# ObjecTime을 이용한 지능형 제조 운영 시스템의 설계 및 구현

김기정○ 홍성수
서울대학교 전기공학부 실시간 운영체계 실험실

# Design and Implementation of Intelligent Manufacturing Operating System Using ObjecTime

Ki-jung Kim○ Seongsoo Hong
Real-Time Operating Systems Laboratory
School of Electrical Engineering, Seoul National University, Seoul 151-742, Korea

## 요 약

본 연구에서는 객체지향적 CASE 툴인 ObjecTime을 이용하여 지능형 제조 운영 시스템을 설계/구현하였다. 객체지향적 프로그래밍 방법론과 ROOM(Real-time Object-Oriented Modeling) 방법론을 사용하여 지능형 제조 운영 시스템에서 필요한 객체를 설계하였으며, 이들 객체의 계층도와 관계를 정의하였다. 객체지향적 설계 방법론을 사용하여 설계한 시스템은 모델링 과정에서 실행화일을 얻는 과정까지 발생하는 오류를 최소화할 수 있다는 장점이 있다. 또한, 본 연구에서는 제조 시스템의 모니터링과 제조 시스템 운영 소프트웨어의 자료 입력 및 수정을 위한 그래픽 사용자 인터페이스 툴을 제작하였다.

## 1. 서론

제조 시스템이 복잡해지고 확장성이 강조되면서 운영 소프웨어의 중요성이 나날이 커지고 있다. 기존의 운영 소프트웨어는 정해진 레이아웃(layout)이나 컴포넌트(component)를 대상으로 작성된 경우가 대부분이기 때문에 확장성이나 재사용성이 떨어지는 것이 현실이다. 전체 시스템을 운영 제어하는 운영 소프트웨어는 유연생산시스템(FMS: Flexible Manufacturing System)의 가장 핵심적인 부분이며 시스템의 성능에 큰 영향을 주기 때문이다. 아울러 운용 소프트웨어는 계속 발전하는 운영 기술과 생산 설비의 변화에 따라 최소의 시간과 경비로 대응할 수 있어야 한다.

실제로 많은 운용 소프트웨어들이 연구 개발 되었지만 새로운 설비가 추가 될 때마다 방대한 운용 프로그램을 수정 보완하기 위해 많은 비용과 시간이 소모되는 단점이 있었다. 이러한 문제를 해결하기 위해 FMS 운용 소프트웨어 분야에서는 '확장성', '호환성', '재사용성' 그리고 '이식성'을 보장할 수 있는 객체 지향 기법이 도입되었다[2, 3].

본 연구는 확장성과 재사용성을 보장하는 제조 시스템 운용 소프트웨어의 개발을 목적으로 한다. 먼저 제조 시스템의 제어 구조를 설계하였고, 제조 시스템에 사용되는 객체를 정의, 설계하고 이를 시뮬레이션을 통해서 테스트하였다. 현재 제조시스템을 실제 목적 플랫폼(target platform)인 윈도우NT로 포팅하고 있으며, 제조 시스템을 모니터링하고 제조 시스템 운영 소프트웨어를 관리하기 위한 그래픽 사용자 인터페이스 툴을 제작하였다.

본 논문에서는 2장에서 객체지향설계방법론을 소개하고, 3장에서는 IMOS(Intelligent Manufacturing Operating System)의 객체 구조에 대해 설명한다. 4장에서는 제조시스템의 모니터링과 IMOS의 관리를 위한 툴의 구현에 관해 설명한다. 마지막으로 5장에서는 앞으로의 연구방향에 대해서 논의한다.

## 2. 연구배경

본 연구에서 목적으로 하는 IMOS는 모델링과 프로그래밍 단계에서 객체지향적 접근 방식과 ROOM을 도입하여 개발하고 있다. 제조 시스템 운영 소프트웨어 개발에 적용하기 위한 이들의 개념을 아래에서 간단하게 설명한다.

### 2.1 객체지향 기법

객체지향 접근방식은 초기에 프로그래밍 언어인 SIMULA(Dahl and Nygaard, 1966), Smalltalk(Goldberg and Robson, 1983)와 함께 사용하면서 시작되었고, 그 후 Adiga, Glassey, King, Fisher에 의해 제조분야에 적용되기 시작하였다.

객체지향 접근방식은 기존 순서지향 방식의 프로그래밍 기법에서 나타난 프로그램의 호환성, 확장성, 재사용성이 부족한 문제를 극복하여 보다 용이하게 복잡한 시스템을 모델링할 수 있다. 그리고 소프트웨어의 생산성 및 신뢰성을 향상시킬 수 있도록 모듈화의 개념을 프로그래밍 기법에 적용시키면서 발전하게 되었다. 즉 개발된 소프트웨어의 프로그램을 변경하여 이를 재사용할 때나, 시스템이 변경되었을 경우 이에 적절하게 대응할 수 있도록 하기 위하여 소프트웨어 모듈화의 중요성을 강조하게 되었다.

객체지향 접근방식에서는 현실의 모든 대상을 객체(object)로 설명한다. 같은 성질을 갖는 객체들에 대해서는 하나의 클래스를 정의한다. 클래스에는 데이터의 구조와 함께 그에 적용되는 연산들이 방법(method)으로 정의된다. 객체지향 개념은 객체와 클래스의 개념 외에 캡슐화, 다형성의 개념을 포함한다.

객체지향 기법은 실제 시스템 내에 있는 객체들과 추상화 된

객체사이의 대응관계를 형성하여 이로부터 클래스, 방법과 이들의 계층구조를 정하는 데 유용하게 사용되며, 특히 CIM(Concept Information Model) 환경 등의 복잡한 시스템을 모델링하는 데 적합한 것으로 알려져 있다.

### 2.2 ROOM 방법론

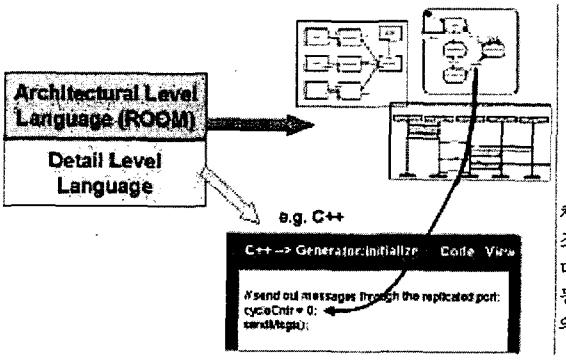


그림 1: ROOM 방법론

ROOM은 이벤트 기반 분산 실시간 시스템(event driven distributed real-time system)에 적합하게 개발된 방법론이다. ROOM의 기본 개념은 모델링 차원(*modeling dimensions*)과 추상화 레벨의 방법론(abstraction levels paradigm)으로 대표된다. ROOM을 이용한 전체적인 설계과정은 그림 1과 같다.

모델링 차원은 시스템의 각 요소에 대한 구조와 그에 상응하는 시스템의 동적 특성, 상속성으로 나뉜다. 추상화 레벨의 방법론은 시스템의 상위 개념을 기술하는 개요 단계(schematic level)와 실제 구현에 대한 내용을 나타내는 상세 단계(detail level)로 구성된다.

개요 단계에서의 중요한 모델링 단위로서는 행위자(actor)가 있다. 행위자는 특정한 기능을 수행하는 객체(object)이다. 행위자는 독립적으로 동시에 수행 가능하며 메시지를 통하여 통신한다. 통신을 하는 행위자간에는 구조(structure)가 존재한다. 이러한 모델링 단위는 객체 지향 개념에 따라 클래스로 정의된다. 클래스는 행위자 클래스(actor class), 데이터 클래스(data class), 프로토콜 클래스(protocol class)가 있다[1].

ObjecTime은 하향식 설계 방법론이며 실시간 시스템을 모델링하기에 적합한 ROOM(Real-time Object-Oriented Modeling)에 기초하고 있다. ObjecTime의 개발환경은 ROOM에 기초를 둔 모델링으로부터 실제 시스템에서의 실행화일의 생성에 이르는 모든 과정을 지원한다. 이 과정은 마우스의 드래그 앤 드롭, C++ 코드의 자동 생성으로 간결화 된다. 즉, 프로그래머는 자기가 생각한 모델에서 프로그램을 만드는 과정에서 발생할 수 있는 오류를 최소한으로 줄일 수 있다.

## 3. 제조시스템을 위한 객체 구조의 구현

### 3.1 IMOS의 객체구조

IMOS는 유연생산시스템을 위한 운용 소프트웨어이며 실제 자원을 사용하여 시스템을 운영할 수 있을 뿐만 아니라 가상의 자원을 연결하여 가상의 시스템을 운영하는 시뮬레이터로도 이용가능토록 설계하였다.

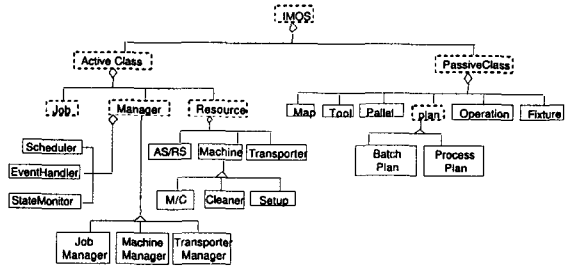


그림 2: IMOS 객체 구조

유연생산시스템은 생산품의 다양성뿐만 아니라 제조기계의 교체 등과 같은 환경의 변화에 대처해야 한다. 따라서 기본제어구조와 자원을 포함한 시스템 내의 모든 개체를 객체로 구현하였다. 시스템 내의 객체의 그림 2과 같이 실제 동작을 가지는 자원 등의 능동적 객체뿐만 아니라 정보의 형식을 지정하거나 메시지의 형식을 지정하는 수동적 객체들까지를 포함한다.

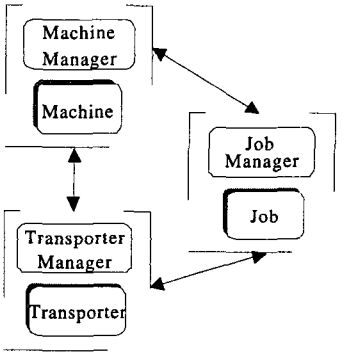


그림 3: 분산 제어기 구조

객체는 정적인 특성(static characteristics)과 동적인 특성(dynamic chracteristics)로 나뉘어 정의된다. ROOM은 객체의 정적인 특성은 도식적인 기호를 이용하여 구조를 정의함으로서 나타내고, 동적인 특성은 상태도표(Statechart)를 정의함으로서 나타낸다.

IMOS의 기본 제어기는 그림 3와 같은 분산형 제어 구조로 설계하였다. 이는 중앙 집중식 제어기에 걸리는 과도한 부하를 피하고 재구성이 용이하도록 하기 위해서이다. 또 각각의 제어기는 그 내부에 외부와의 메시지 응답을 결정하는 **EventHandler**, 해당 제어기가 관리하는 자원의 상태를 모니터하는 **StateMonitor**, 스케줄링을 담당하는 **Scheduler**를 가지고 있으며 실제로 구현한 시스템은 제어기의 하부에 해당하는 자원들이 각각 연결되도록 하였다.

구성된 각 객체는 내부와 외부에서 발생되는 이벤트에 의해 구동되며 이러한 이벤트는 시스템 내에서 메시지로 구현된다. IMOS에서는 주요한 메시지 집합을 그림 4와 같이 정의하였고 이에 대한 각각의 객체의 동작은 해당 객체의 동적인 특성으로 구현하였다.

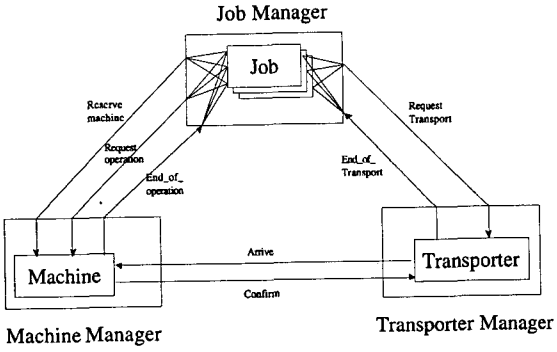


그림 4: Message Set

## 4. IMOS 모니터링 시스템의 구현

### 4.1 모니터링 시스템의 개요

CASE 툴인 ObjecTime을 이용하여 설계한 IMOS는 ObjecTime이 제공하는 프로토타이핑 언어인 RPL을 사용하여 먼저 프로토타입을 구현하였다. 이 단계에서는 시뮬레이션을 지원하는 툴의 도움을 받아서 각 모듈의 동작 상태나 메시지의 흐름을 추적할 수 있지만 실제 시스템을 운영하기 위하여 목적 플랫폼으로 포팅이 되고 나면 위와 같은 정보를 모니터링할 수 있는 방법이 존재하지 않는다. 그러나 실제 FMS에서는 각 모듈들의 상태를 모니터링하거나 공장의 동작을 시작하거나 멈출 수 있는 관리 툴이 필요한 것이 현실이다. 따라서 윈도우 머신상에서 제조 시스템을 모니터링하고 운영소프트웨어를 관리할 수 있는 프로그램을 Visual C++로 구현하였다.

모니터링 시스템은 머시닝 센터(Maching Center)와 트랜스포터(Transporter), 자동창고(AsRs)의 상태 및 작업여부를 수치적으로 확인할 수 있게 하였을 뿐만 아니라 OpenGL로 구현한 3D 모델을 통해 실제 공장의 동작 모습을 실시간으로 애니메이션을 통해 보여줄 수 있도록 하여, FMS의 동작상태를 쉽게 파악할 수 있게 하였다. 그리고 생산계획관리, 동작(Operation) 관리, 팔레트(Pallet), 치구의 지정 및 변경 작업을 모니터링 시스템에서 처리하게 하였으며, FMS의 초기화 및 세팅에 필요한 작업도 모니터링 시스템 상에서 이루어진다. FMS에서 발생하는 모든 메시지와 이벤트는 화면출력과 디스크의 저장을 통해 로깅함으로서 공장의 정보를 기록하거나 또는 돌발 상황에서의 복구를 보다 용이하도록 하였다.

### 4.2 ObecTime과의 통신

IMOS와의 통신은 TCP/IP에 기반한 소켓을 통하여 이루어진다. ObjecTime을 사용하여 설계한 머신 매니저(machine manager), 잡 매니저(job manager), 트랜스포터 매니저(transporter manager)는 ObjecTime이 지원하는 통신모듈인 GTF(Generic Transport Framework)을 이용하여 메시지를 주고 받을 수 있다.

모니터링 시스템은 ObjecTime이 사용하는 프로토콜을 인코딩하거나 디코딩하여 통신할 수 있도록 통신 라이브러리를 구현하였다. 통신 라이브러리는 윈도우 소켓을 사용한 비동기 통신 서버(asynchronous communication server)로 구현되었으므로, 3개의 독립적인 매니저들과 블록킹(blocking)없이 6개의 프로토콜을 사용하여 메시지를 주고 받을 수 있다.

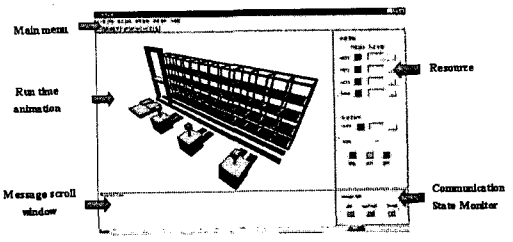


그림 5: Monitoring GUI

제조시스템의 모니터링을 위한 툴의 전체적인 구조는 그림 5와 같다.

## 5. 결론

본 연구에서는 확장성과 재사용성을 보장하는 객체지향적 소프트웨어 설계툴인 ObjecTime을 이용하여 객체를 정의하고 이를 통해 IMOS를 구현하였다. IMOS는 ObjecTime이 지원하는 프로토타이핑 언어인 RPL로 구현되어 시뮬레이션을 통해 설계의 정확성이 확인되었으며, 현재 C++을 사용하여 윈도우 NT 플랫폼으로의 포팅이 진행 중이다. 뿐만 아니라 제조시스템 운영 소프트웨어를 모니터링하고 시스템을 시작하고 새로운 자료를 입력 또는 기존의 자료를 수정할 수 있는 운영 소프트웨어 관리 툴을 개발하였다.

현재 구현되어 있는 기본적인 동작뿐만 아니라 오류제어모듈을 구현하고 타시스템으로의 이식성을 높이기 위하여 범용 제조시스템을 위한 객체의 설계를 추가하는 것이 앞으로의 연구과제이다.

## 참고 문헌

[1] Selic B., G. Gullekson, and P. T. Ward. *Real-Time Object-Oriented Modeling.* John Wiley and Sons, 1994.

[2] Adiga S. Software modeling of manufacturing systems: A case for an object-oriented programming approach. *Annuals of Operation Research*, pages 363–378, 1989.

[3] Smith J. S. and Sanjay B. Joshi. Reusable software concepts applied to the development of fms control software. *International Journal of Computer Integrated Manufacturing Vol.5 No.3*, pages 182–196, 1992.