객체 지향 설계 모듈의 결합 방법

허 계 범[†] · 이 종 섭[††] · 정 계 동[††] · 최 영 근[†††]

요 약

대부분의 객체 지향 분석 및 설계 방법은 구조적 분석과 정보 모델링에 기반을 두고, 객체 지향 프로그래밍 언어에 근거한 직관적인 분석 및 설계 모델을 사용하고 있다. 그러므로 모델간의 의미 정확성과 일관성이 유지되지 못하여 시스템 구현시 많은 문제점을 가지고 있다.

본 논문에서는 새로운 시스템 개발 생명 주기방법에 따른 객체, 동적, 기능적 모듈을 위한 분해와 설계 방법을 제안한다. 따라서, 시스템 개발 전체 생명 주기를 새롭게 제시하고, 객체 지향 설계 절차와 명세화에 따른 객체, 동적, 기능 모듈의 분해를 위한 모듈 분해 기준과 이를 이용한 객체 지향 설계 방법을 제시한다. 제안된 방법은 개발자가 사용자 소프트웨어 요구사항들을 편리하게 반영할 수 있도록 해준다.

끝으로 본 논문에서 실 시스템 적용 사례를 들어, 객체 지향 설계 모듈의 결합 방법의 타당성과 실용성을 입증하고자 한다.

A Method for Combining Object-Oriented Design Modules

Kwae Bum Heo[†] · Jong Sup Lee[††] · Kwae Dong Jeong[††] · Young Eun Choi[†††]

ABSTRACT

Most object-oriented analysis and design methodologies are based on structured analysis and information modeling and are using for intuitive analysis and design models based on object-oriented programming languages. Therefore there are many problems such as when a system is implemented incorrect semantics and inconsistency between models.

This paper submits a decomposition and design method for object, dynamic and functional module of the methodology of a new system development life-cycle. Thus, we present a new system development life cycle, and suggests a object-oriented design method and standards of module decomposition for the decomposition of object, dynamic, functional models due to object-oriented design procedures and specifications. This proposed method enables developers to reflect user's software requirements conveniently.

We prove the validity and practicality of this object-oriented design method through implementing a real-system.

1. 서 론

객체 지향 소프트웨어 개발 방법들은 객체 지향 프로그래밍 언어 기술에서 발전하여 자연스러운 모델링(nature modeling)을 통해 도메인 분석 및 설계과정에서도 효과적으로 사용되는 소프트웨어 개발 방법론으로서 구현이 편리하고 재사용성이 뛰어나며, 유지보수가 쉬운 장점이 있다[1, 2, 3, 9, 11, 12, 14, 19].

그러나 많은 객체 지향 분석 및 설계 방법은 대부

† 정 회 원:광운대학교 전자계산학과 박사과정
†† 준 회 원:광운대학교 전자계산학과 박사과정
††† 정 회 원:광운대학교 전자계산학과 교수
논문접수:1995년 9월 23일, 심사완료:1996년 1월 12일

분 구조적 분석과 정보 모델에 기반을 두고, 객체 지향 프로그래밍 언어의 구문에 따른 분석 및 설계 모델을 사용하기 때문에 개발 단계별 절차(procedure)와 명세화(specification)등이 일치되지 않고 있다[21, 22].

특히, 이러한 현상은 실 세계의 문제 영역을 해결 영역으로 사상(mapping)하는 설계 단계에서 두드러지고 있는데, 이에 따라 객체 지향 소프트웨어 개발 방법을 적용한 실 시스템 개발에 있어 설계 단계를 소홀히 하는 경우가 많다. 그러나 이 같은 경우 시스템 개발 후에는 객체 지향 시스템의 장점들이 약화되거나, 오히려 유지 보수에 문제점으로 대두되기도 한다[22].

따라서 본 논문은 소프트웨어의 요구 사항을 명확하게 하기 위하여 다음 사항들의 만족을 목표로 한다.

첫째, 사용자의 요구를 정확하게 반영하고, 시스템의 기능적 요구 및 비기능적 요구를 체계적으로 분석할 수 있도록 한다.

둘째, 완전하고 일관성있는 명세서 작성과 구현된 프로그램의 확인 및 검증, 그리고 유지보수의 원활한 지원이 이루어질 수 있도록 객체 지향 설계 절차와 명세화를 개선한다.

셋째, 소프트웨어 개발 편의성과 객체 지향 설계 방법의 실용성을 얻을 수 있도록 한다.

넷째, 현재 관심 분야로 대두되고 있는 객체 지향 분산 컴퓨팅(object-oriented distributed computing)을 감안한 설계 방법들을 부분적으로 제시한다.

이를 위해 소프트웨어 개발 단계를 분석, 설계, 구현 단계로 구분하고, 단계별 객체 모델, 동적 모델, 기능 모델 기술을 적용, 비교적 세부적으로 기술한 J. Rumbaugh[14]의 객체 모델링 기술(OMT)을 바탕으로 한다.

그러나 본 논문의 설계 모듈 방법은 정형화된 객체 지향 모듈 분해 방법[21, 22]을 적용하여 객체, 동적, 기능적인 속성들을 독립적으로 가시화 할 수 있는 부품 합성 원리를 응용한 결합 방법으로, OMT 방법이 객체 모델 다이어그램상에 동적, 기능 모델을 표기의 차이만으로 가시화함으로서 발생하는 복잡도 문제를 해소하고, 구현 단계에서 분석 단계의 정보없이 설계 단계의 정보만으로 구현이 가능하도록 객체, 동적, 기능 모델 모듈 결합 다이어그램과 프로그램 처리 설명서를 제시한다.

본 논문의 구성은 2장의 관련 연구와 3장의 개선된 설계 방법 및 절차를 제시하고 4장에서는 실 시스템의 예를 들어 제시하고, 5장에서는 설계 검증과 비교 분석을 하고 6장에서는 결론과 앞으로 연구 방향에 관하여 논한다.

2. 관련 연구

본 논문에서는 현재 많이 상용화되어 있으며, 대표적인 방법이라 할 수 있는 J.Rumbaugh, Coad와 Yourdon, Martin과 Odell, Booch의 설계 방법을 중심으로 알아보고자 한다.[2, 9, 11, 12, 14, 16, 19]

2.1 Rumbaugh의 객체 모델링 기술(OMT)

Rumbaugh[14]의 OMT는 모든 소프트웨어 구성 요소들을 OMT 그래픽 표기법을 이용하여 모델링하는 방법으로, 객체 모델링, 동적 모델링, 기능 모델링 방법을 분석, 설계, 구현등의 소프트웨어 개발 전 단계에 적용한다. OMT의 설계는 시스템 설계, 객체(상세) 설계로 크게 나누고, 객체 설계 단계에서는 시스템의 다면성을 동시에 표현하기 위해 객체, 동적, 기능 모델을 객체 모델 다이어그램상에 표기의 차이를 두어 표현한다.

2.2 Coad와 Yourdon의 객체 지향 설계

Coad과 Yourdon[16]의 객체 지향 설계 방법은 문제 영역 요소, 사람과 상호 작용 요소, 타스크 관리 요소, 자료 관리 요소등 4 가지의 요소를 구성하여 구현한다. 문제 영역 요소는 객체 지향 분석 모형으로 부터 기능등과 같은 구현시의 문제를 충족시키기 위하여 사용되며, 사람과 상호 작용 요소는 화면과 보고서 양식을 포함하여 인간과의 접속을 설계하기 위한 단계이다. 타스크 관리 요소는 다중 타스크를 요구하는 시스템에서 자료 획득, 분산 장치의 제어 책임, 여러 개의 윈도우가 입력을 위해 선택될 때 사람과의 접속등, 다수 사용자의 사용자 타스크에 대해 다중 부시스템, 소프트웨어 구조등을 설계한다. 또한 자료 관리 요소는 앞 단계에서 만들어진 클래스를 저장하고 검색할 수 있도록 설계한다.

2.3 Martin과 Odell의 객체 지향 분석 및 설계

Martin과 Odell[12]의 객체 지향 분석 및 설계 방법은 정보 공학(IE)의 기업 모델링, 업무 영역 분석, 시스템 설계, 구축으로 이루어지는 시스템 개발 방법론을 기업의 전반적인 영역에 걸쳐 정보 체계의 계획, 분석, 설계, 구축을 위한 객체 지향 구조 및 행위를 분석, 설계하여 구현하는 방법이라 할 수 있다. 그리고 이 방법론의 설계 단계는 객체의 정적인 구조를 설계하는 객체 구조 설계와 동적 구조를 설계하는 객체 행위 설계가 있다.

객체 구조 설계는 객체 구조 분석에서 정의된 객체들의 타입과 그들의 관련성들에 대해 구축할 시스템의 기본적인 객체가 무엇이며, 각 객체가 소유한 고유의 자료 및 객체들 간의 상속 관계를 정의하고, 관련된 자료 구조를 파악하여 데이터베이스를 설계하는 단계이다. 또한, 객체 행위 설계는 객체 행위 분석에서 정의된 객체들의 이벤트, 활동, 상태들의 행위에 대해 시스템 구현의 동적 모델를 설계하기 위한 오퍼레이션 식별, 절차 또는 비절차, 입력 코드, 다이얼로그 설계, 그리고 레이아웃등의 메소드 설계등을 프로토타이핑을 통해 설계한다.

2.4 Booch의 객체 지향 설계

Booch[9]의 방법은 설계의 문서화를 중요시하는 방법으로, 다이어그램을 중심으로 개발하는 특성이 있다. 시스템 개발을 위한 설계 단계의 객체 및 클래스는 시스템 구조를 모형화한 객체 흐름도를 이용하여 추출하고, 객체들 간의 인터페이스 관계를 Ada 프로그램으로 변환시키는 설계 방법을 제안하였다.

Booch의 객체 지향 설계의 특징은 다른 방법들과 다르게 분석과 설계 단계를 구분하지 않았으며, 분석 동안에 이용한 객체 모델을 설계 단계에 그대로 적용하고 있다. 시스템은 논리적 관점인 정적 모델과 물리적 관점인 동적 모델로 크게 나누어 설계되어진다.

논리적 관점은 시스템 구현과는 밀접한 관계가 없이 시스템 주요 추상체 식별과 의미 기술을 위해 사용되어지며, 물리적 관점은 구현을 위한 소프트웨어와 하드웨어 구성품을 기술하는데 이용되는 모듈 구조나 프로세스 구조등을 의미한다.

2.5 기존 설계 방법들의 문제점 및 개선 방향

객체 지향 소프트웨어 개발 방법은 실 세계의 다양한 관계성과 제약 조건등을 함께 고려하여, 실 세계를 있는 그대로 소프트웨어 시스템에 사상함을 추구한다. 따라서 OMT에서는 객체 모델을 중심으로 OMT의 표기법을 이용, 이러한 실 세계의 다면성을 표현한다. 그리고 그밖의 기존 객체 지향 방법들은 소프트웨어 개발 중 분석이나 설계 단계에 치중한 객체 지향 개념의 이론적 적용에 중점을 두고 있다[21, 22].

이러한 방법들의 문제점은 OMT의 경우,

첫째, 분석 단계의 모델링 결과를 설계 단계에서 객체 모델을 기본틀로 하여 표기의 차이만을 이용 동적, 기능 모델을 표현함으로서 규모가 크고 복잡한 시스템을 개발할 때 프로그래머가 코딩시 분석 단계의 정보 산출물을 참조해야 한다는 번거로움과 설계도의 복잡도가 더욱 가중된다는 점이다.

둘째, 분리된 각각의 설계 모델을 그대로 사용함으로서, 실 업무 적용에 있어 사용 용이성이 결여되어 있다.

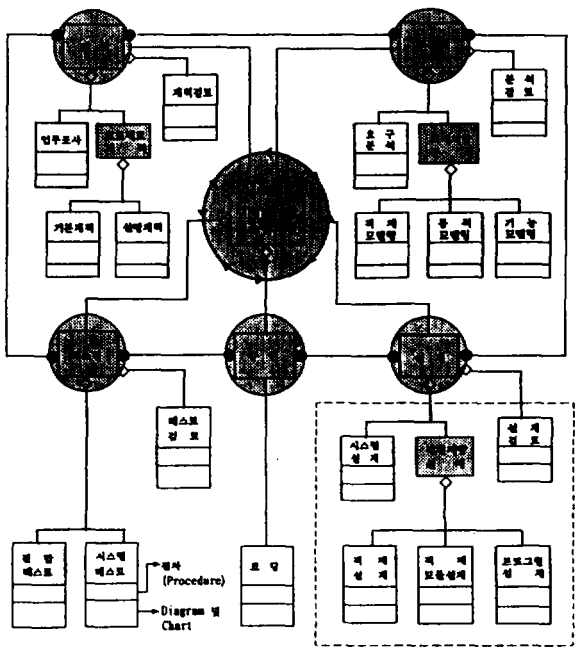
셋째, 설계서의 원시 코드 불일치로 인한 소프트웨어의 생산성 저하와 유지 보수의 어려운 점이 있다.

넷째, 객체 지향 분산 컴퓨팅을 위한 방법 제시가 없어 시대적 발전 요구에 만족 스러운 지원을 하지 못하는 결과를 초래한다.

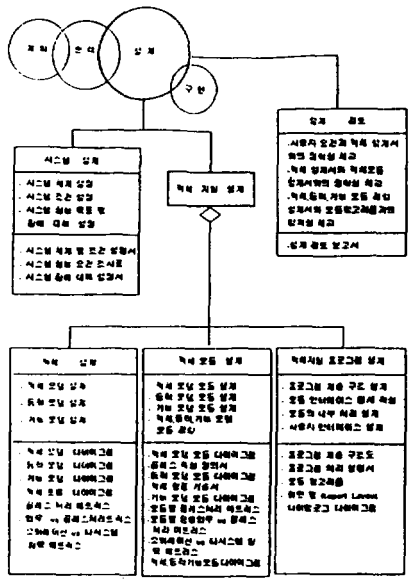
다른 방법들은 소프트웨어 개발 단계들 간에 연관 관계 및 설계 정보의 상세 내역이 결여됨으로서 실 시스템에 응용할 경우 완전한 시스템 구현을 기대할 수 없게 된다. 그러므로 실 세계의 다면성을 시스템화 할 수 있도록 효과적인 모델링 방법과 이에 대한 체계적이고 정형화된 설계 절차와 명세화 방법이 절실히 요구된다.

3. 개선된 객체 지향 설계 방법 및 절차

본 논문이 지향하는 설계 방법은 OMT의 분석 모델을 바탕으로, 설계 절차를 세분화하고, 명세화 방법을 단순화하여 실 업무에 적용할 수 있는 실용적인 제안 설계서를 생성하는데 목적이 있다. 따라서 (그림 1)과 같은 객체 지향 소프트웨어 개발 방법론을 제시하여, (그림 2)와 같이 설계 단계를 시스템 설계, 객체 설계, 객체 모델 모듈 설계, 객체, 동적, 기능 모델 모듈 결합 설계, 프로그램 설계등으로 세분하고, 상향식 분석 결과를 하향식 접근 방법에 따라 단계적으로



(그림 1) 객체 지향 소프트웨어 개발 단계
(Fig. 1) Procedure for object-oriented software development



(그림 2) 객체 지향 설계 과정
(Fig. 2) Processing of object-oriented design

단위 클래스에 대한 프로그램 단위의 전환 정보를 추출하도록 한다.

즉, OMT의 시스템 다면성 표현을 위한 객체, 동적, 기능 모델 결합 방법을 개선한 객체, 동적, 기능 모델를 정형화된 분해 방법[21, 22]을 적용하여 개선된 다이어그램들을 결합하여 모듈 결합 다이어그램과 프로그램 처리 설명서를 제시한다. 그러므로 실 세계의 설계 모델과 구현 모델, 그리고 객체 지향 프로그래밍 언어의 구문 구조 사이에 최대한 일치감을 갖도록 하여, 코딩 작업의 편의성과 구현 시스템의 완전성을 가져올 수 있게 한다.

이 과정들은 OMT의 다이어그램 표기법을 쉽게 활용할 수 있도록 변형하고, 챠트 표기 방식을 제시하여 효과적인 가시화 및 문서화된 정보의 실용성을 향상시킬 수 있도록 한다.

본 논문에서 제안하려는 설계 방법을 보다 효과적으로 기술하고, OMT의 세가지 모델을 표현할 수 있도록 하기 위해 병원 종합 관리 업무 중 간호관리 시스템의 예를 들도록 한다. 간호관리 시스템은 간호행정 관리, 간호업무 관리, ORDER 수행 관리등으로 분류하여 객체 모델을 설계하고, 본 논문에서는 ORDER 수행 관리를 중심으로 설계 단계에서 프로그램 구현 모델로의 자연스러운 전환이 되는 일련의 설계 절차와 명세화를 보이고자 한다.

본 논문에서 제안하는 객체 지향 설계 단계의 전체적인 과정은 다음과 같다[21, 22].

(1) 객체 설계(객체 모델 설계, 동적 모델 설계, 기능 모델 설계)
(2) 객체 모듈 설계(객체 모델 모듈 설계, 동적 모델 모듈 설계, 기능 모델 모듈 설계)
(3) 객체, 동적, 기능 모델 모듈 결합 설계
(4) 프로그램 설계(프로그램 계층 구조 설계, 모듈간 인터페이스 명세 작성, 프로그램 내부 처 리 설계, 사용자 인터페이스 설계)

4. 실 시스템 적용 예

여기에서는 3장에서 제시한 개선된 객체 지향 설계 방법 및 절차를 바탕으로 병원 종합관리 중에서 간호관리 업무를 실 시스템의 적용 예로 보이고자 한다.

4.1 객체 설계(object design)

본 논문의 객체 설계 과정은 분석 단계의 분석 모델과 시스템 구조 설계 과정의 세부 구현 사항을 기반으로 설계의 최종 산출물이라 할 수 있는 프로그램 처리 설명서의 산출 근거를 제시하는 작업을 수행한다. 즉, 분석 설계에서 생성한 객체 모델, 동적 모델, 기능 모델등과 객체 설계 단계에서 생성한 설계 모델들을 이용하여, 소프트웨어 내부 구성 요소와 외부 구성 요소들 간의 접속 관계를 정의하고, 소프트웨어 구성 요소의 세부 기능을 서술하며, 세부 자료구조와 소프트웨어 구성 요소들을 통합할 때 고려해야할 사항들을 기술한다.

이에 따라 객체 설계 절차는 객체 모델 설계, 동적 모델 설계, 기능 모델 설계등으로 세분하고, 객체 설계 과정의 명세화는 객체 모델, 동적 모델, 기능 모델의 상세 다이어그램과 차트로 나타낸다.

4.1.1 객체 모델 설계(object model design)

객체 모델 설계 과정은 분석 모델에서 실 세계 관점의 객체를 구현 관점의 객체로 전환시키는 과정이다. 이 과정에서 설계자는 다음과 같은 절차를 따르도록 한다[6, 7, 8, 9, 10, 13, 14, 15, 17, 19, 21, 22].

(1) 객체의 분류
(2) 변경된 클래스의 추가 및 삭제
(3) 추상화 클래스 설정
(4) 객체간의 인터페이스 정의
(5) 클래스 속성에 대한 후보 식별자 정의

여기에서 (1), (4)항은 기존의 Booch[9]나 J.Rumbaugh [14]의 방법과 유사하나 아래 항목의 세부 내용을 살펴보면 차이가 있음을 알 수 있다.

(1) 객체의 분류는 추상화 객체, 내부 객체, 외부 객체, 인터페이스 객체, 분산 객체등으로 분류 하며, 이들을 구체적으로 기술하면 다음과 같다.

① 추상화 객체는 규모가 크고 복잡한 시스템을 부분 시스템으로 분해하기 위한 객체이며, "is-a"의 관계 형태로 표현된다.
② 내부 객체는 시스템 내부에 존재하는 객체로서, 다시 내부 객체를 세분화하면 엑터(actor), 서버(server), 자료 객체(data object)로 분류할 수 있고, 엑터는 자신의 연산은 제공하지 않지만 다른 객체의 연산은 이용할 수 있는 객체이다. 자료 객체는 엑터, 에이전트, 서버등 이 직접 접근 가능한 자료 저장소에 해당하는 객체이다.
③ 외부 객체는 시스템의 범위 밖에 존재하는 객체로서 내부 객체와 직접적인 관계가 있는 객 체이다.
④ 인터페이스 객체는 객체 모듈을 형성하는 객체로서의 역할을 한다.
⑤ 분산 객체는 이기종 컴퓨터 시스템으로 구성된 클라이언트/서버 환경에서 상호 전송할 수 있을 뿐 아니라 협동적으로 작업할 수 있는 자체 처리 능력이 있는 클라이언트/서버의 구 성 요소이다.

(2) 변경된 클래스의 추가 및 삭제는 분석 단계에서 추출된 클래스들에 대한 재정의 과정으로 누락된 클래스의 추가와 상속성에 의한 통합, 축소등을 정리하여 분석 및 계획 단계로의 프로토타이핑에 의한 작업 전환을 한다.

(3) 추상화 클래스 설정은 객체 분류 단계에서 추출한 추상화 객체를 바탕으로 한다. 모듈이 어떤 기능을 수행하도록 구현하도록 구현되었는가에 대한 추상적인 사상을 중심으로, 객체의 상태와 행동 양식, 메시지등에 대한 행동이 유사한 객체들의 공통된 사항 등을 하나의 추상 적인 클래스로 정의한다. 따라서 추상화 클래스는 데이터와 이 데이터를 조작하기 위한 연 산을 하나의 모듈로 결합시킨 것으로 정보 은닉의 효과를 얻을 수 있으며, 객체들의 집중화 (clustering)로 설계서상의 복잡도 감소 및 데이터 활용의 효율성을 증대시킬 수 있다.

(4) 객체간의 인터페이스 정의는 추상화 클래스에 대한 구현 사항과 이를 사용하는 클래스들간 의 경계를 제공함으로서, 사용자에게 보여질 상세한 구현 사항(how)의 내용을 제한하고, 해 당 구현이 제공하는 기능(what)들만을 기술해 사용자에게 보여 줄 수 있게 한다.

이러한 인터페이스를 명확하게 정의함으로서 표면적으로 드러나는 오류 이외에도 내부적으로 감추어질 수 있는 의미상의 오류를 방지할 수 있다.

(5) 클래스 속성에 대한 후보 식별자(candidate key)

정의는 실 시스템 구현시 클래스의 속성을 대표하며 처리되는 식별자를 추출하여 표기함으로서 객체 및 클래스간의 관련성을 명확하게 유지시켜 준다. 즉, 객체간의 관련성에 대한 모호함을 없애고, 단일성(unique)을 유지하기 위한 과정이다. 여기에서 식별자란, 각 객체에 대해 실행 단계에서 도출한 속성으로 부터 객체를 대표할 수 있는 객체를 말하며, 후보 식별자는 객체 모델 모듈 설계에서 주 식별자 (primary-key)나 부 식별자(sub-key)가 될 수 있다. 그리고 모든 객체 및 클래스는 시스템 구현시 하나 이상의 식별자를 포함하여야 한다.

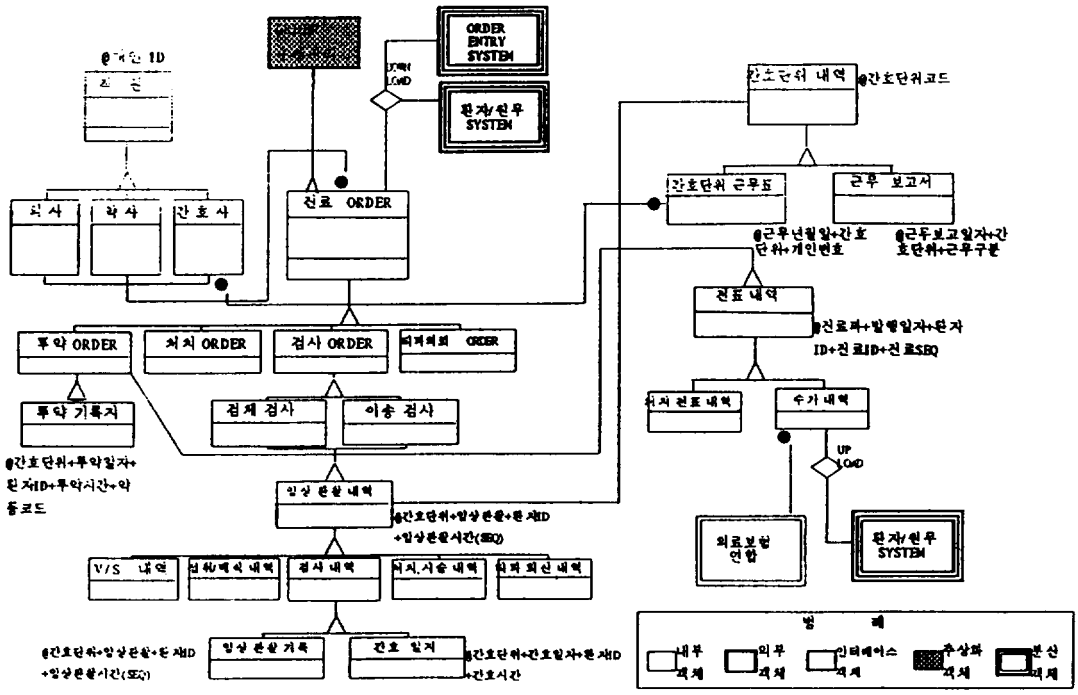
이와 같은 객체 모델 설계 단계의 명세화를 위해서는 (그림 3)과 같은 객체 모델 다이어그램을 이용한다. 그러나 클래스들의 속성 항목은 본 논문에서는 생략하고 4.3.1 객체 모델 모듈 설계의 (그림 7) 객체 모델 모듈 다이어그램에서 제시한다.

4.1.2 동적 모델 설계(dynamic model design)

동적 모델 설계는 객체 모델 설계 후 분석 모델(동적 모델)을 바탕으로, 객체들이 수행하는 연산의 순서, 제어 흐름, 그리고 상호 작용하는 관계를 다루는 모델로, 시간에 따라 객체들과 객체들간의 관계성들이 변화되는 시스템의 제어 관점을 상세히 나타낸다. 이는 공유된 객체들을 통하여 상호작용하는 객체들에 대한 상태도(SD), 상태도를 병행하여 수행하는 객체들의 집합으로 표현되며, 다중 상태도로 구성된다. 즉, 전체 시스템에 대한 활동 패턴을 보여줌으로서 전체 시스템의 제어 구조를 나타낸다[4, 14].

그리고 동적 모델을 가시화하는 제어 흐름, 제어 프로세스, 제어 저장등은 기능 모델과 차별을 두어 점선으로 표시하여, 객체들의 제어 측면을 명확하게 정의한다. 또한 제어 흐름의 연속(--))과 이산(discrete) 의미는 (-)))로 표현하며, 분산 객체(distributed object)의 제어 흐름은 (--))))로 나타낸다[22].

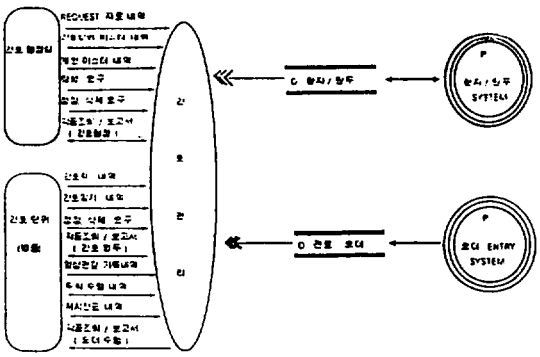
그러나 동적 모델 분석 및 설계 절차와 명세화 방



(그림 3) 간호관리 시스템 객체 모델 다이어그램
(Fig. 3) Diagram of object model for nurse management system

법등은 실 시간 처리 시스템에 적합한 방법이다. 따라서 본 논문에서 나열한 절차들을 일반 응용 업무에 반드시 적용 시킬 필요는 없다. 즉, 개발되어질 시스템의 성격을 정확히 분석하여 필요한 절차와 명세만을 선택하여 사용하면 된다.

본 논문에서는 동적 모델 설계의 절차적인 가시화 방법은 생략하며, 4.2.2 동적 모델 모듈 설계에서 모듈에 대한 객체들의 활동 기술서를 제시한다.



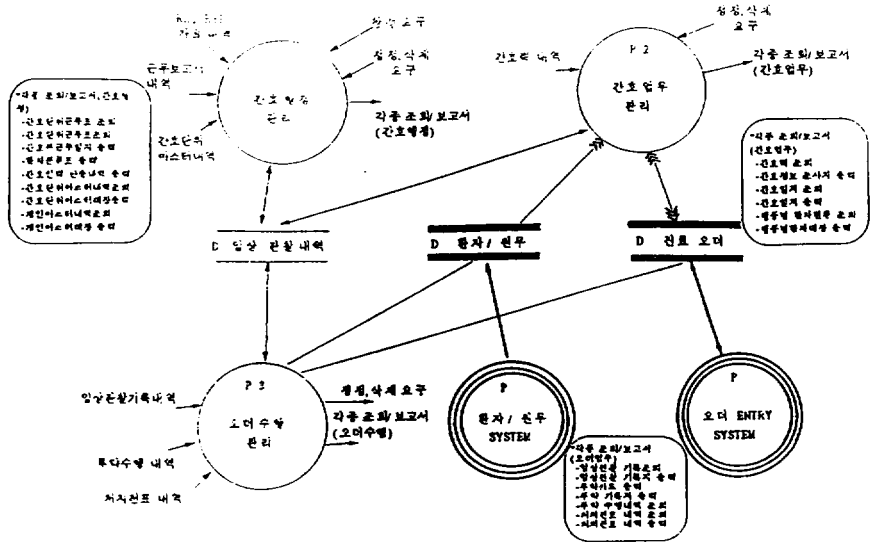
(그림 4) 간호관리 시스템 객체 흐름 배경도
(Fig. 4) Diagram Context of object flow for nurse management system

4.1.3 기능 모델 설계(functional model design)

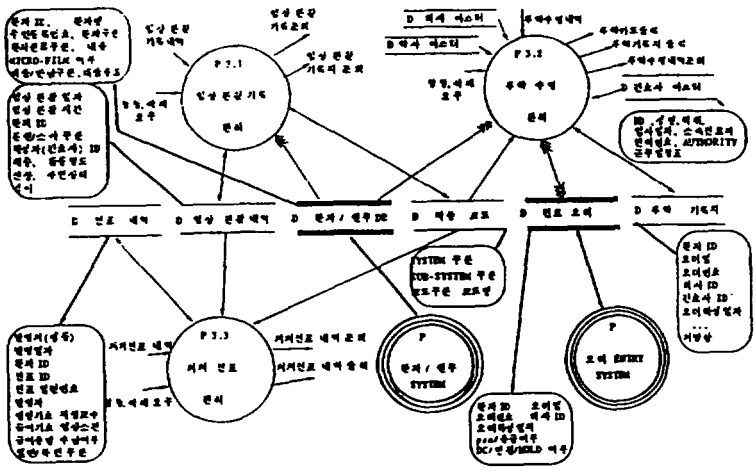
기능 모델 설계는 동적 모델 설계와 같이 객체 모델 설계 후 분석 모델(기능 모델)을 바탕으로 시스템에서 수행하는 데이타, 또는 함수에 의한 데이타의 기능 종속성에 대해 객체 흐름도(OFD)를 이용하여 프로세스의 순서나 객체의 구조에 무관하게 하나의 시스템에 여러값들이 어떻게 처리되어 계산되어지는가를 시스템 기능 관점에서 정형적으로 상세하게 표현한다[2, 14, 21, 22].

본 논문에서 제시하는 객체 흐름 다이어그램상의 저장소는 클래스를 나타내며 (그림 4)와 같이 분석 모델의 입/출력 값에 대한 처리의 흐름을 중심으로, (그림 5)' (그림 6)의 객체들 동작(operation)을 표현하기 위해 객체 변환 프로세스, 객체를 생성하거나 소거하는 행위자(actor), 객체 사전(object dictionary), 객체 흐름(object flow)등을 실제적으로 기술하며, 분석 모델이 객체 처리의 무엇을 나타내면 설계 모델에서는 어떻게 처리되어지는가의 상세한 정보를 기술하는 단계라 할 수 있다.

(그림 5)와 (그림 6)에서 클래스명과 프로세스들은 데이타베이스 관련 시스템의 내부와 분산으로 구분하여 나타내며, 프로세스들은 동적 모델에서 프로그램 구현의 내부적인 이벤트와 같은 의미를 갖는다.



(그림 5) 간호관리 시스템 객체 흐름 다이어그램
(Fig. 5) Diagram of object flow for nurse management system



(그림 6) ORDER 처리 객체 흐름 다이어그램
(Fig. 6) Diagram of object flow for ORDER processing

〈표 1〉 클래스 처리 메트릭스
〈Table 1〉 Matrix of class processing
(I : insert, M : modify, D : delete. A : all, Q : query)

No	class 명 Operation	전표 내역	임상 관찰 내역	각종 코드	환자/원무	진료 오더	투약 기록지	의 사	약 사	간호사
1	투약 수행 등록			Q	I	I	I	I	I	I
2	투약 수행 정정			Q	I	I, M	I, M	I	I	I
3	투약 수행 삭제			Q	I	D	D	I	I	I
:	:									
12	처치 전표 조회	Q	Q	Q	Q			Q	Q	Q

〈표 2〉 응용 업무와 클래스 처리 메트릭스
〈Table 2〉 Application program versus matrix of class processing
(I : insert, M : modify, D : delete. A : all, Q : query)

No	업무명 Class	ORDER ENTRY SYSTEM	환자/원무 SYSTEM	검사 예약	임상 병리과	핵의학 체외	간호 관리	수술 마취	병력 관리	특수 검사부	약제부	혈액 은행	병리과	급식 영향	진단 방사선	치료 방사선	핵의학 체내	재활 의학
1	환자 MASTER	Q, M	Q	Q	Q	Q	Q	Q	Q	Q		Q	Q	Q	Q	Q	Q	Q
2	의사 MASTER	Q	Q				Q		Q				Q		Q		Q	
3	투약 ORDER	Q	Q, M				Q				Q							
:	:																	
70	검사내역	Q	Q	A	A	A	Q			A		A	A		A	A	A	A

〈표 3〉 오퍼레이션과 타 시스템 메트릭스
〈Table 3〉 OPERATION versus matrix of others system
(I : insert, M : modify, D : delete. A: all, Q : query)

NO	구 분 Operation	타시스템 명	클래스명	ATTRIBUTE 명	처리구분
1	근무 보고서 출력	환자/원무	입원 변경 정보 내역	환자 ID, 입원 일자 변경 진료과, 병동, 병실	Q
2	변동별 환자 현황 조회	환자/원무	재원 현황	재원 환자 ID, 병실	Q
:	:				
80	투약 수행 처리	ORDER ENTRY, 환자/원무	ORDER 내역, 환자/원무 내역	ORDER일, 약품 코드 용법, 처방량, 환자 ID, 환자명, 주민등록번호, 환자 구분 내용	I, M

즉, 기능 모델을 가시화하는 객체 흐름, 객체 프로세스, 객체 저장, 분산 객체등은 동적 모델에서 언급한 사항과 같이 차별을 두어 객체들의 처리 측면을 명확하게 정의한다. 그리고 〈표 1〉, 〈표 2〉, 〈표 3〉과 같이 클래스 처리 메트릭스, 응용 업무와 클래스 처리 메트릭스, 오퍼레이션과 타시스템 항목 메트릭스를 본 본문에서 새롭게 제시함으로서 객체들의 프로세스 또는 이벤트에 대한 관계를 보다 명확하게 나타낼 수 있다.

4.2 객체 모듈 설계

본 논문의 객체 모듈 설계는 객체 설계 과정에 이어서 단위 클래스를 하나의 모듈로 가정하고, 모듈에 대한 객체 모델 모듈, 동적 모델 모듈, 기능 모델 모듈 설계를 세부 절차로 하여 시스템을 구성하는 클래스들에 대한 데이터(객체), 기능, 제어 측면을 가시화한다고 할 수 있다. 즉, 객체, 동적, 기능 모델 모듈 결합 다이어그램을 산출할 수 있도록 객체 모델 설계로 부터 모듈 생성 원칙에 따라 모듈을 정의하고, 정의된 모듈에 대한 객체(데이터)측면, 동적(제어)측면, 기능(프로세스) 측면등을 상세하게 설명한다[21, 22].

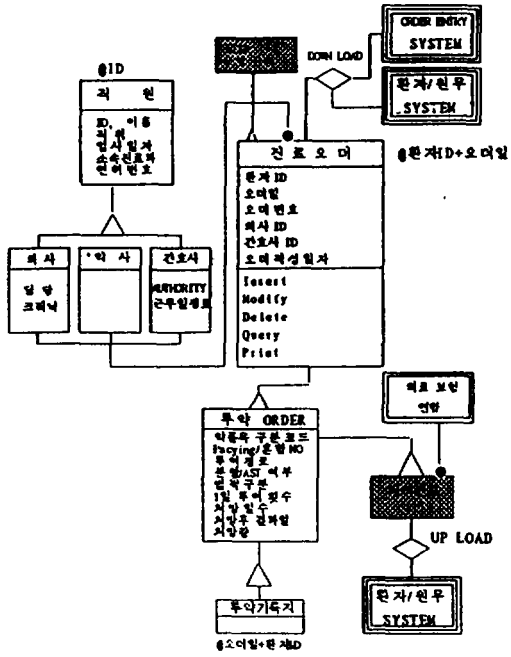
4.2.1 객체 모델 모듈 설계

본 논문은 하향식 설계 접근 방법에 따라 객체 모델 설계 단계에서 모델을 구성하는 모듈을 분해하여, 분해된 추상화 클래스에 대한 세부적인 구성 객체(클래스)들의 정적인 부분을 단계적으로 좀 더 상세하게 표현함으로서, 구현시 객체의 특성을 즉시 알 수 있도록 다음과 같은 사항을 수행한다.

(1)각 객체의 속성과 메소드들의 실체를 정의하고, 상속성을 이용한 객체의 추가 및 삭제 작업을 한다. 이렇게 했을 때 모듈 분해력에 대한 만족을 얻을 수 있다.

(2)이러한 모듈의 정적인 관점을 가시화하기 위해 객체 모델 다이어그램으로 부터 추상화 클래 스 단위로 분해된 객체 모델 모듈 다이어그램(OMMD : Object Model Module Diagram)을 생 성한다. 이러한 객체 모델 모듈 다이어그램을 객체 모델 다이어그램으로 부터 분해할 때는 객체 모듈과 관련된 인터페이스 클래스와 분산 클래스가 누락되지 않도록 하여야한다.

(3)클래스 속성 정의서 작성은 객체들의 상속 관계를 챠트로 구성하여, 구현시 코드로의 사상이 용이하게 지원될 수 있도록 한다. 즉, 상속 관계를 갖는 클래스들을 하나의 챠트에 구성함으로서 클래스 간의 인터페이스 및 상속 관계를 쉽게 파악할 수 있다. 그러므로 속성 정의서는 식별된 객체가 문제 영역에서 어떻게 기술되었는가를 고려하여 인스턴스 접속을 정의하고, 구현되어질 객체 상태 정보를 더욱 세밀히 정의하여, 그 객체의 책임을 완수하도록 한다. 즉, 하나의 객체가 다른 객체와 필요한 사상을 요구함으로서, 객체가 기본적으로 문제영역에서 해당되는 역할을 수행하도록 한다. 따라서 기존의 개발 방법론에서와 같이 데이타베이스 설계 단계 절차가 필요없이 객체 속성 정의서를 참조하여 데이타베이스 구축이 가



(그림 7) 투약 수행 객체 모델 모듈 다이어그램

(Fig. 7) Diagram of object model module for medication execution

클래스 속성 정의서	PROJECT 명	시스템명		작성자	작성일	문서 번호
	기획 중점 분야	진료 정보		이 우 성	95.11.28	NUO-007
CLASS 명	CLASS-ID	내부 처리 형태 및 속성 내용				
진료 ORDER (Parent)	OBJCT00					
속성 (영문)	속성 (영문)	KEY 구분	TYPE	허 용 값	속 성 내 용	
환자 ID	PTIDUX	PK	CHAR(08)	일련번호 < 9,999,999	일련 번호 +check 속성	
오더 일	PRACDT	PK	DATE(08)		시스템 일자	
오더 번호	ORDSEQ	PK	CHAR(05)	A : 작성자 구분 0:주치의 1:인턴 2:전공 교수 3 수석의 4:일직의 B : 처방 구분 0:제한 1:처방제정 2:처방후 가지집 약 처방후 점서 C : 오더 종별 C:처치/에어 M:투약 S:검체 검사 D:오더일련번호	□ □ □ □□ A B C D	
의사 ID	DOCTID	FK	CHAR(05)	1 ~ 1999		
오더 작성일자	ODRTIM		DATE(08)			
간호사 ID	NURSID	FK	CHAR(05)	2000 ~ 3999		
CLASS 명	CLASS-ID					
검사 오더 (Drived)	OBJCT01	내부 처리 형태 및 속성 내용				
검표 ID	SLIPID	PK	CHAR(04)			
검사처방전표수명	DOCTNM		CHAR(08)			
검사실 접수 여부	TAKETYN		CHAR(01)	Y, N		
예약 확인 여부	TSLONE		CHAR(01)	1 ~ 4		
결과 수신 여부	RCVTYNO		CHAR(01)	Y, N		
원무 접수 여부	WNTYNS		CHAR(01)	Y, N		
진료과 코드	JDKCAC	FK	CHAR(03)	IM : 내과 GS : 일반 외과 TS : 흉부 외과 NS : 신경 외과 : PDM : 소아 치무과	□□□	
수정 최일 일자	EXTEYINE		DATE(08)			
CLASS 명	CLASS-ID					
검체 검사 오더 (Drived)	OBJCT02	내부 처리 형태 및 속성 내용				
검체 유형 CODE	SATYPE	PK	CHAR(02)			
검체 BARCODE	SBARCD	FK	CHAR(12)			

(그림 8) ORDER 클래스 속성 정의서

(Fig. 8) Sheet of attributes definition specification for ORDER class

능하게 한다.

또한 (그림 8)의 속성 정의서 예에서 알 수 있듯이 기존의 데이타베이스를 구축하기 위한 정보로 클래스는 화일, 속성명은 필드명이 되며, 속성들이 모여서 하나의 레코드를 형성하고 있다.

그리고 키 구분, 속성들의 형태 및 크기, 허용한 속성들의 내용등을 하나의 구현될 객체에 대한 정보로 직접 제공하고 있음을 알 수 있다.

객체 속성 정의서 작성 방법은 객체 모델 모듈 다이어그램의 상속에 관한 표현을 적용하여 상속을 받은 클래스의 기술은 재정의(redefinition)한 속성과 추가된 속성만을 기술하면 된다.

4.2.2 동적 모델 모듈 설계(dynamic model module design)

동적 모델 모듈 설계에서 수행해야 할 내용으로는, 시스템의 내부에서 발생하는 상태 변화를 일으키는 행위인 이벤트(event) 즉, 내부 이벤트(internal event)를 중심으로, 특정 사건이 발생할 때 마다 시스템에 의해 수행되어야 할 일련의 동작 즉, 반응(response)을 나타내야한다. 또한, 사건들을 묘사하여 의사 결정을 하기 위한 모델의 입력 인자를 결정하며, 시스템이 어떤 행동을 취하고 나서 갖게 되는 결과와 이어지는 이후 행위를 결정할 수 있는 근거가 되는 객체의 상태를 기술한다. 그리고 현재 상태에서 다음 상태로 갈 수 있도록 시스템 또는, 환경의 변경을 유발시키는 상태 전이(state transition)와 시스템 외부인 실 세계의 특정 시점에서 발생하는 활동등을 기술한다.

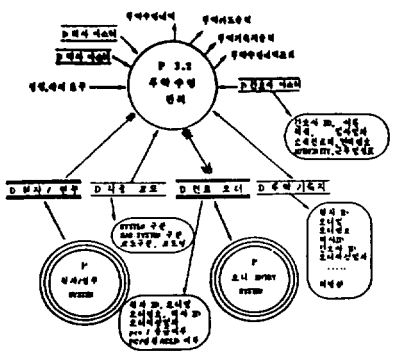
4.2.3 기능 모델 모듈 설계(function model module design)

객체 모델에서 일어나는 각종 동작 상황에 따라 객체 흐름의 변화는 빈번하게 일어난다. 따라서, 기능 모델 모듈 설계는 객체 모델 모듈 설계 이후에 행해지는 설계 과정으로, 객체 설계 단계인 기능 모델을 바탕으로 모듈별로 분해하여 설계한다.

이러한 모듈의 객체 흐름을 중심으로 표현하는 설계과정은 (그림 10)과 같이 특정 모듈의 활동 상황에 따라 클래스로 부터 참조와 갱신이 수반되어, 항상 새로운 정보를 유지할 수 있게 하며, 현황 자료(객체)의 현재 상태를 파악할 수 있도록 유지 되어야한다.

업 등	PROJECT 명	SYSTEM 명	작 성 자	작 성 일	문 서 번 호
가 등 시	병원 종합 관리	간호 관리	정 계 등	95. 11. 26	NUD-009
EVENT 명		투 약	처 리 (Insert, Modify, Delete, Inquery)		
요 청 자	활 등	행 위 자	서 비 스		
간 호 사	선택 한다	투약 처리 기능	투약, 등록, 정정,삭제,		조회
간 호 사	등록 한다	투약일자,	메뉴 화면 DISPLAY		
		환자 ID 등록	각종 코드	CLASS	WRITE
			진료 ORDER	CLASS	WRITE
			환자 / 원무	CLASS	WRITE
			투약 기록 내역	CLASS	WRITE
			의사	CLASS	WRITE
			약사	CLASS	WRITE
			간호사	CLASS	WRITE
간 호 사	등록 한다	투약 내역 정정	진료 ORDER	CLASS	REWRITE
			투약 기록 내역	CLASS	REWRITE
			진료 내역	CLASS	REWRITE
			투약 내역 정정완료 메시지		DISPLAY
			(투약 내역 정정이 된 됴 되었습니다 다음 작업을 처리하 시오)		

(그림 9) 투약 수행 모듈 활동 기술서
Fig. 9) Sheet of module dynamic description for medication execution



(그림 10) 투약 수행 모듈 기능 다이어그램
(Fig. 10) Diagram of module function for medication execution

또한 기능 모델 단계에서 작성된 메트릭스들도 모듈별로 작성을 하여 프로그램 단위별 처리 과정의 생성, 정정, 삭제, 조회, 읽기등 상세한 정보를 기술한다. 본 본문에서는 생략하기로 한다.

4.3 객체, 동적, 기능 모델 모듈 결합 설계

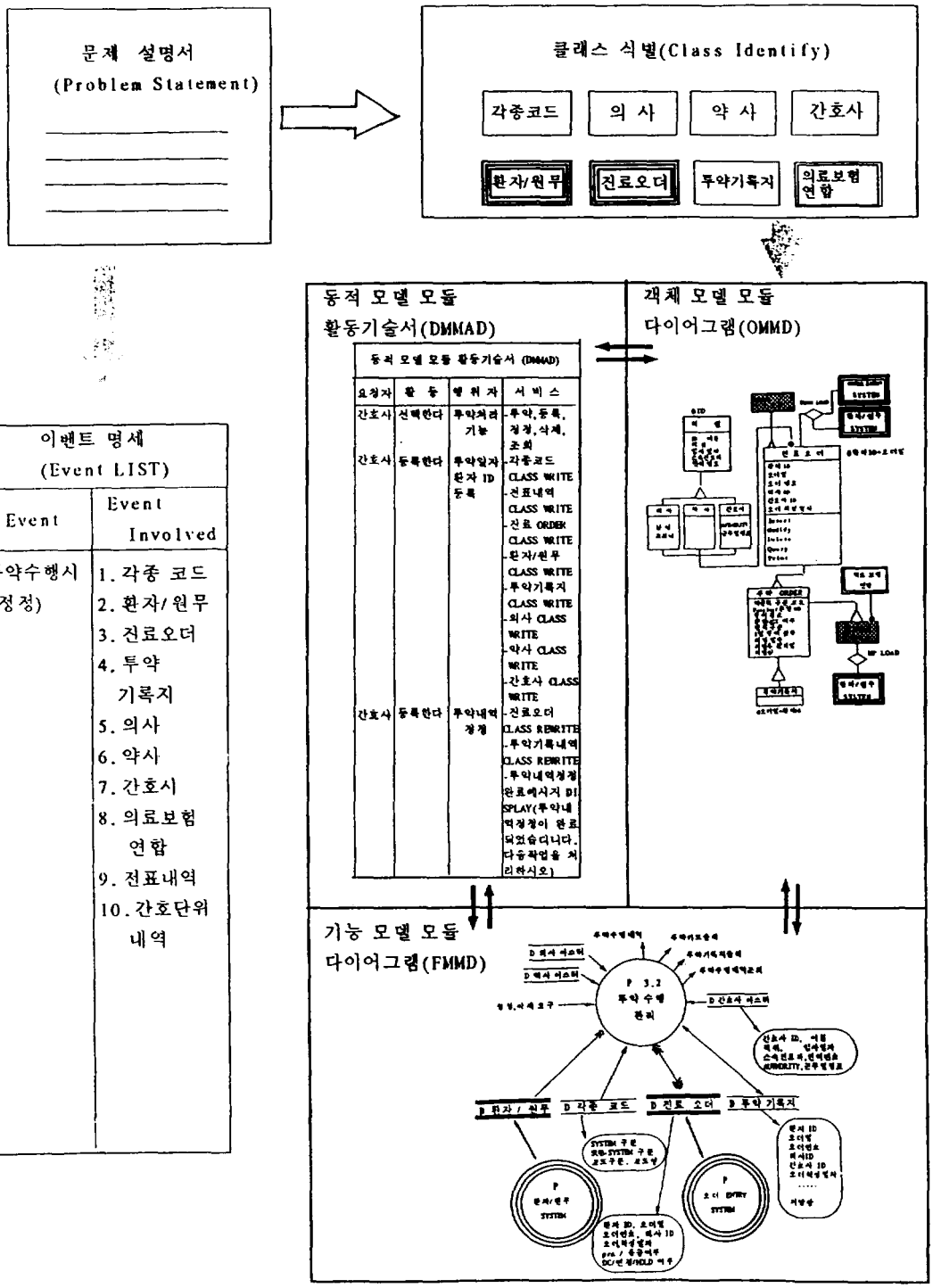
OMT의 객체, 동적, 기능 모델 결합 문제점을 해소하고, 소프트웨어의 부품화에 따른 생산성 향상과 버전 관리의 원활한 유지를 위해 단위 클래스(모듈)에 대한 객체, 동적, 기능 모델 모듈을 각각 독립된 부품으로 정의하고, 이를 부품 결합 원리에 의해 결합한다. 그리고 이에 따른 명세화로 결합 다이어그램을 작성한다[5, 21, 22].

4.3.1 객체, 동적, 기능 모듈 결합 다이어그램 작성

객체 모델 설계에서 객체 모듈 설계에 이르는데는 프로그램 단위까지 반복적인 모듈의 분해가 이루어진다. 이러한 모듈은 각각 객체, 동적, 기능 모델 모듈 차트나 다이어그램을 산출물로 갖게 되며, 이들 각각은 코딩을 위한 독립된 정보를 가지고 있다. 따라서 (그림 11)의 예와 같이 결합 다이어그램의 형식에 단위 클래스(모듈)에 대한 객체, 동적, 기능 모델 모듈의 가시화된 명세를 결합시킴으로서 객체, 동적, 기능 모델 모듈 결합 다이어그램이 완성되어 이것을 프로그램 설명서의 작성에 활용되어지게 한다.

(그림 11)의 예에서 투약처리내역 클래스의 '정정'이라는 행위(behavior)는 어떤 객체를 대상으로 어떻게 무엇을 구현하는가에 대한 정보를 명확하게 알 수 없다. 이는 동적 모델 모듈 활동 기술서에서 이벤트 '정정'의 활동을 참조하고, 기능 모델 모듈 다이어그램에서 프로세스 '정정'에 관련된 해당 클래스(화일)을 참조한 후에 비로서 그 의미를 명확하게 알 수 있으며, 프로그래머는 코딩을 할 수 있는 것이다.

이처럼 객체, 동적, 기능 모델 모듈 결합 다이어그램이 주는 효과는 설계 단계의 정보를 동시에 정확하



(그림 11) 투약 수행 설계 모듈 결합 다이어그램

(Fig. 11) Diagram of design module combination for medication execution

PROGRAM 처리 설명서	PROJECT 명	SYSTEM 명	작 성 자	작 성 일	문서 번호
	병원 종합 관리	간호 관리	허 계 림	95. 11. 26	NUP-005

PROGRAM ID	NUP-005C
PROGRAM 명	투약 수행 정정
사용언어	Visual Basic 3.0
사용 DB	UniSQL

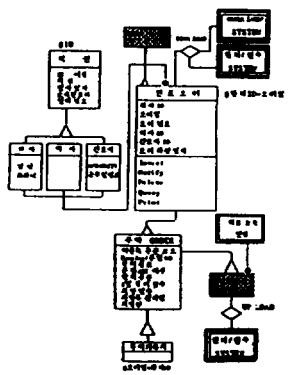
CLASS 참조(Reference)사항

CLASS 명	CLASS-ID	I/O
1.각종코드	NU01	I
2.환자/원무	NU21	I
3.진료오더	NU33	I/O
4.투약지특지	NU08	I/O
5.의사	NU02	I
6.약사	NU03	I
7.간호사	NU07	I
8.전표내역	NU12	I/O

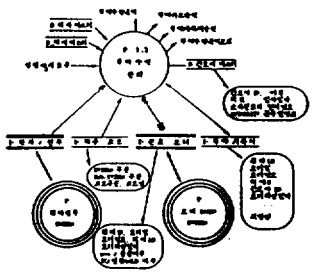
처 리 설 명

0.상병이 있어 찾아온 환자에게 증상을 확인한 후 투약 처방한 내역을 정정하는 program이다.
1.Key 항목 CHECK
 1)환자 ID,ORDER 일자(미등록시 해당항목을 정확히 입력하시오.MESSAGE DISPLAY)
2.입력 항목 CHECK
 1)환자/원무내역(미등록시 등록된 내용이 없습니다.MESSAGE DISPLAY)
 2)진료오더 내역(미등록시 등록된 내용이 없습니다.MESSAGE DISPLAY)
 3)의사,약사,간호사 MASTER 내역(미등록시 등록된 내용이 없습니다. MESSAGE DISPLAY)
 4)투약기록 내역(미등록시 등록된 내용이 없습니다.MESSAGE DISPLAY)
3.CLASS PROCESS
 1)각종 코드 CLASS READ 하여 코드내역 WRITE
 2)환자 / 원무 CLASS READ 하여 재원 환자 ID, 주민등록번호,환자 구분 WRITE
 3)진료 오더 CLASS READ 하여 ORDER일, 환자ID,오더번호,의사ID,간호사ID, 오더작성일자 REWRITE
 4)투약 기록지 CLASS READ 하여 투약기록내역 및 전표내역에 REWRITE
 5)의사 CLASS READ 하여 의사 ID WRITE
 6)약사 CLASS READ 하여 약사 ID WRITE
 7)간호사 CLASS READ 하여 간호사 ID WRITE

객체 모델 모듈 다이어그램 (OMMD)



기능 모델 모듈 다이어그램 (FMMD)



클래스	Operation
각종 코드	Q
환자/원무	I
진료오더	I,M
투약기록지	I,M
의사	I
약사	I
간호사	I
전표내역	I,M

동적 모델 모듈 활동기술서 (DMMAD)

요청자	활 동	행 위 자	서 비 스
간호사	선택한다	투약처리 기능	-투약,등록, 정정,삭제, 조회
간호사	등록한다	투약일자 환자 ID 등록	-각종코드 CLASS WRITE -전표내역 CLASS WRITE -진료 ORDER CLASS WRITE -환자/원무 CLASS WRITE -투약기록지 CLASS WRITE -의사 CLASS WRITE -약사 CLASS WRITE -간호사 CLASS WRITE
간호사	등록한다	투약내역 정정	-진료오더 CLASS REWRITE -투약기록내역 CLASS REWRITE -투약내역정정 완료메시지 DISPLAY(투약내역정정이 완료되었습니다. 다음작업을 처리하시오)

(그림 12) 투약 수행 프로그램 처리 설명서

(Fig. 12) Sheet of program processing description for medication execution

게 참조할 수 있게 함으로서, 구현 단계의 신속한 작업 진행이 가능함과 소프트웨어의 생산성 향상 및 유지 보수의 편의를 기대할 수 있다.

4.4 프로그램 설계

프로그램 설계는 객체 설계 결과와 객체 모듈 설계 결과를 이용하여 프로그램 계층 구조를 설정하고, 이들을 구성하는 각 모듈의 내부 처리 절차(logic) 및 인터페이스를 명세화하는 설계 단계의 마지막 과정이다[21, 22].

4.4.1. 프로그램 계층 구조 설계

프로그램 계층 구조 설계는 전체 시스템이 클래스별 모듈을 형성하여 하나의 객체적인 독립성을 가지고 있어, 계층 구조도에 의한 클래스의 상속과 관계성을 쉽게 파악할 수 있고, 유지보수에 활용되어질 수 있는 중요한 정보이다.

4.4.2 모듈간의 인터페이스 명세서(프로그램 처리 설명서)작성

모듈간의 인터페이스 명세 작성은 분석 단계의 정보와 실 시스템의 구체화된 실행에 대한 표현의 중간 변환 자료를 제공한다.

예를 들어 (그림 12)의 프로그램 처리 설명서는 객체, 동적, 기능적인 상태를 모듈별로 작성하여 프로그래머 코딩 작업의 능률과 문제점 파악의 용이성을 기할 수 있게 한다.

4.4.3 모듈 내부 처리 설계

모듈 내부처리 설계는 프로그램에서 모듈 단위로 처리되어지는 내역을 표현하는 것으로서 물리적 요구사항을 만족시킬 수 있는 최적의 알고리즘을 기술한다. 즉, 프로그램 구현(코딩) 단계로 작업 전환을 할 수 있도록 하는 설계 단계의 마지막 과정이라 할 수 있으며 의사코드(pseudo code)로 작성됨을 원칙으로 한다.

5. 설계 검증 및 비교 분석

설계 검증 과정은 요구 분석 단계에서 생성한 요구 명세서와 객체, 동적, 기능 모델에 대한 설계 단계에서의 정확한 반영에 대한 검토가 필요하다.

〈표 4〉 객체 지향 소프트웨어 설계 품질 평가 분석표

〈Table 4〉 Table of analysis for object-oriented software design quality estimation

Pressman의 S/W 품질 평가 기준	본 논문에서의 지원사항
1. 클래스간의 계층 구조 표현성	설계 단계를 객체 설계, 객체·동적·기능 모델 모듈 설계, 객체·동적·기능·모델 모듈 결합 설계, 프로그램설계 단계로 분류하여 가장 상위의 추상화 클래스,클래스, 이벤트 단위등의 하향식 분할 설계 절차를 제시하였음
2. 클래스간의 기능 독립성 (응집성)	객체 설계 단계(추상화 클래스, 클래스, 이벤트)를 기반으로 객체·동적·기능 모델 모듈 결합 설계 단계에서 시스템의 다면성을 나타내어 결합할 수 있는 방법을 제시하였음.
3. 반복적인 수정 작업 가능성	반복적인 수정 작업을 위해서는 설계 모델이 명확하게 개체 단위로 구분되어져야 하며, 최종 프로그램 처리단위 모듈을 표현 할 수 있어야 한다. 그러므로서 프로토타입으로 소프트웨어 개발 단계를 실행 할 수 있다. 본 논문에서는 이러한 단계를 정형화된 모듈 분해 기준을 통하여 제시하였음.
4. 클래스 단위의 상호 연결 강도 (결합성)	2. 클래스간의 기능 독립성과 반대되는 것으로 가능하면 결합도는 작을수록 좋은 것이다.
5. 모듈화	1. 항목에서 설명 하였으며 정형화된 모듈 분해 방법은 본문 (그림 3), (그림 4)에서 제시하고 있음.
6. 재사용성	객체 지향 방법의 핵심요소로 1.2.3.4.5항목의 모든 내용을 포함하고 있다.

본 논문에서는 J.Rumbaugh의 OMT와 정형화된 모듈 분해 방법을 적용하여 결합 설계 방법을 제안하였다. 그러나 객체 지향 기술을 적용한 정형화된 방법이라 할지라도 이것을 평가 할 수 있는 공식화된 평가 기준과 항목은 없다. 또한 다른 방법들과 비교 분석 하였을때 어느 방법이 얼마나 우수한지를 평가하는 것은 매우 어렵고 주관적인 견해라고 생각한다.

따라서 여기에서는 아래와 같은 Pressman[18]의 소프트웨어 품질 평가기준을 바탕으로 본 논문에서 제시하는 방법이 (1)-(6)의 세부 항목들을 지원하는가 여부에 관하여 〈표 4〉와 같은 분석표를 작성하여 비교 분석하고자 한다.

(1) 클래스간의 계층 구조 표현성
(2) 클래스간의 기능 독립성(응집성)
(3) 반복적인 수정작업 가능성
(4) 클래스 단위의 상호 연결 강도(결합도)
(5) 모듈화
(6) 재사용성

소프트웨어 품질 보증은 소프트웨어 공학과정의 각 단계에 적용되는 보호활동(umbrella activity)이라 할 수 있다. 따라서 소프트웨어 품질 보증은 방법과 도구, 그리고 정형화된 기술 검토, 테스트전략과 기법들, 변경제어를 위한 절차, 제어를 변화하기 위한 절차, 표준안에 일치하는지를 보증하는 절차등을 포함한다[18].

이와 같이 Pressman[18]의 소프트웨어 품질 평가기준을 본 논문에서 제시하는 방법이 지원하는 가를 비교 분석하였다. 그러나 앞에서 서술하였듯이 정량적으로 평가할 수는 없지만 소프트웨어 설계시 가장 중요한 정형화된 분해 방법과 결합 방법을 본 논문에서 제시하고 있다. 그리고 〈표 4〉에서 알 수 있듯이 여섯 가지 품질 보증 항목을 보다 근접하게 만족한다고 할 수 있다.

6. 결 론

본 논문에서는 객체 지향 분석을 토대로 정확한 시스템 구현과 개발자의 편의를 제공하기 위한 객체 지향 설계 단계의 단계별 절차 및 이를 가시화 할 수 있는 다이어그램과 차트를 제시하였다.

기존의 시스템 개발 방법들은 분석, 설계, 구현 단계별 의미 명확성이 부족한 문제점을 안고 있다. 즉, 사용자의 요구사항을 정확히 반영하고, 시스템이 수행해야 할 기능적, 비기능적 요구를 체계적으로 분석하여 완전하고 일관성 있는 명세서를 작성하여야 하는 소프트웨어 시스템의 목표를 지원하지 못하고 있다.

따라서 본 논문에서는 객체 지향 시스템 개발의 전체 생명 주기를 새롭게 제시함과 동시에, 객체 지향 분산 컴퓨팅을 위한 부분적 방법 제시 및 객체 지향 설계 단계에서 구현 단계로의 자연스러운 전환이 이루어질 수 있도록 객체, 동적, 기능 모델 모듈의 분해와 결합을 통하여, 효과적인 시스템 모델링과 가시화를 기대할 수 있으며, 소프트웨어 시스템 구현시 개발자의 편의 제공, 시스템의 완전성과 시스템 개발 완료 후 원활한 유지 보수를 기대할 수 있다.

앞으로의 연구 과제는 객체 지향 방법론이 추구하는 소프트웨어 부품의 재사용을 극대화하기 위한 분석, 설계, 구현 단계를 보다 체계적이고 일관성 있게 관리할 수 있는 버전 관리 및 자동화 도구를 사용하여 분석/설계 모델의 가시화함과 동시에 코드 생성까지 이루어지는 연구가 필요하다. 또한 오늘날 관심 항목으로 대두되고 있는 객체 지향 분산 컴퓨팅를 만족하는 보다 체계적인 설계 방법이 절실히 요구된다.

참 고 문 헌

[1] Ann L, Winblad, Samuel D. Edwards, David R, King, "Object-Oriented Software", AddisonWestey, pp. 178-182, 1991.

[2] Bernd Bruegge, Jim Blythe, Jeffrey Jackson & Jeff shuleft, "Object-Oriented System Modelling with OMT", OOPSLA, pp. 359-376, 1992.

[3] Brian Henderson-Sellers, "A Book of Object-Oriented Knowledge", Prentice-Hall, pp. 32-37, 72-76, 1991.

[4] Bran Selic, Garth Gullekson, and Paul T. Ward, "Real-Time Object-Oriented Modeling", John Wiley & Sons, Inc. 1994.

[5] C. Paul Allen, "Effective Structured Techniques",

Prentice-Hall, Inc. pp. 81-88, 1991.

[6] Deni Chamepeaux, Doug Lea, Penelope Faure, "The process Object-Oriented Design", OOPSLA, pp. 43-51, 45-62, 1992.

[7] Derek Coleman, Partrick Arnold, Stepanie Bodoff Chris Dollim, Helena Gilchrist, Fiona Hayes Paul Jeremaes, "Object-Oriented Development The Fusion Method", Prentice-Hall, Inc. pp. 96-100, 1994.

[8] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, "Design Patterns", Addison-Wesley Publishing Company, 1995.

[9] G. Booch, "Object-Oriented Design with Applications", the Benjamin/Cumming Publiching Company, Inc. pp. 155-184, 187-194, 1991.

[10] Gunther Blaschek, "Object-Oriented Programming with prototypes", Sprinferverlag Berlin Heidelberg, pp. 67-70, 262-263, 1994.

[11] I. Jacoson, M. Christerson, P. Jonson, G. Overgard, "Object-Oriented software Engineering A case Driven Approach", ACM, Inc, pp. 465-493, 1992.

[12] J. Martin, J. Odell "Object-Oriented Analysis & Design", Prentice-Hall, pp. 69-119, 1992.

[13] John D. Mcgregor David A Sykes "Object-Oriented Software Development: Engineering Software for reuse", van Nostrand Reinhold, pp. 113-140, 1992.

[14] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Lorensen, "Object-Oriented Modeling and Design", Prentice-Hall, Inc. pp. 84-49, 148-183, 227-229, 260-274, 1991.

[15] Kevin lano & Howard Haughton, "Object-Oriented Specification CASE studies", Prentice-Hall, Inc (UK), pp. 2-19, 32-24, 72-79, 1994.

[16] P. Coad, E. Yourdon, "Object-Oriented Design", Yourdon Press Computing Series, Prentice-Hall, 1991.

[17] Pedro Sousa and Jose Alves Marques, "Object Clustering in Persistent and Distributed Systems", Proceeding of the International, Workshop on Persistent Object Systems, Transaction, Provence, France, 5-9 September pp. 402-414, 1994.

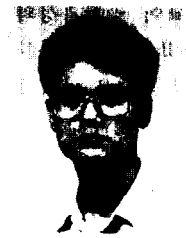
[18] R. S Pressman, "Software Engineering: A Practitioner's approach", Third Edition, McGRAW-HILL, pp. 318, 549-590, 1992.

[19] Stephen L. Montgomery, "Object-Oriented Information Engineering Analysis, Design and Implementation", Academic Press, Inc. pp. 119-141, 184-185, 201-202, 1994.

[20] 김수동, "ISO 9001 Compliant 객체지향 소프트웨어 품질시스템", 한국정보과학회지 제13권 제9호, pp. 27-45, 1995.

[21] 허계범, 최영근 "객체 지향 설계를 위한 모듈 분해 방법", 한국정보처리학회 논문지 Vol. 2,No 3, pp. 299-313, 1995.

[22] 최영근, 허계범 "객체 지향 소프트웨어 공학", 도서출판 한국 실리콘, pp. 237-282, 1995.



허 계 범

1989년 경기대학교 응용통계학과 졸업(학사)

1993년 광운대학교 전산대학원 전자계산학과(이학석사)

1988~1995년 (주)경인양행 전산부 과장

1995~현재 광운대학교 전자계산학과 박사과정

관심분야: 객체 지향 분석 및 설계, 데이터베이스, 프로그래밍 언어론, 객체 지향 분산 컴퓨팅

이 종 섭

1990년 호서대학교 전자계산학과(학사)

1993년 광운대학교 전산계산학과(이학석사)

1994~현재 광운대학교 전자계산학과 박사과정

관심분야: 객체 지향 프로그래밍 언어, 병렬 프로그래밍 언어, 객체 지향 분산 컴퓨팅

정 계 동

1985년 광운대학교 전자계산학과(학사)
1992년 광운대학교 산업대학원 전산계산학과(이학석사)
1992~현재 광운대학교 전자계산학과 박사과정
관심분야: 객체 지향 프로그래밍 언어, 데이터베이스, 병렬 프로그래밍 언어, 객체 지향 분산 컴퓨팅

최 영 근

(정보처리 논문지 제2권 제3호 pp. 299)
1980년 서울대학교 수학교육과(학사)
1982년 서울대학교 계산통계학과(이학석사)
1989년 서울대학교 계산통계학과(이학박사)
1983~현재 광운대학교 전자계산학과 교수
관심분야: 프로그래밍 언어, 병렬 프로그래밍 언어, 병렬 컴퓨터, 객체 지향 프로그래밍 언어 및 설계, 분산 처리