대화식 생산일정계획 수립을 위한 객체 지향형 시스템 설계[1)]

(Design of an Object-Oriented System for Interactive Production Scheduling)

김 승 권*, 김 선 욱**, 이 준 열*, 홍 윤 호*
고려대학교 산업공학과*
단국대학교 산업공학과**
Sheung Kown Kim*, Sun Uk Kim**, Jun Yeol Lee*, Yun Ho Hong*
Dept. of Industrial Engineering, KOREA University*
Dept. of Industrial Engineering, Dankook University**

Abstract

의사결정 지원 시스템의 성패 여부는 사용자가 사용하기에 편리하고 효과적인 모형으로 구축되었는가에 달려 있다. 아무리 훌륭한 시스템 분석 기법을 활용하였다 하더라도 사용자가 불편하여 사용하지 않으면 그 의사결정 지원 시스템은 사장되고 말 것이며 많은 시간과 정성을 들여 개발한 연구결과가 빛을 보지 못할 것이다.

본 연구에서는 일반적인 중소기업의 제조업 환경에서 생산 일정 계획 수립을 위한 의사결정 지원 시스템의 구축을 중심으로, 사용자 편의를 고려한 시스템이 갖추어야 할 사양 및 이에 대한 설계, 객체지향 프로그래밍 기법의 활용, 시스템 구현상의 문제점 등에 대하여 구체적으로 설명하고 해결방안에 대하여 고찰한다.

I 서 론

1. 객체지향 프로그래밍(Object-Oriented Programming)의 소개

"객체지향 프로그래밍(object-oriented programming)"이라는 용어는 1967년 "Simula-67"이라는 프로그래밍 언어를 소개하면서 처음 사용되었다. 그러나 정식적인 의미에서의 객체지향 프로그래밍 기법에 관한 소개는 1970년 중반 Xerox 사에서 "Smalltalk"를 발표하면서부터 널리 알려지게 되었다. Xerox 사에서 1980년 "Smalltalk-80"을 최초로 일반 프로그래머에게 공개하면서부터 객체지향 프로그래밍 기법에 관한 연구가 활발하여졌다. 그 이후 "Smalltalk"의 특징을 일부 추출하여 C programming 언어와 객체지향 언어를 통합한 새로운 형태의 언어인 C++이 발표되었다. 그 중에서 UNIX 환경에서 적용되는 AT&T C++, Zorteck C++와, IBM PC 환경에서 적용되는 MS C++ 및 Borland C++ 등이 발표되었다. 이 외에 이러한 객체지향 프로그래밍 기법을 적용하여 본 다른 프로그래밍 언어로는 CLOS, ADA, LISP 등이 있다[9,13].

객체지향 프로그래밍의 주요 특성으로는 정보 은닉(information hiding), 데이터 추상화(data abstraction), 동적 바인딩(dynamic binding), 특성 계승(inheritance) 등을 들 수 있다[5,7,12]. 이러한 특성들은 객체지향 언어의 개발 목적 중의 하나인 소프트웨어의 재사용성을 이루기 위한 필수요건들이다. Lamer Ledbetter와 Brad Cox는 객체지향 프로그래밍 언어로 개발한 소프트웨어 모듈을 일종의 "Software ICs"라고 부르면서, "Hardware ICs"가 하나의 단위모듈로써 전체 하드웨어 제품의 부품과 같이 자유롭게 부착하였다 뗄 수 있도록

1) 이 연구는 한국과학재단의 목적기초연구비 지원으로 이루어졌음.(과제번호 90-02-00-06)

하는 것과 같이 객체지향 프로그래밍은 사용자가 언제든지 필요한 모듈을 수정없이 그대로 사용할 수 있는 특성을 가지고 있음을 시사하였다[3].

객체지향 프로그래밍 언어의 또다른 특징 중의 하나는 시뮬레이션 묘사에 편리하다는 점이다. 객체지향 프로그래밍은 모든 사물을 객체로 정의하고 자신의 고유한 특성을 묘사한 후 객체와 객체간의 메세지를 전달하도록 설계되기 때문에, 복잡한 시스템이라 할지라도 객체들의 특성만 잘 묘사하면 사용자가 요구하는 전반적인 정보를 수월하게 수집하고 분석할 수 있다. 특히 특성계승 기법은 객체와 객체간의 수직적·수평적 관계를 묘사하기에 적합하며 이러한 기법은 사용자로 하여금 모형설계를 손쉽게 한다.

2. 기존의 일정계획 수립에 관한 연구에 관하여

지금까지 수많은 연구자들에 의해 생산일정계획 수립에 관한 많은 연구가 발표되었지만, 대개는 극히 단순한 생산 시스템 모형을 대상으로 하였다. 그러나 실제 상황은 기존의 연구에 비하여 상당히 복잡하므로 표준 모델로 수립하기 조차 어려우며, 비록 해법이 존재하더라도 NP-Complete에 해당하여 실용적인 알고리즘을 찾아 보기는 힘들다.

생산 환경이 자동화 및 FMS(Flexible Manufacturing System) 등으로 전환되어 감에 따라 최상의 일정계획을 수립하는 알고리즘 개발보다는, 유연성 있는 일정계획 수립이 가능한 소프트웨어 개발에 더 많은 관심을 가지게 되었다. 더우기 생산 관리자가 일정계획을 수립하는 데 있어서 고려하는 "goal"이 단일 항목이기보다는 다중 항목인 경우가 많기 때문에, 특정 알고리즘을 고정하여 사용하기보다는 다중의 기법을 소유하였다가 사용자가 일정계획 수립과정에서 컴퓨터와의 interface를 통하여 다중 목표를 요구수준까지 향상시킬 수 있는 소프트웨어 개발이 필요하다.

이 외에 "Real-Time Scheduling"을 수립하기 위하여 각각의 기계에 어떠한 작업 분배 규칙(dispatching rule)를 적용하여야 효율적인 일정계획을 수립할 수 있는 지 여부를 시뮬레이션을 통하여 알아 보는 연구들이 있다[2]. 최근에는 컴퓨터 그래픽스의 발전에 힘입어 GUI(Graphics-User Interface)를 통하여 사용자가 의사결정을 수월히 하도록 하는 의사결정 지원 시스템(decision support system) 개발에 관한 연구가 많이 발표되었다[4,6,8,10]. 또한 각각의 기계에게 단순한 dispatching rule들을 적용하기보다는 작업 상황에 따라 적절한 dispatching rule을 선택하도록 하는 일종의 지식기반 일정계획 전문가 시스템(knowledge-based scheduling expert system)에 관한 연구가 필요하게 되었다[1,11].

그러나 이상의 연구들은 대부분 구조적 프로그래밍에 의한 소프트웨어로 구현된 것들로, real world에서의 시스템 내의 객체간의 메세지 전달을 소프트웨어의 모형에 그대로 구현하였다고 볼 수 없다는 단점이 있다. 현재까지는 앞에서 소개한 논문들이 유기적으로 결합하여 하나의 종합 시스템으로 구현하려는 시도가 활발히 이루어졌으나, 그러한 과정을 소개하는 연구는 그리 많은 편이 아니었다.

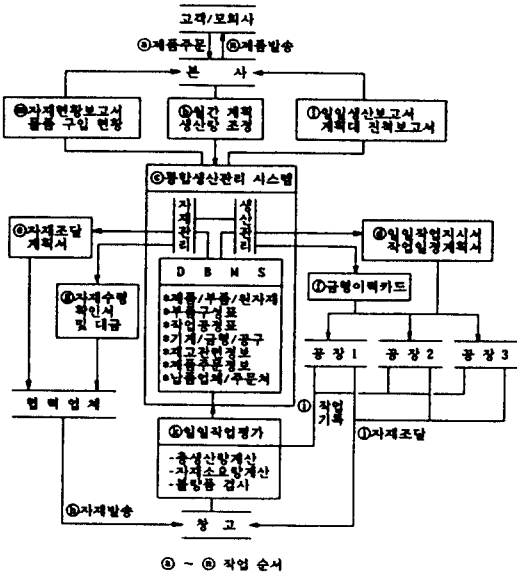
3. Object-Oriented Programming(OOP)과 Graphics-User Interface(GUI)를 이용한 생산일정계획 수립 시스템 개발

앞에서 언급하였듯이, 지금까지는 생산과정이 이루어지는 real world를 객체들의 집합으로 보고, 각각의 객체간의 유기적인 관계를 설계하는 과정을 소개하는 연구는 별로 없다. 본 논문에서는 생산 시스템 환경에서 일정계획을 수립하는 시스템을 구현하고자, OOP와 GUI를 이용하여 생산 시스템을 설계하는 일련의 과정을 소개하고자 한다.

Ⅱ 객체지향형 시스템 설계

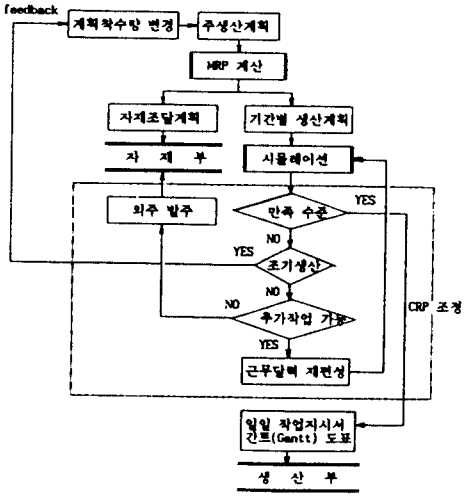
1. 생산 계획관리 업무 소개

본 논문에서는 설명을 좀 더 쉽게 하기 위해서 지금까지 본 연구의 대상업체를 중심으로 하여 시스템 설계과정을 서술하기로 한다. 본 연구에서는 중소제조업체인 R사를 연구 대상업체로 선정하여 연구가 진행되었다. 먼저 그 대상업체의 업무특성 및 요구분석이 선행되었다. 이 대상업체는 주로 주문에 의하여 생산이 이루어지며, 그 주문 발생 빈도는 규칙적인 경우와 불규칙적인 경우가 있다. 규칙적인 주문의 경우 미리 부하소요계획 및 자재소요계획을 수립해 놓을 수 있으나, 불규칙적인 주문은 사전 예측이 불가능하므로 실시간에 가까운 일정계획의 수립이 가능해야 한다. [그림 1]은 본 연구팀이 R사의 업무특성과 요구사항을 고려하여 새롭게 설계한 업무 흐름도이다.



[그림 1] 자재부/생산부 관리업무 시스템

본 연구의 범위는 [그림 1]에서 보듯이 중소기업의 생산부 업무와 자재부 업무를 통합하여 생산계획을 수립하고 각 부서별 관리업무를 수행하는 것으로 한정한다.



[그림 2] 생산계획 메인 모듈의 설계

[그림 1]에서 중앙에 위치하고 있는 통합 생산관리 시스템은 각종 생산관련정보를 관리하는 DBMS (Date Base Management System)과, 이들 정보로부터 생산일정계획 및 자재소요계획을 수립하는 메인 모듈로 구성되어 있다. 생산계획 메인 모듈은 다음과 같이 크게 3단계로 나뉘어진다[14].

- 자재소요계획을 수립하는 MRP 모듈
- OOP 기법을 이용한 시뮬레이션 모듈
- 작업부하조정을 위한 전문가 시스템(GUI 활용)

[그림 2]는 생산계획을 수립하는 메인 모듈을 나타내고 있다. [그림 1]에서 묘사된 DBMS내의 정보들은 관계형 데이터베이스에 의하여 관리되며, 부품구성표 및 공정목록표 등과 같은 계층형 구조는 메인 모듈에서 OOP를 이용한 특성계승 구조로 변환하여 관리된다. [그림 1]의 DBMS의 소개는 연구범위에 벗어나므로 여기서는 생략하기로 한다.

[그림 3]은 지금까지 발표된 MRPI 계산과정과 동일하다. 본 연구에서 사용한 방식 중 특이한 점은 자재별 부품구성표의 계층레벨이 가장 높은 것(완제품의 계층레벨 = 0)에서부터 시작하여 계층레벨이 낮아지는(계층레벨 숫자가 큰) 순으로 단위 MRP 계산을 수립하도록 하였다는 것과, 자재조달의 리드타임을 결정하는 경우, 근무 달력의 휴무 여부나, 근무 방식에 따른 일일 부하량을 고려한다는 점을 들 수 있다.

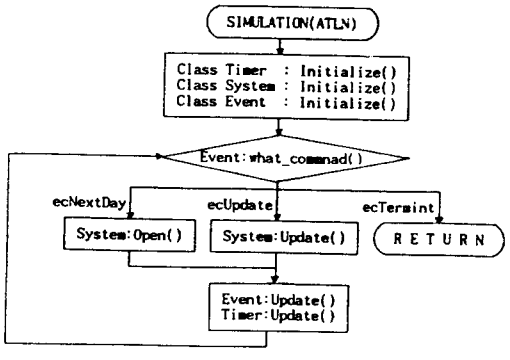
그리고 "shop floor scheduling"을 수립하기 위한 시뮬레이션을. 객체지향 프로그래밍을 이용하여 구현하고자 할 경우, 시뮬레이션 시작시간과 종료시간을 관리하는 객체와 event를 관리하는 객체, event 발생시 시스템 상태의 변화를 관리하는 객체 등의 설계는 필수적이다. [표 1]은 이러한 관점에서 객체들을 정리한 것이다. 이 3개의 객체는 시뮬레이션을 위하여 특별히 규정한 객체이며, 클래스 "System" 내에서 소개된 객체들은 앞에서 언급한 production cell 내의 객체들이다.

클래스	객 체 특 성	해 설
Timer	Instance variables:	
	current_time, max_time:	Timer의 현재 시간 / 최대 근무시간
	current_date, max_date:	Timer의 현재 날짜 / 계획 기간 최종 날짜
	Methods:	
	Initialize():	Timer 초기화
	Update():	Timer를 다음 Event 발생시점으로 이동시킨다.
Event	Instance variables:	
	elapsed_time:	현재 Timer에서 다음 Event까지의 경과 시간
	Class Event_Table	Event 정보 클래스
	Instance variables:	
	time_:	언제 (Event가 발생한 시점)
	from_:	어디서 (작업이 완료된 기계 코드)
	to_:	어디로 (새로운 작업이 수행되는 기계 코드)
	what_:	무엇을 (자재 코드)
	QTY_:	얼마만큼 (작업 물량)
	Methods:	
	Initialize():	Event 초기화
	Update():	다음으로 발생될 Event를 구한다.
	what_command():	현재의 System Status에 따라 메세지를 전달
System	Instance variables:	
	Class Inventory	재고 목록 클래스
	Instance variables:	
	CODE:	자재 코드
	QTY:	재고 수량
	Class Machine	작업 기계 클래스
	Instance variables:	
	CODE:	기계 코드
	in_process:	현재 기계가 작업중인지 아닌지를 기록
	what_item:	현재 가공중인 자재의 코드번호
	Class Work_list:	작업해야 할 일람을 기록한 리스트
	Class Queue_list:	작업에 필요한 자재들의 목록(작업대기 자재)
	Methods:	
	load_queues():	work가 존재하면 새로운 자재를 Queue_list에 등록
	restore_queues():	하루 작업이 종료되면 queue의 자재를 창고로 옮김
	load_works():	처리해야 할 작업량을 Work_list에 등록
	begin_work():	Work_list에 등록된 작업중 dispatching rule에 따라 점수를 매기고 최고 점수를 가진 작업을 처리함
	update_works():	현재일 작업은 종료하고 남은 작업은 다음 날짜의 작업으로 연기하고 다음 날짜의 시작업도 등록함
	material_arrival():	어느 자재가 얼마만큼 이 기계에 도착하는가 ?
	material_departure():	어느 자재가 얼마만큼 다른 기계로 떠나는가 ?
	get_jobs():	queue의 일량을 빼고 남은 작업량을 구한다.
	get_finish_time():	현재 작업중인 기계의 작업 종료 시점을 구한다.
	Methods:	
	Initialize():	Event 초기화
	Open():	현재 날짜의 작업을 종료->다음 날짜의 작업 개시
	Update():	다음으로 발생될 Event를 구한다.

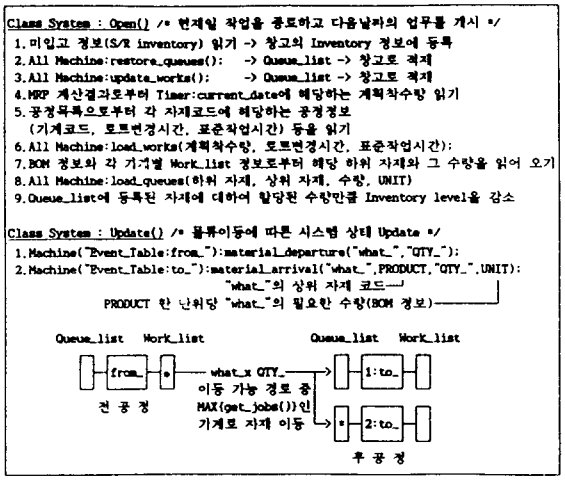
[표 1] 시뮬레이션에 필요한 객체들의 정의

[표 1]에서 소개한 객체들을 통한 시뮬레이션 전체의 흐름도는 [그림 4]와 같다. 여기서 "SIMULATION(ALTN)"의 전달 인수 "ALTN"은 각 기계별로 고유한 "dispatching rule"을 적용한다는 것을 나타내는 각각의 대안별 인덱스 번호이다.

[그림 5]는 EVENT가 발생하였을 경우, 해당 EVENT의 특성에 따른 시스템 상황 변화를 묘사하고 있다. 여기서는 클래스 SYSTEM의 내부 메세지 교환 과정을 묘사하고 있다.



[그림 4] 객체지향 시뮬레이션 흐름도



[그림 5] 객체간의 메세지 전달의 예

2. 일정계획 수립을 위한 객체지향적 시스템 설계

Eero Eloranta는 Computer-Aided Production Management System(CAPM)을 설계함에 있어서 생산 시스템 cell과 관련된 객체들을 다음과 같은 정리하였다[6].

- Data aspect : machine data, layout, production plan(graphical Gantt chart, CPM representation), customers and suppliers, materials(raw material, part, product), inventory, product structures, orders, etc.
- Application-related base operations : loading, bill-of-materials explosion and implosion, due date assignment, etc.
- General base operations - intelligent-form specification, (form) mail functions *ad hoc* database operations, etc.
- Application software related to the object
- Representation at the user interface, e.g. as an intelligent form.

본연구에서 R사의 생산 현장을 중심으로 재정리한 정보 및 객체들은 다음과 같다.

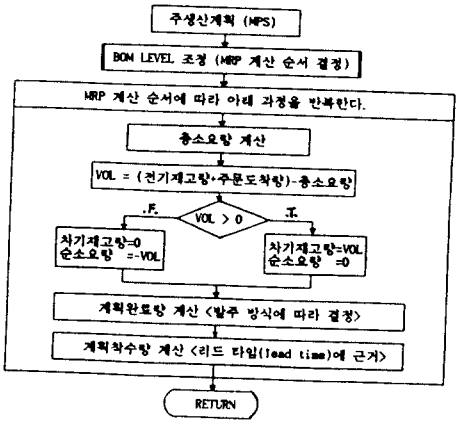
- 생산관련자료 : 자재정보(완제품/부품/원자재), 기계정보/기계배치도/금형정보, 재고정보(현재고/미입고목록), 주문정보(자재코드/수량/납기),
- 자재 및 작업관련정보 : 부품구성표, 공정목록표
- 생산운영지식 : 자재조달방식/리드타임/조달량, 작업분배규칙(dispatching rule), 작업 지연목록, 근무달력, 기계별 작업 부하 측정 등
- 사용자 인터훼이스 : GUI tools(Icons, Dialog Boxes, Operation Buttons, etc.)

본 연구팀이 구현한 시스템 모형은 단순하지만 방대한 자료들을 입력 받고 관리하기 위하여 "clipper V5.01"이라는 상용 데이터베이스 관리 팩키지를 사용하였다. 이렇게 하서 입력 받은 정보들은 [그림 2]에서 소개한 메인 모듈에 적용할 때에는 각각 객체로 규정되어 고유한 정보를 입력 받게 되어 있다. 그리고 정보량이 적은 객체들(예를 들어 기계나 작업장 등의 객체)은 GUI tool의 icon에 의해 생성되도록 설계하였다.

특히 부품구성표와 같은 제품의 계층구조가 명확한 자료들은 단순히 관계형 데이터베이스만으로 자재간의 계층성을 묘사하기 부적합하다. 따라서 객체지향 데이테베이스(object-oriented database)와 같이 특성계승을 구현할 수 있는 새로운 자료관리 시스템이 필요하게 되었다[13].

본 연구에서도 부품구성표의 경우는 자재간의 계층구조 특성을 활용하기 위하여 객체지향적 특성을 활용하였다. 예를 들어 자재 A는 자재 B와 C에 의해 만들어진다고 할 때, 자재 A를 제품 X나 Y의 부품이라고 규정하게 되면 자동적으로 자재 B와 C도 제품 X와 Y의 부속품이라는 사실이 유도되도록 설계하였다.

[그림 2]의 메인 모듈에서의 맨처음 단계에서는 초기 생산계획을 수립하기 위한 MRP 계산이 수행된다. [그림 3]은 MRP 계산을 하기 위한 기본 모듈이다.



[그림 3] MRP 계산 모듈

3. 부하소요계획 수정을 위한 지식기반 의사결정 지원 시스템의 설계

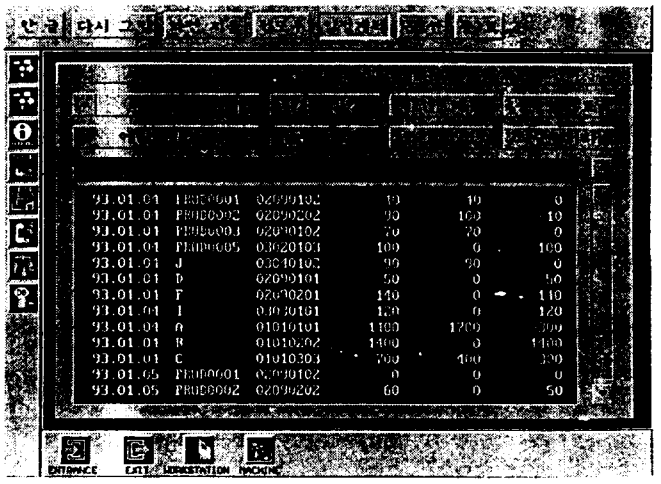
이 과정은 [그림 2]에서 점선으로 묘사한 부하계획(CRP:Capacity Requirement Planning) 조정을 담당하는 과정이다. 생산일정계획 수립에 있어서 생산 관리자의 일반적인 관리 중점은 납기 준수와 부하 평준화를 들 수 있다.

작업 지연으로 인하여 납기를 만족하지 못하는 상황에 이르렀을 때, 가장 먼저 살피게 되는 것이 계획기간 동안의 부하 평준화 여부이다. 즉, 작업이 특정기간 동안 몰려 있는 경우 필연적으로 작업 지연이 발생하게 되는데 이를 극복하는 가장 우선적인 방법이 작업 부하를 가급적이면 평준화시켜 작업이 몰리게 되는 상황을 분산시켜 보는 방법을 적용할 수 있다.

그 다음으로 처리할 수 있는 대책으로는 작업 근무시간을 연장시켜 보는 방법을 채택할 수 있다. 여기서 주의해야 할 점은 대부분의 생산업체가 고유한 근무시간표와 근로 규정, 근무달력 등을 소유하고 있기 때문에 일정계획 수립시 이러한 사실들을 고려하여야 한다. 본 연구에서는 근무 방식을 정상근무, 잔업근무, 철야근무, 오전근무, 휴무 등으로 분류하고 각각의 근무 형태에 따라 고유한 근무시간표를 입력할 수 있도록 설계하였다.

조기 생산을 통한 부하 평준화 방법과 작업 근무시간을 연장하는 방법을 적용하기 어려운 상태에 있는 경우, 마지막으로 취할 수 있는 조치가 바로 "외주 처리"하는 방법이 있다. 이는 도저히 현재 상황으로는 납기를 만족시킬 수 없을 때, 자사에서도 생산이 가능한 제품/부품을 다른 생산업체에게 납기일까지 생산하도록 하는 조치이다.

[그림 6]은 부하조정을 위한 지식기반 의사결정 지원 시스템(Decision Support System:DSS)을 묘사하고 있다. 여기서 각각의 버튼과 아이콘들은 객체지향 프로그래밍 기법에 의거하여 만든 객체이다. "조기생산" 버튼을 선택하면 각 날짜 별로 각 자재의 부족량을 출력해주며, 자재별 작업완료시간과 조기생산시간을 출력해준다. 사용자가 마우스로 조기생산하고자 하는 자재를 선택하면 사용자가 원하는 조기일짜로 생산착수시점이 변경된다. "외주처리" 버튼을 선택하면 작업지연 항목중에서 외주처리가 가능한 항목들이 정렬된다. 이중에서 사용자가 선택하는 항목만이 외주처리가 가능하다.



[그림 6] 부하조정을 위한 DSS 설계

"근무달력"은 작업이 지연된 자재가 발생한 날짜들을 정렬한 후, 각 날짜 별로 근무 형태(정상근무/잔업근무/철야작업/오전근무/휴무)를 선택할 수 있도록 설계하였다.

여기서 각 자재들 중 작업지연이 발생하는 자재가 발생하면 그 자재의 하위 자재들 중 작업지연을 유발하는 자재들을 부품구성표(bill of materials)와 공정목록표(operation list), 주문정보, MRP(Material Requirement Planning) 정보를 검색하여 사용자에게 제시한다.

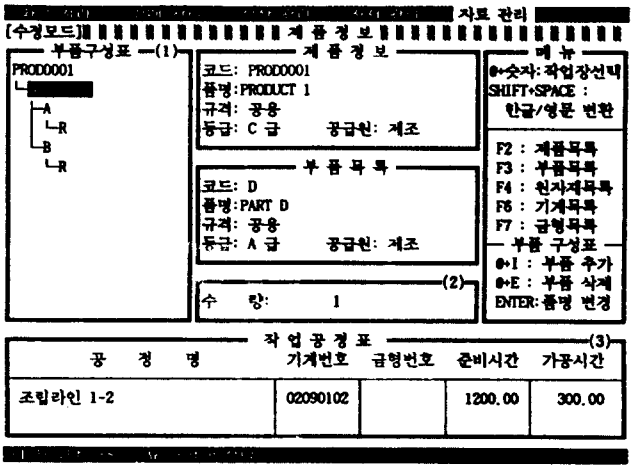
현재로서는 시스템 자체내에서 문제점이 발생하였을 경우, 이를 내부적으로 조치하는 지식을 설계하지 않았지만 작업처리 우선 규칙들을 추가한다면 이 부분은 충분히 전문가 시스템으로 구현될 수 있을 것이다.

III 시스템 운영

1. 자료입력

본 시스템에서 생산계획 수립을 위하여 입력받아야 할 생산관련정보 및 객체들을 정리하면 다음과 같다.

- 자재정보(완제품/부품/원자재)
- 기계정보/기계배치도/금형정보
- 재고정보(현재고/미입고목록)
- 주문정보(자재코드/수량/납기)
- 부품구성표
- 공정목록표
- 자재조달방식/리드타임/조달량
- 작업분배규칙(dispatching rule)

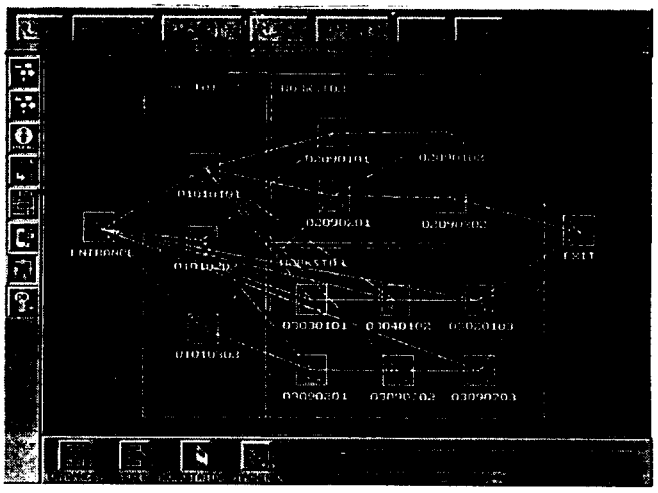


[그림 7] 부품구성표 입력

[그림 7]은 앞에서 서술한 생산관련 정보들 중에서 부품구성표를 작성하는 과정을 묘사하고 있다.

[그림 8]은 입력된 생산관련정보와 공정목록표를 컴퓨터 그래픽스를 이용하여 검증하는 과정을 묘사하고 있다.

여기서는 생산되는 모든 제품의 물류흐름을 한꺼번에 출력한 것으로, 각각의 제품 별로 하나씩 출력할 수도 있다. 이러한 기능은 사용자가 문서형태로 정보를 입력하였을 경우, 정보입력의 오류를 시각적으로 검사할 수 있게 한다.

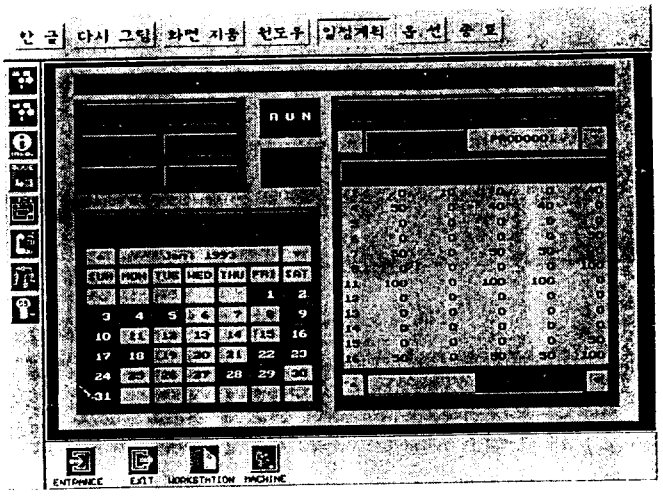


[그림 8] 물류흐름 검사

그리고 생산 현장을 GUI를 이용하여 화면상에 그대로 묘사할 수 있게 하기 위하여 작업장 정의 및 기계 생성 및 배치를 아이콘 조작으로 가능하게 하였다. 본 시스템은 IBM PC 환경에서 개발되었으며, 화면 크기의 제약과 RAM 메모리의 제약으로 작업장은 최대 8개까지 생성이 가능하며 기계는 최대 100개까지 가능하도록 제한하였다. 따라서 이 소프트웨어는 중소기업의 생산 현장에 적합할 것으로 기대된다.

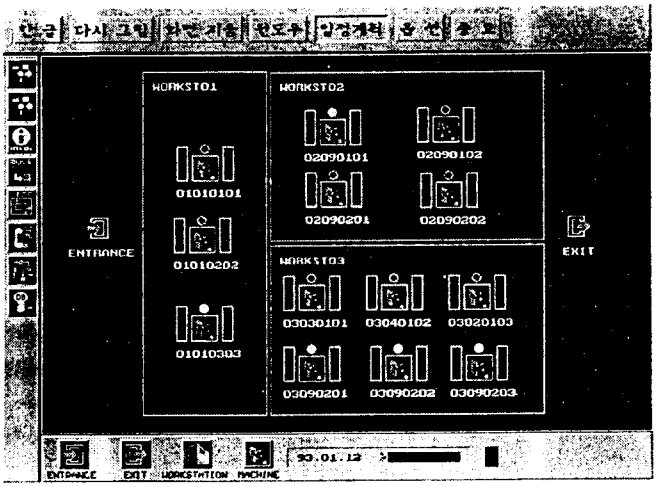
2. 일정계획 수립

[그림 9]는 1 절에서 수집한 정보로부터 MRP 기법을 통하여 초기 자재수급계획을 수립한 과정을 보여주고 있다. 여기서 본 연구팀이 MRP 기법을 적용하는데 있어서 고려한 항목 중의 하나는, 자재별 부품구성표의 계층레벨의 우선 순위에 따라 MRP 계산을 수립하도록 하였다. 또 하나는 근무 달력의 휴무 여부, 근무 형태에 따른 일일 부하량을 고려하여 MRP의 리드타임을 고려하였다는 점을 들 수 있다.



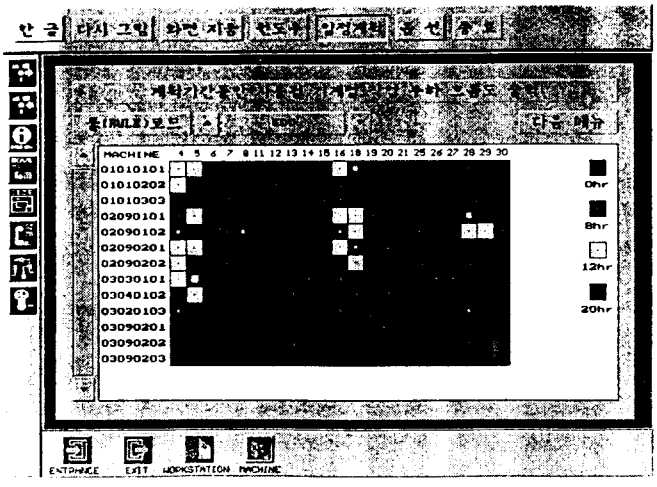
[그림 9] 자재소요계획 수립(MRP 계산)

[그림 10]은 앞에서 계산한 MRP 정보를 기준으로 shop floor planning을 수립하기 위하여 simulation을 수행하는 과정을 묘사하고 있다. 여기서는 사용자의 계획기간동안의 각 기계별 업무상황과 물류흐름의 이해를 돕기 위한 애니메이션 기능을 첨가시켰다. 기계의 왼쪽에 있는 박스에는 그 기계에서 필요로 하는 자재량이, 오른쪽의 박스에는 그 기계가 처리하여야 하는 작업량이 할당되어 있다. 기계 위에 있는 원은 램프를 묘사한 것으로, 기계의 작업 여부에 따라 램프가 점멸된다.



[그림 10] simulation & animation 과정

[그림 11]은 계획기간동안의 각 기계별 소요 부하량을 그래픽으로 묘사한 것이다. 수평방향으로는 한 기계가 계획기간동안 수행하여야 할 근무패턴들이 표시되며, 수직방향으로는 특정 날짜에서 각 기계들의 근무 소요시간을 묘사하고 있다. 이를 통하여 관리자는 특정 기계(작업자)에게 일감이 집중되는 현상을 쉽게 파악할 수 있어 현장 작업자의 업무계획을 조정해 볼 수 있다.

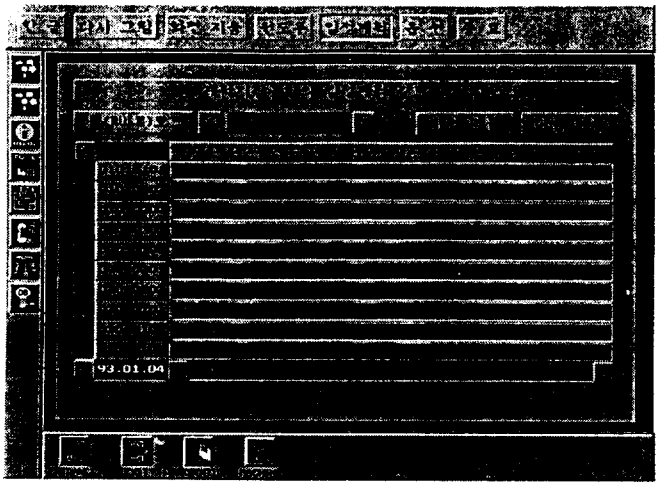


[그림 11] 작업부하 흐름도

3. 최종 결과 출력

[그림 11]는 [그림 5]에서 소개한 부하조정 모듈을 거친 후, 최종적으로 출력되는 Gantt Chart이다.

현재 개발한 소프트웨어는 각 날짜별 근무형태에 따라 근무 시간이 결정되도록 설계하였으며, 각각의 근무형태에는 고유한 근무시간표에 따라 일정계획표를 수립할 수 있도록 구현되었다. 아직은 근무자별로 각각 고유한 근무계획을 가질 수 있도록 하는 기능은 구현되지 않았다. 다만 여기서는 모든 근로자가 원칙적으로 동일한 근무시간에 의거하여 작업을 수행하도록 하였다.



[그림 12] Gantt Chart 출력

Ⅳ 결 론

지금까지 대화식 생산일정계획 수립을 위한 객체지향 시스템 설계에 관하여 살펴 보았다. 생산일정계획을 수립하는 소프트웨어를 개발하는 데 있어서 본 연구팀이 객체지향 프로그래밍 기법을 사용하는 이유는 바로 "객체지향 프로그래밍의 높은 이식성" 때문이다. 즉 OOP는 필요할 때 언제든지 필요한 객체를 추가하기 쉬우며, 그 객체를 추가하더라도 서로가 독립성을 유지하기 때문에 전체 프로그램에 미치는 영향이 다른 프로그래밍 언어에 비하여 상당히 적다는 장점이 있다.

그러나 객체지향적으로 모델을 설계한 것을 그대로 프로그래밍으로 옮기는 과정은 프로그래머에게 있어서는 오히려 상당한 어려움을 주고 있다. 그 이유는 먼저 객체들의 고유 특성을 정의하고 객체간의 특성계승 관계를 규명한 후, 객체간의 상호관계를 묘사해야 하기 때문에 프로그래밍 작업보다도 설계작업이 더욱 힘들어지기 때문이다. 그리고 각각의 프로그래밍 모듈은 서로 독립성을 유지할 수 있어야 하는데 이러한 작업은 결코 쉽지 이루어지지 않는다. 또한 객체지향형 프로그래밍은 사용자가 이용할 수 있는 program library(=object)가 존재하고 있을 경우에는 소프트웨어 개발이 쉽지만 아무런 library가 존재하지 않으면 이들을 먼저 개발한 후에 프로그래밍을 하여야 하므로 큰 부담이 따른다.

본 연구에서 구현된 시스템의 문제점으로는 생산 계획을 수립한 상황과 실제 상황이 다른 경우 이를 즉석에서 생산 계획을 수정할 수 있는 기능이 부족하다. 주어진 일정계획 기간동안의 생산계획을 미리 수립한 후에, 어떤 협력업체가 재 납기일까지 자재를 조달하지 못하였을 경우, 또는 고객의 특별 요구에 의하여 납기가 앞당겨졌을 경우, 계획기간 동안의 생산계획 전부를 재수립할 필요없이 기존의 Gantt Chart를 "Graphic Editor"로 직접 편집할 수 있으면 생산 관리차원에서도 중요한 역할을 하게 될 것이다.

마지막으로 본 연구를 통하여 구현된 시스템은 FMS 환경하에서의 생산계획 및 관리 모형으로 쉽게 확장시킬 수 있다. 그런데 현재 이 시스템은 물류 운반자를 객체로 취급하고 있

지 않고 단지 운반 시간이라는 정보만 활용하고 있다. 따라서 이 시스템을 FMS와 같은 자동화 시스템에 적용하려면 "물류 운반 시스템"을 설계가 선행되어야 한다. 즉 시뮬레이션 모듈에 물류 운반자 객체를 설계하여야 할 것이다. 그리고 "real time scheduling"을 수립하기 위해서는 기계의 작동 여부를 메세지로 고려해야 한다. 만일 기계가 고장이 나면 그 기계를 수리할 때까지 그 기계에 영향을 받는 시스템내의 모든 물류를 특별 통제할 수 있어야 할 것이다.

지금까지 소개한 내용은 중소기업이 제품을 주문받았을 때부터 생산일정계획을 수립할 때까지의 일련의 작업을 조직적으로 수행하는 과정을 묘사하였다. 물론 본 시스템이 생산 현장의 문제점들을 모두 다루지는 못하고 있으나, 이 연구 결과를 바탕으로 계속 연구를 수행하면 현장 문제를 어느 정도 해결할 수 있으리라 기대된다.

앞으로 본 연구를 기반으로 하여 가급적이면 "real-time scheduling"이 가능한 생산계획/관리 시스템으로 발전시키고자 한다. 아울러 통합 생산관리 시스템에 내부적인 통계 처리 기능, 공학적인 지식 함유(예를 들면 표준 작업시간 계산, CPM 관리, 안전재고 계산 등)를 통하여 전문가 시스템으로서의 기능을 갖추어 생산관리자의 업무를 대행할 수 있도록 발전되어야 할 것이다.

[참 고 문 헌]

[1] A.M.Mulvehill, "A user Interface for a Knowledge-Based Planning and Scheduling System," *IEEE Transactions on System, Man, and Cybernetics*, Vol.18,No.4, pp.514-521.(1988)

[2] A.S.Kiran & M.L.Smith,"Simulation Studies in Job Shop Scheduling-I·II,"*Computers & Industrial Engineering*, Vol.8, NO.2, pp.87-105.(1984)

[3] B.J.Cox, *Oject-Oriented Programming: An Evolutionary Approach*, Addison-Wesley, Massachusetts,1986.

[4] Christopher V.Jones & William L.Maxwell, "A System for Manufacturing Scheduling with Interactive Computer Graphics," *IIE Transactions*, pp.298-303.(Sep.1986)

[5] David Robson,"Object-Oriented Software Systems," *BYTE*,Vol.6,No.8,pp.74-86.(1981)

[6] Eero Eloranta, *Computer-Aided Production Management: Chap.12 User Interface*, Asbjorn Rolstadas,pp.181-199.(1988)

[7] Geoffrey A.Pascoe, "Elements of Object-Oriented Programming", *BYTE*, Aug.1986

[8] L.J.Walker & J.D.Woolven,"Development and use of a Visual Interactive Planning Board within Alcan Aluminium," *European Journal of Operational Research*, Vol.54, pp.299-305.(1991)

[9] M.Stefik & Daniel G.Bobrow, "Obect-Oriented Programming: Themes and Veriations," *The AI Magazine*, Vol.6,No.4,pp.40-62.(1986)

[10] Martin F.Clark, "Witness: Unlocking the power of Visual Interactive Simulation," *European Journal of Operational Research*, Vol.54,pp.293-298.(1991)

[11] Rosser T.Nelson, Charles A.Holloway and Ruby M.Wong, "Centralized Scheduling and Priority Implementation Heuristics for a Dynamic Job Shop Model," *AII Transactions*, Vol.9,No.1,pp.95-102.(Mar.1977)

[12] T.J.Ross, L.R.Wagner, and G.F.Luger, "Object-Oriented Programming for Scientific Codes.I:Thoughts and Concepts, II:Examples in C++,"*Journal of Computing in Civil Engineering*, Vol.6,No.4,pp.480-515.(Oct.1992)

[13] Won Kim & Fredeick H.Lochovsky, *Object-Oriented Concepts, Databases and Applications*, Addison-Wesley, New York, 1989

[14] 김승권, 김선욱, 이성윤, "효과적인 일정계획 활동을 위한 그래프에 의거한 전문가 시스템," 경영과학, 제 9 권 제 2 호 (1992)