잡한 시스템을 간결하게 표현할 수 있게 된다[6]. 고전적인 Petri-Net에서는 Marking을 제외하면 (P, T, I, O)의 4개 요소로 이루어졌으나, CTPN에서는 토큰의 색을 지정할 수 있는 C 요소와 시간을 부가할 수 있는 D 요소가 추가되어 (P, T, I, O, C, D)의 6개로 이루어져 있다. 각각의 요소들은 다음과 같이 정의된다.

$P=\{p_1,p_2\cdots p_n\},\ n\rangle 0$

$T=\{t_1,t_2\cdots t_m\},\ m\rangle 0$

$C(p_i)=\{a_{i1},a_{i2}\cdots a_{iu_i}\},\ u_i=|C(p_i)|,i=1,2\cdots n$

$C(t_i)=\{b_{i1},b_{i2}\cdots b_{iv_j}\},\ v_j=|C(t_j)|,\ i=1,2\cdots m$

$D(t_j)=\{d_{j1},d_{j2}\cdots d_{jv_j}\},\ where\ t_j$ is timed trasition

$I(p,t)=C(p)\times C(t)\rightarrow N$

$O(p,t)=C(p)\times C(t)\rightarrow N$

CTPN의 Marking은 P 에 대하여 정의되는 벡터 함수 M(p) : C(p)→N이며 Marking의 구성요소는 (u_i×1)벡터이다. Place의 토큰들은 place 색의 합으로서 $M(p_i)=\sum_{j=1}^{u_i} n_{ih}a_{ih}$와 같이 표현되는데, n_{ih} 는 place p_i 에 있는 색이 a_{ih} 인 토큰의 개수를 나타낸다. 특정 트랜지션의 모든 입력 place에 대하여 $M(a_{ih}) \geq I(a_{ih}, b_{jk})$이면, 그 트랜지션은 Marking M에서 색 b_{jk} 에 대하여 실행 가능하게 된다.

3. FMS 제어 소프트웨어의 설계

본 논문에서 대상으로 하는 생산시스템(CMS: Cellular Manufacturing System)은 Loading 스테이션, Unloading 스테이션, CNC 공작기계설비, 로봇 등으로 구성되어 있는 가공셀들의 집합체이며, 이들 셀들은 물류처리 시스템으로 연결되어 있다. 이러한 CMS의 제어구조는 1) Factory level 2) Cell level 3) Equipment level의 세 수준이 통합적으로 운용되는 계층적 제어 시스템(HMS; Hierarchical Manufacturing System)으로 형성되어 있으며 그림 1에 그 관계를 나타냈다.

HMS는 데이타 베이스로부터 추출한 스케줄링 정보, 공구특성(예, 공구유형, 공구수명, 잔여수명 등), 그리고 부품 제조에 관한 특성(예, NC 프로그램, 부품유형, 가공설비, 작업수행시간, 공정경로) 등의 정보를 참조하여 전체 시스템을 제어한다. 이러한 시스템 내에서 많은 작업들이 동기적 혹은 비동기적으로 발생하게 되는데 이러한 작업들의 동기화(synchronization)는 각 작업들이 서로 연관되어 있기 때문에 매우 복잡하다. 이는 동기화 메카니즘을 통해서 물류의 통합과 생산정보의 통합을 실현함으로써 가능해진다.

Petri-Net를 이용한 CMS의 계층적 제어 메카니즘을 그림 2에 도시하였다. 제어 메카니즘은 토큰 플레이어 컨트롤러(TPC; Token Player Controller)에 의하여 구현된다. TPC는 공정계획 및 스케줄링 모듈과 인터페이스 되며 Petri-Net 상에서 자원 및 정보를 나타내는 토큰의 흐름을 제어한다.

3.1 토큰 플레이어 컨트롤러 (TPC)

TPC는 1) Coordinating CTPN 2) Transportation CTPN 3) Cell CTPN, 그리고 4) 이러한 CTPN을 운용하는 Token player로 구성되어 있다. TPC는 시스템의 핵심적인 부분으로서 시뮬레이션과 실시간 제어의 수행을 담당한다. FMS의 실시간 제어 및 시뮬레이션은 제시된 세 개의 CTPN을 운용하여 실현하고, 공정계획 및 스케줄링 모듈은 TPC 모듈로부터 필요한 시스템의 상태 정보를 추출한다. 설계된 CTPN들 사이의 상호 통신 체계는 Petri-Net의 확장인 Signal place에 의하여 정의되고 토큰 passing을 통하여 구현된다. 예를 들어, 시스템으로의 부품 투입부터 시작하여 가공을 완료할 때까지의 시나리오를 Net들 사이의 토큰Passing을 이용하여 표 1에 제시하였다. 구체적인 신호 입/출력 Place는 그림 3, 그림 4, 그리고 그림 5에 제시하였다. 시스템으로의 부품 투입을 나타내기 위해서 스케줄러가 토큰을 Coordinating CTPN 신호 입력 Place p1으로 Passing한다. 부품 수송을 요구하기 위해서 토큰을 Coordinating CTPN 출력 Place p10으로부터 Transportation CTPN 입력 Place p1으로 Passing시키면 된다. 이와

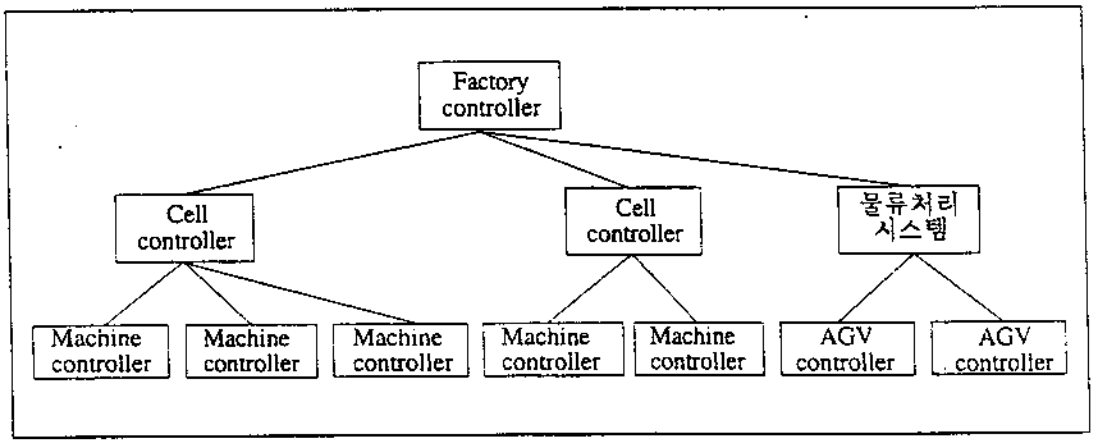

그림 1. 계층적 제어 시스템 구조

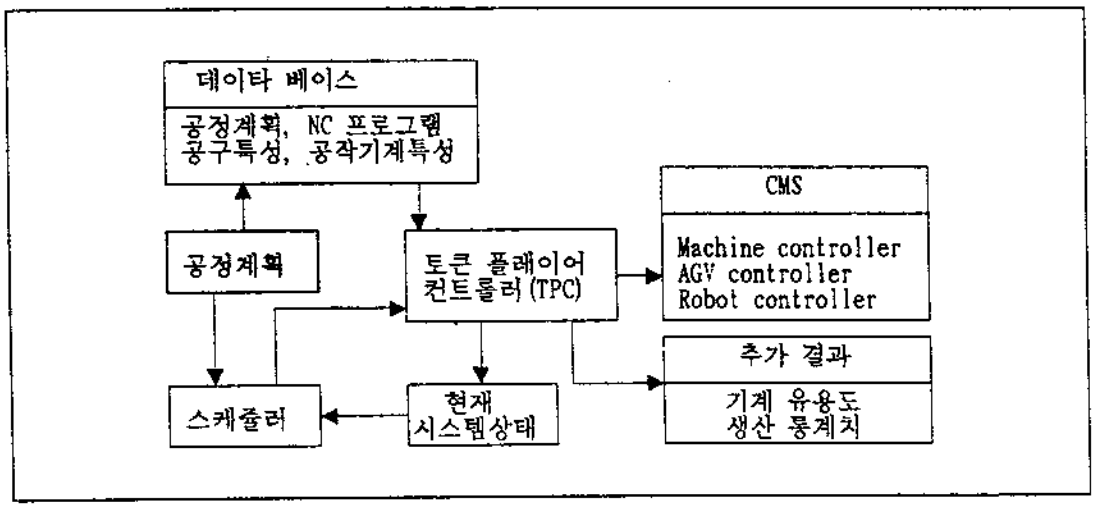

그림 2. 설계된 제어시스템 구조

같이 표 1에 나타난 과정을 거쳐 마지막으로 토큰이 Cell CTPN 출력 Place p24로부터 Coordinating CTPN 입력 place p16으로 Passing하면 셀가공이 완료되었다는 것을 나타낸다. 이러한 CTPN들 사이의 토큰 passing 및 CTPN내의 토큰 운용은 모든 CTPN을 총괄하는 TPC에 의하여 수행된다. Coordinating CTPN에서의 부품 입출력을 제외한 모든 CTPN들은 닫힌 시스템(closed system)이다.

Coordinating CTPN은 스케쥴러가 생성한 명령을 수행한다. 그림 3에서 볼 수 있듯이 Coordinating CTPN은 다른 Petri-Net들과의 통신을 통하여 셀간의 제어에 필요한 자원들의 조정에 대한 책임을 진다. 따라서 Coordi-

표 1. CTPN 연결도

신호출력 place	신호입력place	사건
스케줄러	Coordinating CTPN p1	FMS로 부품 투입
Coordinating CTPN p10	Transportation CTPN p1	수송 요구
Transportation CTPN p10	Coordinating CTPN p11	부품 pickup
Transportation CTPN p12	Coordinating CTPN p13	수송 완료
Coordinating CTPN p14	Cell CTPN p1	셀가공 요구
Cell CTPN p3	Coordinating CTPN p15	셀가공 시작
Cell CTPN p24	Coordinating CTPN p16	셀가공 완료

nating Petri-Net는 전체 시스템의 동작과 상태를 예측, 추적, 및 감독하는데 중심적인 역할을 담당한다. 예를 들어, Coordinating CTPN은 각 셀의 버퍼 상태를 추적하여 각 해당 셀의 입력 버퍼의 사용이 불가능 하면 부품의 수송을 요구하지 않는다. 또한 부품이 셀들 간에 유연하게 흐를 수 있도록 AGV를 호출하고 감독한다. 가장 중요한 임무는 부품의 위치 정보를 계속해서 수정할 뿐만 아니라 부품 상태를 추적하고 통제하는 것이다. FMS로 투입되는 부품은 셀의 방문 순서와 함께 공정계획을 가지고 있다. 공정계획에 있어서 선택 가능한 가공 경로가 둘이상 존재할 때는 작업장의 환경에 따라 실시간으로 하나의 경로를 선택하게 되는데 이의 선택 방법은 3.2 실시간 자원 할당에서 제시하였다.

그림 4에 제시된 Transportation CTPN 은 신호 입력 place p1으로 전달되는 coordinating CTPN의 수송 요구 신호에 의하여 실행되며 셀간 부품의 수송에 대한 책임을 진다. 수송 요구가 Coordinating CTPN으로부터 Signal place를 통하여 전달되면 수송 작업이 수행된다. 목적지까지의 이동은 세그멘트 단위로 수행되는데 경로상의 혼잡(congestion)이나 유용성 등을 판단하여 경로를 선정한다. 목적지까지의 수송이 완료되면 Coordinating CTPN으로 신호 출력 place p12를 통하여 알린다.

본 논문에서 고려한 셀은 부품의 가공을 위한 기계설비, 운반을 위한 로봇, 그리고 부품의 저장을 위한 버퍼로 구성되어 있다. 이러한 구조의 셀을 제어하는 Cell CTPN을 그림 5에 제시하였는데, Coordinating CTPN으로부터 토큰 passing을 통하여 작업 명령을 받는다. Coordinating CTPN의 신호 출력 place p14 로부터 Cell CTPN의 신호입력 place p1으로 작업 수행 명령이 하달되면 Cell은 작업을 시작한다. 부품들은 이송을 위한 로봇과 가공을 위한 기계를 사용할 수 있을 때까지 팔렛 저장소에 대기한다. 한 번의 작업을 수행하는 동안 기계설비와의 통신을 통해 loading, processing, unloading과정을 거치며, 공정계획에 명시된 모든 작업이 끝나면 신호 출력 place p24 를 통하여 작업의 완료 신호를 Coordinating CTPN으로 보낸다.

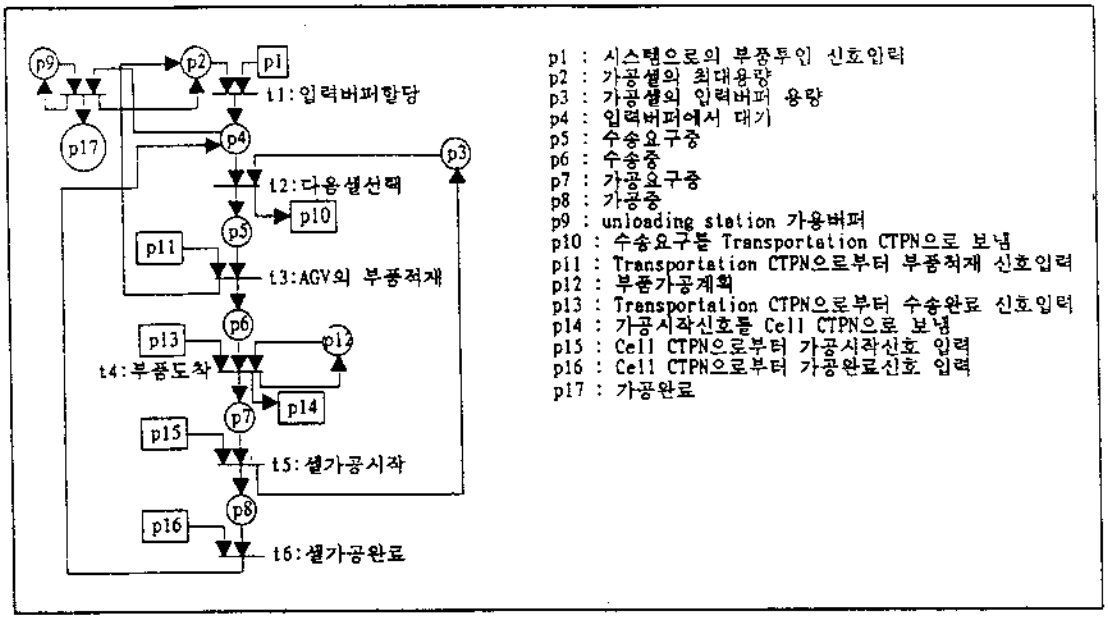

그림 3. Coordinating CTPN

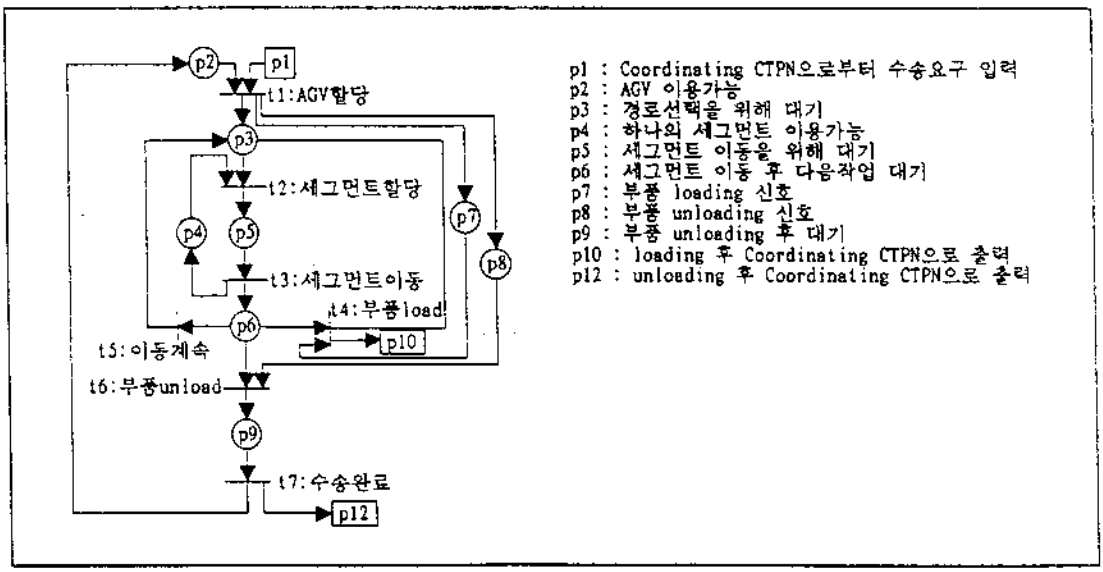

그림 4. Transportation CTPN

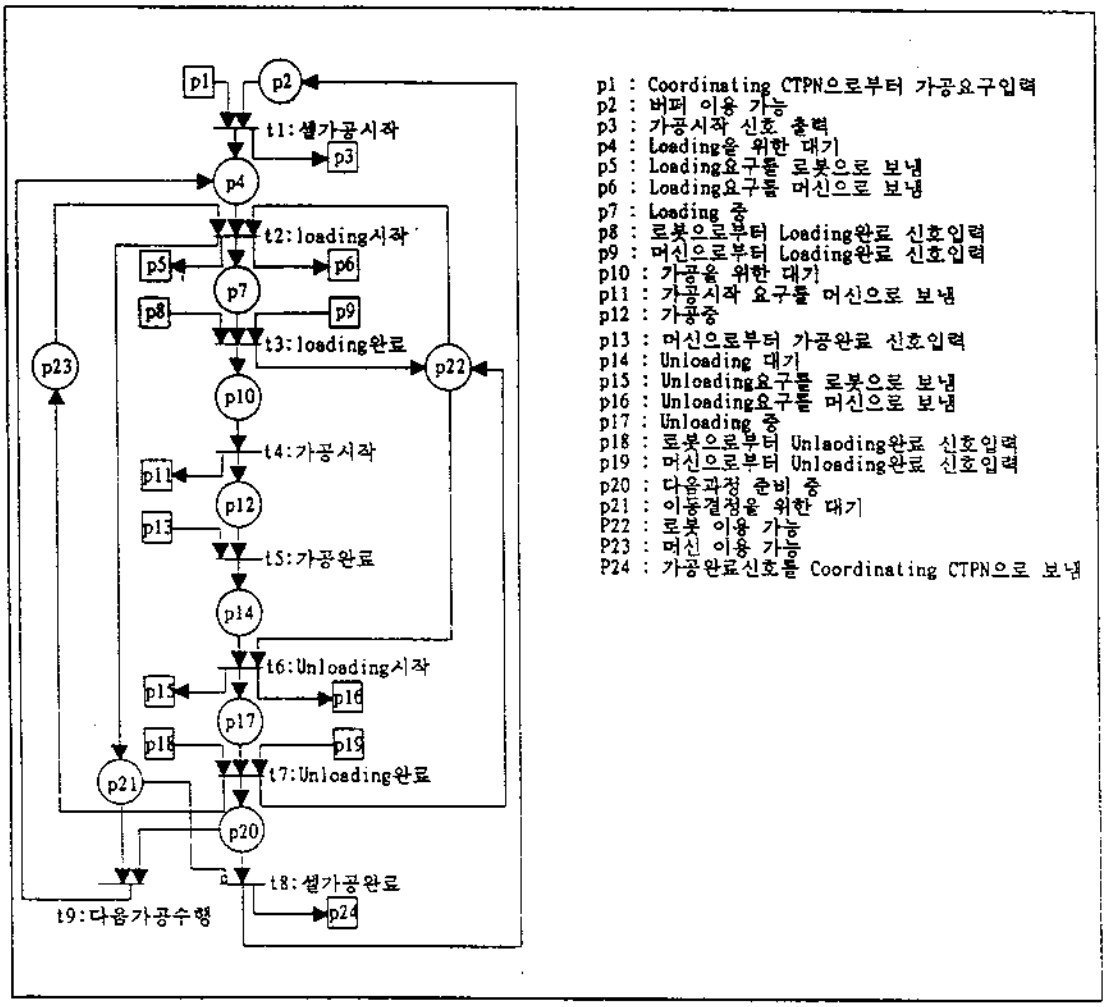

그림 5. Cell CTPN

3.2 실시간 자원 할당

실시간 Dispatching rule의 목적은 부품 입력(release), 부품 이동 등과 같은 상충문제를 해결하기 위한 것이다. 비록 자원들의 효율적인 사용을 위하여 branch-and-bound와 같은 최적 스케줄링 방법들이 사용될 수 있으나, 이들은 갑작스러운 변화에 대한 재스케줄링을 다루는데 너무 많은 시간을 필요로 한다. 반면에 Dispatching rule들은 최적해는 아니더라도 짧은 시간내에 좋은 해를 제시해 줄 수 있는 것으로 알려져 있다[9]. 따라서 본 논문에서는 실시간 시스템을 운용함에 있어서 필요한 계산부하(computational complexity)가 적게 하면서 온라인 스케줄링이 가능하도록 트랜지션의 상충현상을 Dispatching rule을 통해 해결하였다. 실제적인 응용에 있

어서는 단일 Rule 뿐만 아니라 Rule들의 조합이 사용될 수 있다.

4. 고장 감지 및 진단을 위한 모니터링 시스템

FMS는 기계설비에 오류가 발생할 경우에도 대체 경로를 통하여 정상적인 작업을 수행할 수 있는 능력을 가지고 있다. 그러나 복잡한 통합시스템은 다양한 구성요소(components)간의 상호관계(causal relations) 때문에 고장난 구성요소가 전체 시스템의 형태(behavior)에 미치는 영향을 파악하기는 쉬운 일이 아니다. 이 때문에 시스템 고장을 검출하고 그 원인을 제거하는 일은 더욱 어려운 일이다.

본 논문에서는 Petri-Net를 이용한 고장 감지 방법을 제안하고 이를 통하여 진단을 수행할 수 있는 2-level 구조의 진단시스템을 제시한다. 그림 6에 그 구조를 나타내었는데 하위 수준은 진단 기능을 수행하는 지능형 로컬 컨트롤러에 의해 제어되는 기계, 로봇, 그리고 컨베이어 등으로 구성되고, 상위 수준은 물류를 처리하고 로컬 컨트롤러와의 통신을 수행하며 시스템 수준의 진단 및 복구를 수행하는 시스템 컨트롤러로 구성되어 있다. 따라서 모니터링 업무는 2-level에서 각각 수행될 업무들로 분리될 수 있다.

하위 시스템의 모니터링은 로컬 컨트롤러에 의하여 수행된다. 이는 연속 상태 모니터링과 장치(devices) 상태(condition) 모니터링을 포함한다. 연속 상태 변수는 절삭 공정에 관련된 센서의 신호, 터치 프로우브 센서로부터 얻을 수 있는 작업물 및 공구의 기학학적 데이타 등이며, 장치의 상태는 기어 박스의 기계적 효율, 공구의 상태 및 기계의 진동과 같은 정보다. 상위 수준의 모니터링은 감지된 고장에 대한 로컬 컨트롤러의 진단 정보의 관리, 부품의 이송 및 추적, 진단 및 복구를 포함한다. 또한 상위 수준의 컨트롤러는 하위 시스템의 고장 발생시 재스케줄링 및 정비 계획의 기능을 포함한다.

본 논문에서는 시스템 수준, 즉 상위 수준에서의 지능형 컨트롤러의 구축 방안을 제시한다. 이는 1) 부품의 흐름을 모니터링하고 로컬 컨트롤러와의 통신을 수행하는 모듈 2) 사건의 완료를 계획하는 미래사건 스케줄링 모듈 3) 수집한 운영 데이터와 미래사건 스케줄링 으로부터 예측된 데이터와의 비교를 통해 고장을 검출하고 회복(recovery)절차를 수행하는 진단 모듈로 구성된다. 각 Petri-Net에 포함되어 있는 신호 입/출력 place들은 시스템 수준의 제어시스템과 로컬 컨트롤러간의 통신 상황을 모델링하기 위하여 사용된다. 이들이 실시간으로 응용될 때, 시스템의 생산장치(Device)들과 정보를 주고 받음으로써 사건(Event)들의 생성, 진행, 종말 등을 추적할 수 있게 한다. 실제적으로 본 논문에서는 사건의 지속 시간이 정규분포를 이룬다고 가정한다. 이러한 사건들의 수행 시간의 평균 및 표준편차는 표준 추정 방법을 이용하여 운영 데이타로부터 결정될 수 있다.

그림 5의 Cell CTPN에서 한 쌍의 신호 입력/출력 place들로 표현된 사건들의 지속 시간이 평균 m_i와 표준편차 s_i를 따르는 정규분

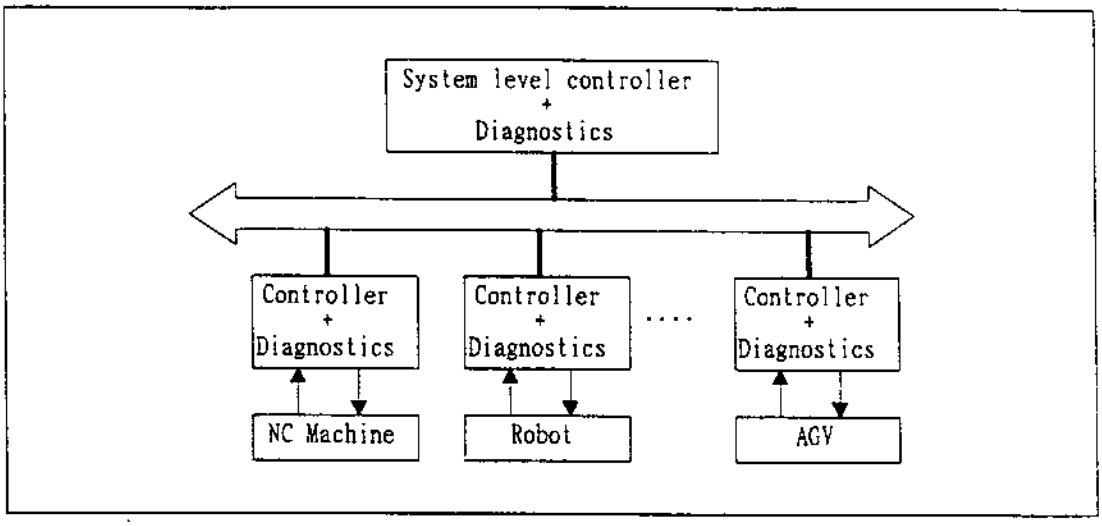

그림 6. 2-level 진단 시스템의 구조

포로 가정하자. 감지하고자 하는 시스템의 중요도에 따라 트랜지션의 실행 시간 범위는 변경될 수 있다. 만약 사건의 지속 시간이 거의 고정적이면, 오류의 감지를 위하여 상한치와 하한치를 설정한다. 한 사건이 시작되면 미래사건 목록에 (m+2s) 시점에서의 실행 완료를 스케줄링 한다. 그 후 토큰이 해당되는 신호 입력 place에 (m-2s)시점과 (m+2s)시점 사이에 도착하면 시스템은 이상 없이 정상적으로 운영되고 있다고 판단하고 그렇지 않다면 시스템에 이상(구성요소의 고장)이 발생한 것으로 판단한다. 고장이 감지 되면 로컬 컨트롤러에 제어 메시지를 송신하여 복구 절차를 수행하게 된다. 시스템 컨트롤러의 고장 감지 및 진단 모듈은 로컬 컨트롤러의 신호에 따라서 시스템 수준의 진단을 수행한다. 이러한 과정을 그림 7에 제시하였다. 실제 진단은 Fault tree나 전문가 시스템 등을 이용하여 수행될 수 있다.

5. 시뮬레이터 개발 및 실험 결과

실시간 컨트롤에 사용될 Petri-Net은 생산 자원에 직접 연결되어 센서 데이타를 읽거나 NC 프로그램을 전송하는 역할을 수행한다. 이와 같은 Petri-Net으로부터 외부와의 접속점을 없애고 set-up 시간과 작업 시간을 예상하여 시뮬레이션 clock을 이용하면 시뮬레이션용 Petri-Net을 만들 수 있다. 이는 시스템을 실제적으로 운영하는 동안 통계 데이타가 동시에 수집될 수 있음을 의미한다. 또한 전술한 바와 같이 시뮬레이션 및 실시간 운영의 결과를 비교함으로써 시스템 작용(behavior)의 이상 여부를 판단할 수 있다. 본 논문에서는 SUN sparc 워크스테이션의 X Window 환경에서 C언어를 이용한 시뮬레이터를 개발하였고 반복 실험을 통하여 설계된 제어 로직을 검증하고 시스템의 수행도 분석을 수행하였다.

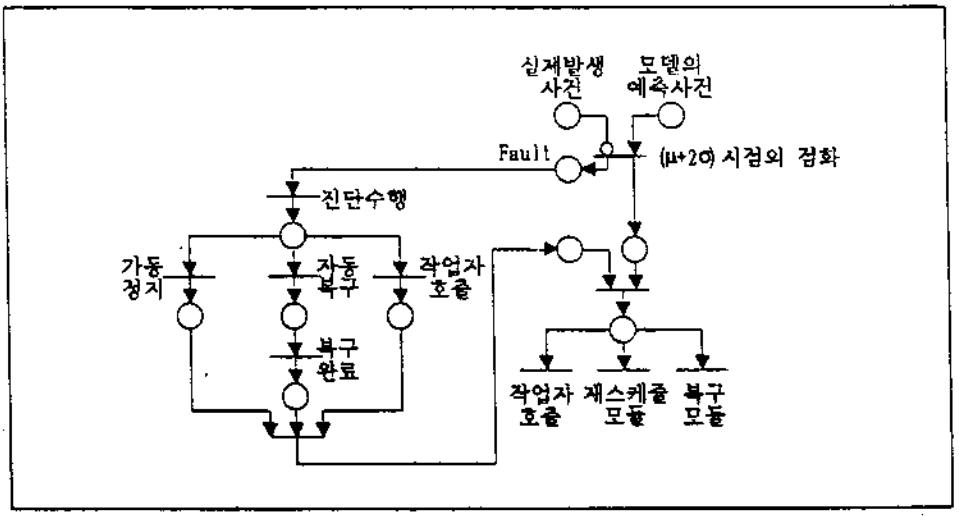

그림 7. 고장 감지 및 진단 모듈

실험을 위한 가공셀은 그림 8에 나타내었듯이 CNC 선반, CNC Turning 머신, 한대의 로봇, 그리고 6개의 부품을 저장할 수 있는 버퍼로 구성되어 있다. 본 연구에서 개발된 셀컨트롤러는 여러 가지 Dispatching rule들의 수행능력을 평가할 수도 있다. 이러한 시뮬레이션 절차를 통해 계산되는 Dispatching rule들의 수행능력 분석을 통하여 사용자는 실제 셀을 구축하고 운용함에 있어 가장 좋은 수행도를 제공하는 Dispatching rule을 선택할 수 있게 된다. 수행도 분석을 위하여 표 2에 제시된 것과 같이 다섯 개의 다른 부품들로 이루어진 Batch 작업을 고려한다. 각 Batch는 1) 가공이 수행될 머신 번호 (선반: 1, Turning 머신:2) 2) Loading 시간 3) 가공 시간 4) Unloading 시간으로 구성된 가공 정보를 필요로 한다.

본 논문에서 사용한 Dispatching rule들은 다음과 같다 : FIFO(First In First out), SPT (Shortest Processing Time), SRPT(Shortest remaining processing time), LRPT(Longest remaining processing time), FORP(Fewest operation remaining of each part), LORP (Largest operation remaining of each part). 각각의 Dispatching rule에 대한 시뮬레이션을 수행함으로써 로봇, 선반, 그리고 Turning 머신의 이용 상황(utilization) 및 부품의 흐름 상황을 알아볼 수 있다. 표 3은 LRPT Rule을 적용하여 시뮬레이션을 수행한 결과를 나타내는데, Make-span이 61초임을 알 수 있다. 표 4에 여섯 가지 Dispatching rule에 대하여 시뮬레이션 결과를 나타내었다.

수행도 분석과 더불어 제어 및 고장 진단도 셀컨트롤러의 모니터링 윈도우를 통하여 시뮬레이션할 수 있다. 이를 LRPT 규칙을 표 2의 가공 정보에 적용하여 간단히 설명하면 다음과 같다. 시스템의 수행이 시작되면 셀컨트롤러는 LRPT 규칙에 따라 가장 긴 가공시간(19초)를 갖는 부품 5를 선반으로 load하라는 신호를 보낸다. 2초 후에 loading이 완료되면, 셀컨트롤러는 NC 프로그램을 선반에 다운로드하여 13초간 가공을 수행한다. 다음

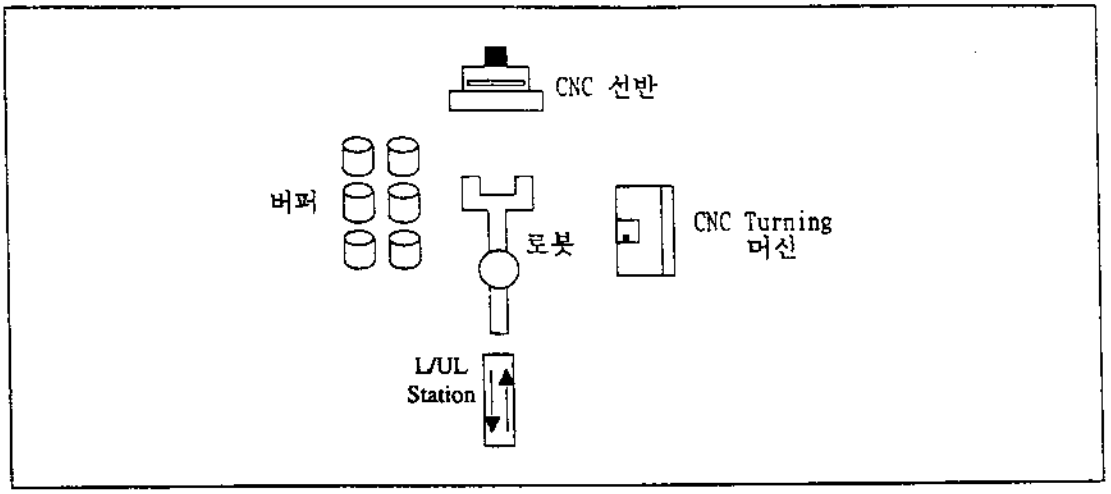

그림 8. 실험을 위한 가공셀

표 2. 고려한 부품의 가공 정보

부품번호	기계순서	Loading 시간	가공시간	Unloading 시간
1	2-1	2-2	5-8	2-2
2	2-1	2-2	9-6	2-2
3	2-1	2-2	11-4	2-2
4	1	2	8	2
5	1-2	2-2	13-6	2-2

표 3. LRPT 규칙 적용 시의 실험 결과

부품	대기시간	가동시간	총지속기간
1	40	21	61
2	6	23	29
3	26	33	59
4	29	12	41
5	28	27	55

으로 가장 긴 가공시간(15초)을 갖는 부품 2를 Turning 머신으로 load하여 가공을 시작한다. 진단 과정의 시뮬레이션을 위하여 시뮬레이션 시작 4초 후에 선반의 고장을 의미하는 상황을 만들 수 있다. 만약 처리에 28초가 소요된다면, 셀컨트롤러는 가공을 위하여 부품 5를 선반에 다시 load하도록 로봇에 신호를 보낸다. 더 이상 에러가 발생하지 않는다고 하면, 셀컨트롤러는 모든 작업을 93초 시점에서 완료한다. 시뮬레이션 결과를 보면 선반의 고장으로 인하여 Make-span이 61초에서 93초로 증가하였음을 알 수 있다. 또한, 장비의 이용 상황이 악화되더라도, 시스템 일부의 고장으로 인하여 시스템 전체의 가동이 중지되지 않는다는 것을 알 수 있다. 이를 통하여 셀컨트롤러가 에러 복구의 능력

표 4. 각 Rule 적용 시의 부품흐름 상황

Dispatching rule	부품의 흐름			Make Span
	로봇	선반	Turning	
FIFO	1, 4, 1, 2, 4, 1, 2, 3, 1, 2, 2, 5, 3, 5, 3, 5, 3, 5	4, 1, 2, 5, 3	1, 2, 3, 5	65
SPT	1, 4, 1, 2, 4, 1, 2, 3, 1, 2 ,2, 5, 3, 5, 3, 5, 3, 5	4, 1, 2, 5, 3	1, 2, 3, 5	65
SRPT	4, 1, 1, 2, 4, 1, 2, 3, 1, 2, 2, 5, 3, 5, 3, 5, 3, 5	4, 1, 2, 5, 3	1, 2, 5, 3	67
LRPT	5, 2, 2, 3, 5, 2, 2, 4, 3, 1, 4, 3, 1, 5, 3, 1, 5, 1	5, 2, 4, 3, 1	2, 3, 1, 5	61
FORP	4, 1, 1, 2, 4, 1, 2, 3, 1, 2, 2, 5, 3, 5, 3, 5, 3, 5	4, 1, 2, 5, 3	1, 2, 3, 5	67
LORP	1, 5, 1, 2, 5, 1, 2, 3, 1, 2, 3, 5, 2, 3, 5, 3, 4, 4	5, 1, 2, 3, 4	1, 2, 3, 5	62

을 제공함을 알 수 있다.

6. 결론 및 추후 연구 과제

컴퓨터 통합 생산시스템의 주요한 부분인 FMS를 구축하는데 있어서 가장 큰 난점은 특성이 서로 다른 기기들간의 인터페이스와 다양한 정보를 이용한 시스템의 실시간 제어이다. 본 논문에서는 1) 셀단위 생산시스템을 호스트 컴퓨터에서 실시간으로 제어할 수 있는 계층적 제어 CTPN을 설계하였고, 2) 시스템 구성 장비들의 고장을 감지하고 진단할 수 있는 방법론을 제시하였으며, 그리고 3) Petri-Net를 이용하여 설계된 제어로직을 테스트하기 위한 시뮬레이터를 개발하였다.

AGVs의 셀 간 이동 및 셀에서의 부품 가공과 같은 Timed events를 시뮬레이션하기 위하여 signal place를 이용하였으며, 이를 통하여 호스트와 셀 컨트롤러와의 통신 상황을 표현하였다. 그리고 우선순위가 정해지지 않은 상충 현상을 해결하기 위하여 Dispatching rule를 도입하여 부품의 가공 경로에 대한 스케줄링 문제를 해결하였다. 이는 체계적인 모델링의 장점을 살리면서 상황 변화에 즉시 대처할 수 있는 스케줄링의 기능을 제공한다. 또한, 본 논문에서는 무작위로 발생하는 고장에 대처하기 위하여 모델링 domain에서 고장을 감지하고 복구할 수 있는 새로운 방법을 제안하였다. 이를 통하여 watchdog와 같은 하드웨어가 아니더라도 고장에 탄력성 있게 대처할 수 있는 제어시스템의 개발이 가능함을 보였다.

그러나, 본 논문에서는 Petri-Net를 이용하여 FMS의 계층적 제어 시스템을 개발하기 위한 초기 단계로서 모델링 및 시뮬레이션에 중점을 두었기 때문에 여러 가지 추후 연구 과제가 제기될 수 있다. 1) 모델링에 사용된 신호 입/출력 place의 통신 상황을 실제 하드웨어의 입/출력 포트와의 인터페이스를 통하여 구현해야 한다. 2) Petri-Net 모델에 자동 에러 복구 기능이 추가되어야 한다. 3) 부품의 분리(decomposition) 및 조립(assembly) 작업을 지원해야 한다.

참 고 문 헌

[1] Ben-Arieh, D. and Miron, I., 1991, "Concurrent modeling and simulation of reactive manufacturing systems using Petri nets", *Computers and Industrial Engineering*, Vol.20, No.1, pp.45-5.

[2] Cecil, J. A., Srihari, K. and Emerson, C. R., 1992, "A Review of Petri-Net Applications in Manufacturing", *International Journal of Advanced Manufacturing Technology*, Vol.7, pp.168-177

[3] Cossins, R. and Ferreira, P., 1992, "Celeritas: a colored Petri net approach to simulation and control of flexible manufacturing systems", *International Journal of Production Research*, Vol.30, No.8, pp.1925-1956.

[4] Dunkler, O., Mitchell, C. M., Govindraj, T., and Ammons, J. C., 1988, "The effectiveness of supervisory control strategies in scheduling flexible manufacturing systems", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol.18, No.2, pp. 223-237.

[5] Huang, Han-Pang and Chang, Po-Chu, 1992, "Specification, modeling and control of a flexible manufacturing cell", *International Journal of Production Research*, Vol.30, No.11, pp.2515-1543.

[6] Jensen, K., 1990, "Coloured Petri Nets: A High Level Language for System Design and Analysis", *Lecture Notes in Computer Science*, Vol.483, pp.342-416.

[7] Knapp, G.M. and Wang, Hsu-Ping, 1992, "Modeling of Automated Storage/Retrieval Systems Using Petri Nets", *Journal of Manufacturing Systems*, Vol.11, No.1, pp.20-29.

[8] Liu, Chin-Ming and Wu, Feng-Cheng Wu, 1993, "Using Petri nets to solve FMS problems", *International Journal of CIM*, Vol.6, No.3, pp.175-185.

[9] Matsuura, H., Tsubone, H., and Kanezashi, M. , 1993, "Sequencing, dispatching, and switching in a dynamic manufacturing environment", *International Journal of Production Research*, Vol.31, No.7, pp. 1671-1688.

[10] Menon, S.R., Ferreira, P.M., and Kapoor, S.G., 1990, "A Colored Petri Net System for Simulation and Control of FMSs", *Advances in Manufacturing Systems Engineering*.

[11] Martinez, J., Muro, P. and Silva, M., 1987, "Modeling, Validation and Software Implementation of Production Systems Using High Level Petri Nets", *Proceedings of IEEE International Conference on Robotics and Automation*, Raleigh (USA), pp.1180-1185.

[12] Murata, T., 1984, "Modeling and Analysis of Concurrent Systems", *Handbook of Software Engineering*, Chapter 3, Vick, C. R., and Ramamoorthy, C. V., eds, Van Nostrand Reinhold, Network.

[13] Matsuura, H., Tsubone, H., and Kanezashi, M., 1993, "Sequencing, dispatching,

and switching in a dynamic manufacturing environment", *International Journal of Production Research*, Vol.31, No.7, pp.1671-1688.

[14] Peterson, J. L., 1981, *Petri Net Theory and Modeling of Systems*, Prentice-Hall, Inc.

[15] Ramchandani, C.V.and Ho, G.S., 1980, "Performance Evaluation of Asynchronous Concurrent Systems using Petri Nets", *IEEE Transactions on Software Engineering*, Vol.SE-6, No.5, pp.440-449.

[16] Sahraoui, A., Atabakhche, H., Courvoisier, M., and Valette, R., 1987, "Joining Petri nets and knowledge based systems for monitoring purposes", *Proceedings of IEEE International Conference on Robotics and Automation*, Vol.2, pp.1160-1165.

[17] Valavanis, H.P., 1990, "On the Hierarchical Modeling Analysis and Simulation of Flexible Manufacturing Systems with Extended Petri Nets", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 20, No.1, pp.94-110.

[18] Viswanadham, N., and Narahari, Y., 1992, *Performance Modeling of Automated Manufacturing Systems*, Prentice Hall, Inc.

[19] Zhou, M.C. and Dicesare, F., 1989, "Adaptive Design of Petri Net Controllers for Error Recovery in Automated Manufacturing Systems", *IEEE Transactions on Systems, Man, Cybernetics*, Vol.19, No.5, pp. 963-973.