The Journal of Korean
Institute of CALS/EC
Vol. 4, No. 1, June 1999

객체지향 방법론을 이용한 제품정보관리(PDM) 시스템에서의 워크플로우 설계

최종윤*, 최경희**, 안병하**

Workflow Design on Product Data Management System Using Object-Oriented Modeling Technique

Jongyoon Choi, Kyunghee Choi, Byung Ha Ahn

Abstracts

The challenge is to maximize the time-to-market benefits of concurrent engineering while maintaining control of data and distributing it automatically to the people who need it when they need it. The way PDM systems cope with this challenge is that master data is held only once in a secure vault where its integrity can be assured and all changes to it monitored, controlled and recorded. The structure of PDM is various from vendor, but they have common module. That is PDM and it is most important. The goal is to design the workflow in PDM using object-oriented modeling method. The past methods have concentrated on the flow between workflow engine and agent, but this paper will focus on task. We will model task as individual object. This paper uses OMT(Object Modeling Technique) by James Rumbaugh for base modeling tool and uses DCOM(Distributed Component Object Model) for base ORB(Object Request Broker). Research object is to design the static object model, to design state change by dynamic model and to design data transition by functional model.

Key Word : PDM, 워크플로우, OMT, DCOM

* LG-EDS 시스템
** 광주과학기술원 기전공학과

1. 서론

최근 PDM 시장은 매년 30% 이상씩 성장하고 있으며, 여러 상용 제품들이 개발되어 판매되고 있다. 대표적인 제품으로는 ComputerVision 의 Optegra, EDS 의 IMAN, HP 의 WorkManager, IBM 의 ProductManager 등이 있다. 이들 제품들은 구조에 있어서 서로 상이한 특징을 갖고 있으나 이들을 살펴보면 공통적으로 데이터저장 모듈, 워크플로우 모듈 등을 포함하고 있음을 알 수 있다[강석호,1996].

본 논문에서는 객체지향방법론을 이용하여 PDM 에서의 워크플로우를 설계하고자 한다. 기존의 방법론들은 워크플로우 엔진과 작업자(Agent)사이의 흐름에 중점을 두었으나 본 논문에서는 프로세스와 작업(Task)에 중점을 두고 이를 각각의 객체로 설계하는 방안을 연구하고자 한다. 시스템의 설계를 위하여 Rumbaugh 의 OMT(Object Modeling Technique) 를 기본 모델링 방법론으로 사용한다.

이 논문의 구성은 다음과 같다. 2 장에서는 워크플로우에 관련된 기본적인 내용과 관련 연구들을 살펴보기로 한다. 3 장에서는 본 논문에서 사용된 OMT 에 대하여 간략히 알아본 후, 시스템구조를 모델링 하기로 한다. 4 장에서는 DCOM 을 사용한 실제 시스템의 구축에 관하여 설명한다. 5 장에서 추후의 연구방향을 알아보기로 한다.

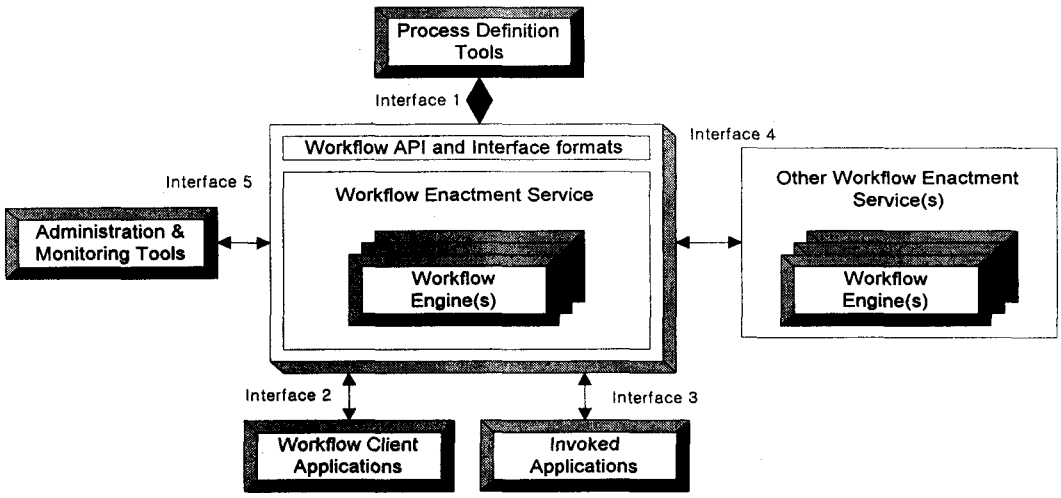
2. 워크플로우 관련 연구

2.1 워크플로우의 기본 이슈

워크플로우는 70 년대 후반에 사무자동화와 일괄처리(Batch Processing)에서 유래되었으나 최근에 들어서야 네트워크 컴퓨팅, 클라이언트/서버, 객체지향 기술과 같은 관련 정보기술 들의 발전으로 실질적 구현이 가능하게 되었다. 워크플로우는 그룹웨어와 같은 여러 분야에서 사용되고 있는데, PDM 에서 사용되는 워크플로우는 작업이 항상 작업자에게 할당되고 처리시간이 작업자에 의해 좌우되는 사건중심(Event-Driven) 시스템이다.

워크플로우 시스템의 구성요소는 다음과 같다[Ngu,1996].

- Workflow – 작업들 일부분 또는 전체적 순서의 집합
- Task – 인간의 행동 또는 다른 작업을 위한 처리(Operation)와 기술(Description)의 일부분 또는 전체적 순서
- Manipulated Object – 문서, 데이터 레코드, 이미지, 전화, 팩스 등과 같은 작업의 대상 또는 결과물
- Role – 특정 작업을 수행하는데 요구되는 인간의 능력 또는 정보시스템의 서비스
- 작업자 – 역할들을 이행하고, 작업들을 수행하며, 워크플로우 실행 중 상호 작용하는 인간 또는 정보시스템



<그림 1> WFMS Reference Model[WfMC,1997]

1993 년에 워크플로우와 관련된 표준화 작업을 위하여 WfMC (Workflow Management Coalition)이 결성되었다. WfMC 에서는 서로 다른 제품들간의 상호교류를 위한 Interface 의 개발을 위해 워크플로우 Reference Model 을 제안하였다[WfMC,1997].

<그림 1>에서와 같이 Reference Model 은 모두 5 개의 Interface 를 갖고 있으며 Data 의 교환을 위해 WAPI(Workflow Application Programming Interface)을 포함하고 있다. 본 논문에서는 인터페이스 2 를 대상으로 하고 있으며 인터페이스 3 의 경우는 DCOM 으로 구현된다.

워크플로우의 효율적 관리를 위해 필요한 것은 다음과 같다[Sheth,1995].

- 구성요소 지향적
- 다양한 워크플로우 Application 을 지향
- Application 의 정확성과 신뢰성을 확신시킬 수 있어야 한다.
- 워크플로우 Application 의 진화, 대체, 추가 등을 지원하여야 한다.

이를 위해서는 워크플로우의 모듈화 및 표준화가 필수적이다.

2.2 워크플로우 관련 연구들

최근에 많은 워크플로우에 관한 연구들이 서로 다른 워크플로우 간의 접속을 위한 시스템의 설계 및 표준화를 대상으로 하고 있다[WfMC,1997]. WfMC 가 결성 이후 이에 관한 연구를 주도적으로 추진하고 있다[Ngu,1996]. 그 외에도 시스템의 Active Rule 설계에 관한 연구가 많은 부분을 차지하고 있다[Casati,1996a] [Zimmermann,1996].

2.3 기존 연구의 한계 및 새로운 연구의 필요성

기존의 구조는 한 프로세스에 한 개의 엔진을 대응시켜서 한 Enactment Service 내에 여러 개의 엔진이 존재하는 구조를 띄고 있다. 이러한 경우, 동시공학 환경에서 흔히 발생할 수 있는 여러 프로세스가 상호 작용하는 워크플로우의 경우, 각각의 엔진간의 영역문제 및 Data 호환문제로 인한 여러가지 문제가 발생할 수 있다[Graw,1996]. 이는 기존의 엔진들이 구조적(Structured) 모형에 기반을 두고 워크플로우 엔진과 작업자사이의 프로세스를 시작과 끝을 갖는 흐름으로 설계하였기 때문이다.

이에 본 논문에서는 프로세스와 작업에 중점을 두고 이를 각각의 객체로 설계하는 사건중심(Event-Driven)의 엔진제작방안을 연구하고자 한다. 프로세스는 작업의 집합객체(Aggregate Object)이며 Event 발생시 Rule의 확인 후 이를 실행하는 구조를 1 개의 엔진으로 해결한다. 엔진이 1 개만 존재하여서 발생하는 과부하를 막기 위하여 엔진의 각 기능을 모듈화 한 후 이를 각기 다른 Computer 에 분산시킨 후 분산객체를 통하여 상호접속 하도록 한다.

3. 연구방법

3.1 OMT

3.1.1 Three Model Approach by Rumbaugh

OMT 는 시스템의 표현을 위해 3 가지 모델을 사용한다[Runbaugh,1991].

- *Object Model* – 시스템의 정적 구조를 표현한다. 시스템을 구성하는 객체, 객체간의 연관관계, 객체의 속성, 객체의 Operation 을 표현한다. 이는 Object Diagram 으로 표현되며, 연관관계를 나타내는 Arc 와 객체 클래스를 나타내는 Node 로 표현된다.
- *Dynamic Model* – 시스템의 시간에 따른 변화구조를 표현한다. 이는 시스템의 제어부분을 규정하여 적용키 위한 것이다. 이는 State Diagram 으로 표현되며, Arc 는 Event 에 의한 State 간 변이, Node 는 State 를 의미한다.
- *Functional Model* – 시스템 내에서 자료값의 변화를 표현한다. 이는 Data Flow Diagram 으로 표현된다.

위의 세가지 모델은 완벽한 시스템의 설계를 위해 상호 참조된다.

3.1.2 왜 OMT 를 쓰는가?

객체지향 방법론은 과거의 구조적 방법론에서 크게 변화하였다. 구조적 방법론은 시스템의 기능들을 분석하여 이를 설계하는 것을 강조하나, 객체지향 방법론은 문제 속의 자료를 중심으로 객체를 찾아낸 후, 이에 프로시저를 추가하여 시스템을 구성하는 것을 목표로 한다. 객체기반의 소프트웨어들은 문제자체에 중심을 두었으므로 쉽게 변화 시킬 수 있다. 여러 객체지향 방법론 중에서 OMT 는 적용, 분석, 설계에서 강점을 보인다[김성회,1997].

3.2 Problem Statement

OMT 의 첫번째 단계는 Problem Statement 이다. 무엇을 하는지에 대한 문제정의를 내리는 것이다. 이 단계에서는 문제의 범위, 요구사항, 가설, 요구성능 등을 구술한다.

시스템의 동작방식 등의 방법론은 이에 포함되지 않는다.

3.2.1 문제의 범위

워크플로우는 특정 목적을 위해 문서나 정보가 미리 정해진 규칙에 의거하여 참여 인력사이에서 움직이는 시스템의 자동화를 위한 프로시져이다[WfMC,1997]. 따라서 워크플로우 엔진은 여러 작업자가 하나의 목표를 위해 일관된 규칙에 의거한 흐름에 따라 작업을 하는 경우를 대상으로 한다.

3.2.2 기본적 요구기능

PDM 에서의 워크플로우 엔진은 다음과 같은 기본 기능을 지녀야 한다.

- 프로세스 정의 – 프로세스가 어떠한 구조로 이루어졌는지 구술한다.
- 워크플로우 조정(Coordination) – 프로세스를 구성하는 작업을 어떻게 수행할지 결정한다.
- 워크플로우 실행 – 결정된 작업을 실행한다.
- 워크플로우 관찰(Monitoring)– 현재의 State 를 파악하고 있어야 한다.
- 오류 처리 – 오류의 경우에 대처 가능해야 한다.
- 사용자 관리 – 작업자의 추가/삭제, Log-In/Out 의 관리가 되어야 한다.

3.3 Object Modeling

Object Model 은 Dynamic Model 과 Functional Mmodel 보다 먼저 구축된다. 이는 시스템의 정적인 구조를 설명한다.

3.3.1 객체 클래스의 선정

Object Model 을 설정하는 첫 단계는 Problem Statement 로부터 객체클래스를 선택하는 것이다. Problem Statement 에 모든 객체 클래스가 명백히 나타난 것은 아니다. 밖으로 보이지 않는 내부적 객체의 추가도 가능하다. 그 후 중복된 객체를 제거한 후 객체간의 연관관계를 설정한다.

WFMS 에서의 객체는 엔진에 관련된 제어 객체와 프로세스에 관련된 작업 객체로 구분할 수 있다.

a) 작업 객체

프로세스는 다수의 작업 객체가 모여서 이루어진 집합 객체로 표현된다. 단위작업은 책임주체가 1 명의 작업자인 기본단위 구성 객체이다. Fork 작업, Join 작업, Conditional 작업은 실제 작업이 아닌 프로세스분기상의 가상 작업으로 엔진이 실행 한다.

- Fork 작업 – 하나의 작업종료 후 다수의 작업이 동시에 실행
- Join 작업 – 다수의 작업종료 후 하나의 작업이 실행
- Conditional 작업 – 사전조건의 만족여부에 따라서 작업들이 실행

b) Control Object

본 논문에서는 워크플로우 엔진을 6 개의 객체로 구성하였다.

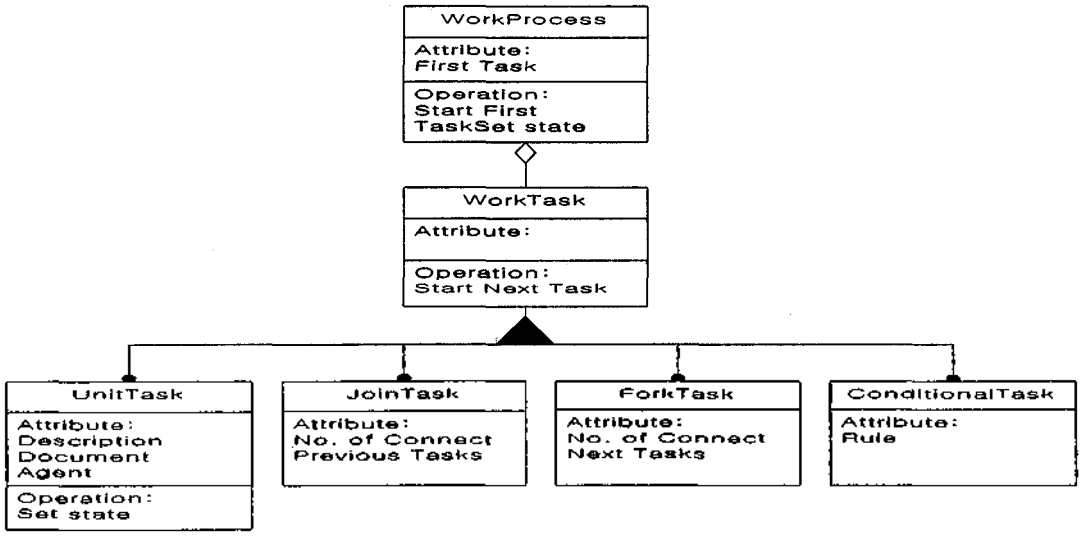
- Scheduler – 프로세스의 정의를 읽어 들인 후 단위작업에 작업자를 할당한다.
- Dispatcher – Scheduler 에서 할당된 단위작업이나 Fork 작업, Join 작업, Conditional 작업을 Worklist Handler 로

전달(실행)

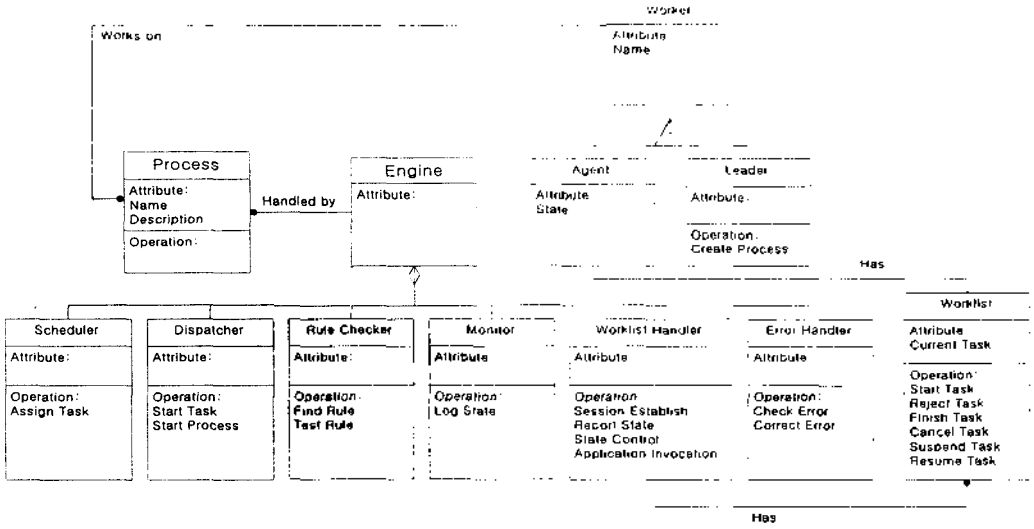
- Rule Checker – 프로세스상의 작업의 수행에 관련된 Rule 을 처리
- Error Handler – 오류발생시 이를 알리고 교정활동을 실행
- Worklist Handler – 각각의 작업자에게 할당된 작업의 상태 및 진행을 관리
- Monitor – 워크플로우 상의 프로세스, 작업, 작업자의 상태를 모니터링 한다.

위의 6 모듈들은 State Database 를 참조 한다.

3.3.2 Object Diagram



<그림 2> 작업과 프로세스의 Object Diagram



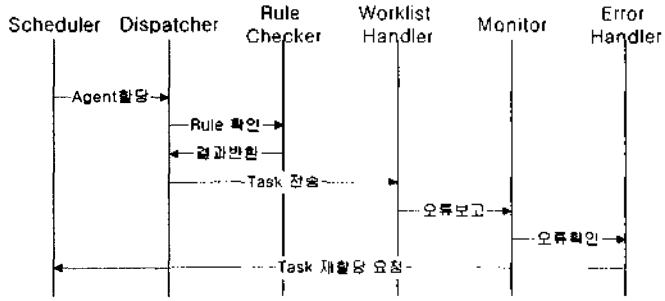
<그림 3> 엔진의 Object Diagram

3.4 Dynamic Modeling

각각의 객체들은 시간이나 환경에 따라 변하는 Dynamic Model을 갖는다. 즉, 외부 자극(Event)에 따라 객체의 값(State)이 변화하게 된다.

3.4.1 시나리오

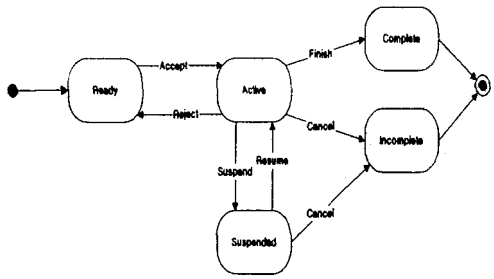
시스템의 실행 중 발생되는 Event를 열거한다. 이 경우 중요한 것은 Event의 발생자와 수여자의 구분이다.



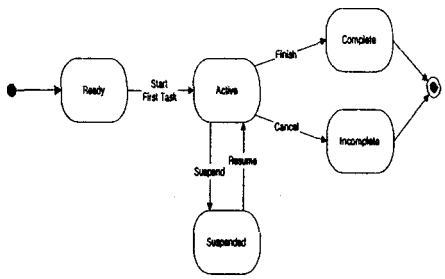
<그림 4> 엔진의 시나리오

3.4.2 State Diagram

State는 Node로 표현되며, 화살표는 전이(Transition)로 표현된다. Event는 화살표 위에 표기된다. 작업과 프로세스의 변화는 Event를 발생시키므로 가장 중요한 모형이다.

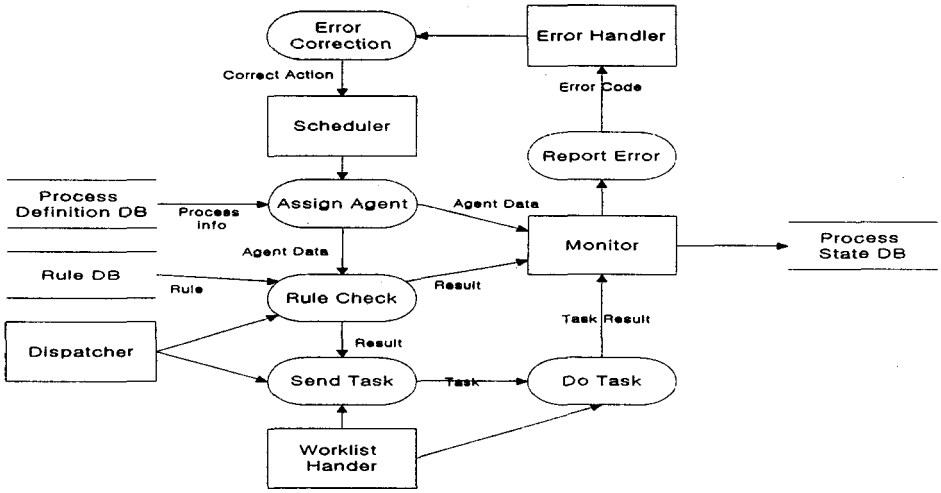


<그림 5> Work 작업의 State diagram



<그림 6> Work 프로세스의 State diagram

3.5 Functional Modeling



<그림 7> Functional Modeling

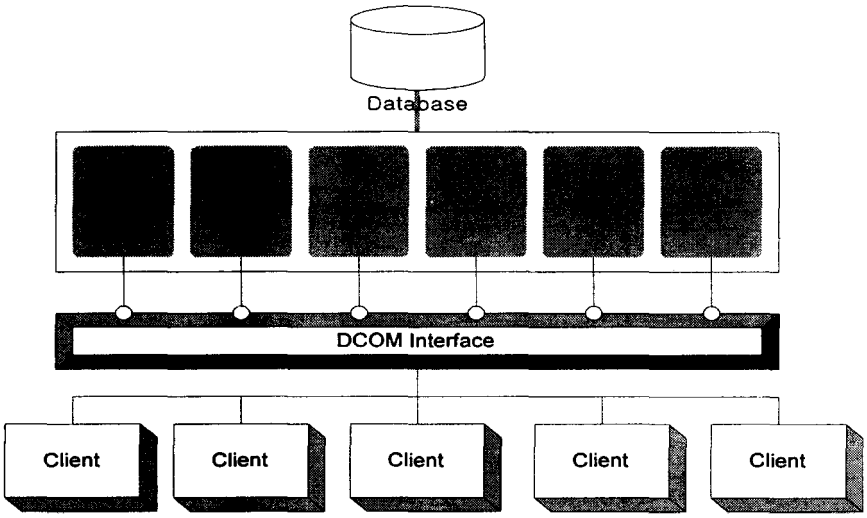
<그림 7>은 입출력 사이의 변환관계를 Data Flow Diagram 으로 표현한 것이다. Scheduler 에서 단위 작업 할당이 끝나면 Scheduler 는 이를 Monitor 에 알리고 Monitor 는 Work Process 의 State 를 Active 로 변환한다. Dispatcher 에서 Task 의 실행이 시작되면 Dispatcher 는 Task 의 상태를 Ready 로 변환한다. Worklist Handler 는 Dispatcher 에서 전달된 Task 를 Agent 의 Task List 에 저장하고 이를 보고하면 Task 의 상태를 Active 로 변환한다. Task 가 작업수행 도중 Suspend 에 들어가거나 Cancel, Finish 의 결과를 내고 종료되면 이를 보고하고 Reject 시 Ready 를 보고한다. 이러한 모든 보고에 대하여 Monitor 는 Task State Database 와 Process State Database 의 Object State 속성을 변환함으로써 대응한다.

4. 적용

4.1 시스템의 구조

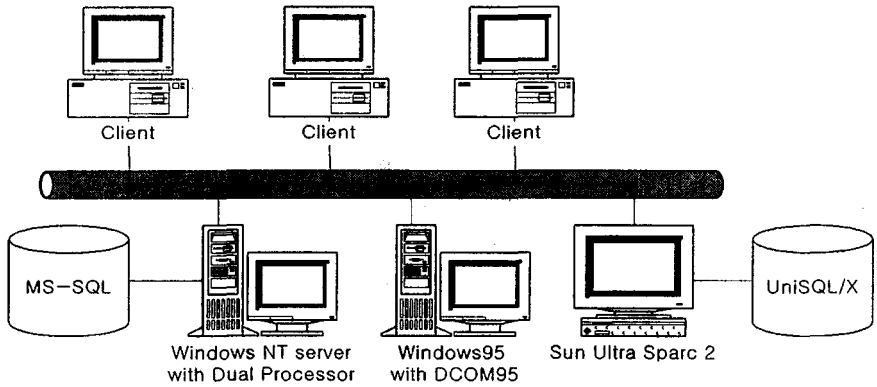
이 논문에서는 3 Layered Client/Server 시스템구조가 제안되었다. 다음의 그림은 시스템의 구조를 보여준다.

- Database – 작업자의 상태, 프로세스구조, 프로세스와 작업 상태가 저장된다.
- Client – 작업자에게 Work-list 의 User Interface 를 제공한다.
- Sever – 6 개의 모듈로 구성되며 각각의 모듈은 한 컴퓨터의 다른 프로세스나 Network 상의 분산된 컴퓨터상에 존재한다.
- DCOM Interface – Server 와 Client 사이의 연결을 담당한다.



<그림 8> 시스템구조

4.2 실제 Model



<그림 9> 구조도

본 논문에서의 모델은 위와 같은 시스템 구성에서 적용되었다. 현재의 전산환경은 Client/Server 구조의 분산환경으로 옮겨가고 있다. Client/Server 구조는 Unix Machine 에서 구현되었으나, 최근에는 개인용 컴퓨터의 발전으로 PC 기반의 시스템을 많이 사용하고 있다. 본 논문에서는 DCOM 의 사용을 위해 Windows NT 를 사용하였다.

DCOM 의 경우 Windows NT 환경에서 Server 와 Client 모두 작동가능하고 Windows 95 에서는 Client 로만 가능하다. DCOM95 를 설치하면 Server 로도 사용가능하나 신뢰성이 보장되지 못한다[Grimes,1997]. Windows 98 에서는 기본적인 지원이 가능하다.

Network Protocol 의 경우 소규모 로컬 네트워크에서는 IPX/SPX 나 NetBeui 등이 사용되었으나 인터넷과 인트라넷의 확산으로 TCP/IP 도 점차 널리 쓰이는 추세에 있다. DCOM 의 경우 위의 3 Protocol 을 모두 지원하여 지역적인 거리에 관계없이 사용 가능하다. 본 논문에서는 ORDBMS 인 UniSQL/X 가 TCP/IP 를 통하여 Unix 시스템에서 호출되는 관계로 TCP/IP 를 사용하였다. 추후 확장성도 고려해야 한다. DCOM 모델의 장점은 프로그램을 모듈화 한 후 모듈간의 인터페이스를 설정하면 이를 준수하는 한 다른 모듈에 영향을 주지않고 새로이 추가된 성능을 넣을 수 있다는 점이다. 또한 인터페이스를 준수하는 한 모듈별로 다른 프로그래밍 언어를 사용할 수 있다. 본 논문에서는 Visual Basic 이 사용되었다. 모델내의 Some Object 들은 실질 DB Object 이다. UniSQL 을 통하여 이들을 저장하며 Visual Basic 으로 이들을 제어 한다.

5. 결 론

5.1 Summary of Contribution

이 논문은 워크플로우의 설계 및 적용에 있어서 OMT 와 DCOM 을 이용하여 객체지향개념을 도입하고자 하였다. 이는 Unix 기반의 Client/Server 에서 PC 기반의 분산시스템 환경으로의 이전을 위해 필수적인 사항이다.

5.2 Required for Further Research

첫째, Concurrent Engineering 환경에서는 하나의 작업이 완전히 끝나기 전에 다른 작업을 실행시킬 수 있다. 따라서 작업자 간의 통신의 개념을 도입할 필요가 있다.

둘째, DCOM 과 Visual Basic 은 분산 시스템의 구축에 있어서 많은 제약을 갖고 있다. JAVA 와 CORBA 로의 Porting 이 필요하다.

셋째, Structure 에 중점을 둔 결과로 인해 WAPI 의 도입을 차후로 미루었다. 다른 워크플로우와의 접속을 위해서는 WAPI 의 도입이 필요하다.

참고문헌

[WfMC,1997]a 워크플로우 Management Coaliation, Workflow Handbook 1997, John Willey & Sons, 1997

[강석호,1996]a 강석호, 김영호, 황영헌, 김대환, "PDM 시스템의 평가에 관한 연구", 한국 CALS/EC 학회지 제 1 권 제 1 호, 1996. 8

[Rogerson,1997] Dale Rogerson, Inside COM, Microsoft Press, 1997

[Grimes,1997] Richard Grimes, Professional DCOM, WROX Press, 1997

[Casati,1996a] F. Casati, S. Ceri, B.Pernici, G. Pozzi, "Deriving Active Rules for Workflow Enactment", Proceedigs 7th international Conference, DEXA '96, 09/09/96

[Casati,1996b] F. Casati, S. Ceri, B.Pernici, G. Pozzi, "Conceptual Modeling of Workflows", Politecnino di Milano, 1996

[Zimmermann,1996] J. Zimmermann, H. Branding, A. P .Buchmann, A. Deutsch, A. Geppert, "Design, Implementation and Management of Rules in an Active Database System", Proceedigs 7th international Conference, DEXA '96 , 9/09/96

[Runbaugh,1991]a James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, William Lorensen, Object-Oriented Modeling and design , Prentice Hall, 1991

[Ngu,1996] Anne H.H. Ngu, Toncan Duong, Uma Srinivasan, "Modeling Workflow Using Tasks and Transactions", Proceedings Seventh International Workshop on Database and Expert Systems Applications, 9/09/96

[Kroha,1993] Petr Kroha, Objects and Databases, McGraw-Hill International, 1993

[Sheth,1995]a A. Sheth, "An Over View of Workflow Management: From Process Modeling to Workflow Automation Infrastructure", Distributed and Parallel Databases, 119-153, 1995

[Graw,1996] G. Graw, V. Gruhn, H. Krumm, "Support of Cooperating and Distributed Business Process", Proceedings 1996 International Conference on Parallel and Distributed Systems, 06/96

[Carter,1992] Donald E.Carter, Barbara Stilwell Baker, Concurrent Engineering: The Product Development Environment for the 1990s, Addison Wesley, September 1992

[김성회,1997]a 김성회, 조성식, 김선욱, 박홍국, "OMT 를 이용한 그룹의사결정지원 시스템의 설계 및 구현", 産業工學:제 10 권 제 1 호 1997 3

저자소개

최종윤

성균관대학교 산업공학과 (학사)

광주과학기술원 기전공학과 (석사)

현재 LG-EDS 시스템 기술연구부문 분산컴퓨팅센터 객체기술사업팀 연구원

최경희

한국과학기술원 산업공학과 (학사)

광주과학기술원 환경공학과 (석사)

현재 광주과학기술원 기전공학과 연구원

안병하 (E-mail : bayhay@pia.kjist.ac.kr)

한국 공군사관학교 (학사)

한국과학기술원 산업공학과 (석사, 박사)

현재 광주과학기술원 기전공학과 교수