Journal of Digital Contents Society Vol. 9 No. 2 June. 2008(pp. 165-175)

이벤트 기반의 효율적인 클래스 식별

최미숙*, 이종석**

요 약

현재 소프트웨어 개발을 위한 방법론은 객체지향에서 컴포넌트지향으로 컴포넌트지향에서 서비스지향 발전되어 오고 있다. 컴포넌트지향 개발 방법과 서비스지향 개발 방법들은 객체지향 UML 모델을 기반으로 분석되어지므로 효율적인 객체지향 분석 방법이 필요하다. 따라서 본 논문에서는 UML 기반의 클래스 식별 및 유스케이스 모델링이 개발자의 직관과 경험에 의존하는 문제점을 보완하여 입력 데이터-처리 프로세스-출력 데이터를 사용한이벤트 기반의 유스케이스 및 클래스 분석 가이드라인과 분석 프로세스를 제안 한다.

Efficient Class Identification based on Event

Misook Choi*, Jongsuk Lee**

Abstract

Currently, software development methods have been advanced to service-oriented from component-oriented, to component-oriented from object-oriented. The component-oriented and service-oriented software development methods are analyzed by object-oriented UML model. So, the efficient analysis method for object-oriented UML model needs.

In this paper, we suggest the analysis guideline and process based on event using Input Data-Process-Output Data Table for identifying use cases and classes efficiently. And the suggested method complements the problems depending the developer's perspective and experience.

Keyword : Object-Oriented Method, Class Identification, Usecase Identification, UML Diagrams Input Data-Process-Output Data table

1. 서론

UML(Unified Modeling Language) 기반의 시스템 분석을 위하여 가장 중요하면서 기본적인 모델은 기능적 관점의 모델인 유스케이스 다이어그램(Usecase Diagram), 구조적 관점의 정적 모델인 클래스 다이어그램(Class Diagram) 그리고 행위 관점의 시퀀스 다이어그램(Sequence Diagram)이다. 그 중 도출하기 가장 어려운 다이어그램은 클래스 다이어그램이다. 클래스 다이어그램을 도출하기 위해서 기존의 소프트웨어 방법론은 요구사항을 명세한 시나리오를 분석하여 클래스를 식별한다. 시나리오 중 동사 또는 동사형을 찾아 메소드를 찾고 명사 또는 명사형을 찾아 후보 클래스를 찾은 후, 클래스를 선별하는 과정을 거친다[1]. 그러나 시나리오를 기반으로 명사나 동사를 찾아 식별하는 것은 비현실적인 방법이다. 또한, 현재 비즈니스 시스템 분석을 위해서 대부분 많이 사용하는 UML 모델링 기법[2]은 요구사항을 명세한 시나리오를 분석하여 유스케이스를 식별하고 식별된 유스케이스의 실체화 과정을 통해서 클래스와 클래스들 간의 관계를 식별하여 클래스 다이어그램을 완성한다. 그러나 클래스 다이어그램 도출을 위한 기존의 클래스 식별 방법은 유스케이스 기반의 시나리오를 중심으로 명사를 선별하여 식별하므로 개발자의 주관이나 경험에 많이 의존한다. 또한,

※ 제일저자(First Author) : 최미숙
접수일자:2008년04월25일, 심사완료:2008년05월13일
* 우석대학교 기초및자연과학연구소
khc67_kr@hanmail.net
**우석대학교 컴퓨터교육과
▣ 이 논문은 2006년도 정부(교육인적자원부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임 (KRF-2006-353-D00029)

클래스 식별을 위해서 도출해야 하는 유스케이스 모델링은 유스케이스를 식별하기 위한 명확한 지침을 제시하고 있지 않아서 추상화 레벨이 분석가 별로 다르게 기술되어지고 있으며 개발자의 경험에 많이 의존되어지고 있는 실정이다[3][4][5]. 따라서 시나리오가 완전하지 않거나, 분석가가 도메인에 익숙하지 않을 경우 클래스 식별 및 유스케이스 식별은 어렵다. 또한 분석 초기에 개발 담당자와 업무 담당자가 공통으로 이해하기 힘든 표기법과 모델을 사용하기 때문에 의사소통이 어렵다.

따라서, 본 연구에서는 유스케이스의 식별과 유스케이스 시나리오 기반의 명사를 선별하여 추출하는 클래스 식별의 어려운 문제점을 보완하고자 액터 기반의 이벤트를 중심으로 입력 데이터와 출력 데이터 그리고 처리 프로세스를 전체적으로 한 번에 파악하여 그들의 관계를 고려함으로 해서 클래스 식별과 유스케이스 식별이 용이하도록 하는 방법을 제안한다. 즉, 본 연구에서는 기존의 방법론[1][2]에서 제안하지 않았던 액터(Actor) 기반의 이벤트를 중심으로 입력, 처리 프로세스, 출력의 진행 과정을 통하여 클래스와 유스케이스를 용이하게 식별하는 방법을 제안하고 그 방법을 적용하여 효율적으로 클래스 다이어그램을 도출하는 프로세스를 제안한다. 또한 분석 초기에 개발 담당자와 업무 담당자가 공통으로 이해하기 힘든 표기법과 모델을 사용하기 때문에 의사소통이 어려운 문제점을 보완하고자 액터 기반의 이벤트를 중심으로 입력 데이터-처리 프로세스-출력 데이터를 기술한 표를 통하여 분석한다.

본 논문의 2장에서는 관련 연구를 제시하고 3장에서는 효율적인 유스케이스 식별 방법과 클래스 식별 방법을 제시하고 4장에서는 결론을 제시한다.

2. 관련연구

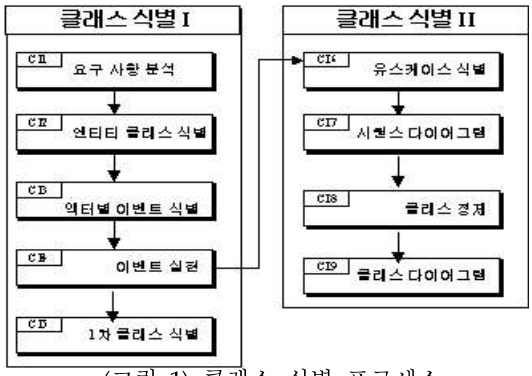
유스케이스 모델링 기법은 비즈니스 요구 사항으로부터 시스템의 경계(boundary)를 정의하고 액터와 유스케이스를 찾아 시스템의 기능별로 유스케이스당 시나리오를 작성한다. 각 시나리오를 분석하여 유스케이스 실체화 과정을 통해서 객체, 객체들의 책임과 역할을 구체화하여 객체를 추출한다. 유스케이스 실체화 과정은 객체를 경계 객체, 제어 객체, 엔티티 객체의 3가지 타입으로 구분하고 유스케이스 시나리오에 의하여 3가지 타입의 객체들 간의 상호작용을 시퀀스 다이어그램을 통해서 실현시킨다. 이러한 과정을 반복하면서 클래스를 찾아내고 그들 간의 관계를 분석하여 클래스 다이어그램을 완성한다[6][7][8]. 이러한 유스케이스 모델링은 비즈니스 요구 사항으로부터 유스케이스를 찾기도 어려울 뿐만 아니라 유스케이스 시나리오로부터 객체를 분석하는 방법이 개발자 각자의 경험에 의존하므로 내용은 같아도 모델의 표현이 다양하게 만들어지는 것이 문제점이 존재한다[4]. 또한 개발 담당자와 업무 담당자가 공통으로 이해하기 힘든 표기법과 모델을 사용하기 때문에 의사소통이 어려운 문제점이 존재한다. 따라서 UML 모델에 익숙하지 않은 개발 담당자와 업무 담당자가 경험이 없어도 이해할 수 있는 다이어그램을 사용하여 유스케이스와 클래스를 식별할 수 있는 방법이 필요하다. 또한, 개발자의 주관과 경험에 따라 유스케이스와 클래스가 다르게 식별되는 문제점을 보완할 방법이 필요하다.

3. 이벤트 기반의 효율적인 클래스 식별

본 장에서는 2장에서 제시한 기존의 방법을 보완하여 이벤트 기반의 효율적인 클래스 식별 방법을 제시한다.

3.1 이벤트기반의 클래스 식별 프로세스

본 연구에서는 다음 (그림 1)과 같은 프로세스로 클래스를 식별한다.



(그림 1) 클래스 식별 프로세스

3.2 이벤트 기반의 효율적인 클래스 식별 방법

본 절은 위의 (그림 1)의 클래스 다이어그램 도출을 위한 클래스 식별 프로세스에 의하여, 제안하는 클래스 식별 방법을 온라인 도서 주문 시스템의 사례를 통하여 제시한다.

개발자 각자의 경험과 직관에 의존하는 기존의 클래스 식별 및 유스케이스 식별 문제점을 보완하기 위해 본 연구는 액터 기반의 이벤트를 중심으로 시스템을 분석하여 유스케이스와 클래스를 식별하는 방법을 제안한다. 또한 UML 모델에 익숙하지 않은 개발 담당자와 업무 담당자가 이해할 수 있는 이벤트 기반의 입력 데이터-처리 프로세스-출력 데이터 표를 사용하여 클래스 다이어그램을 도출 한다.

3.2.1 요구 사항 분석

시스템을 분석하기 위하여 먼저 비즈니스 관점의 업무에 대한 요구사항을 분석한다. 따라서 온라인 도서 주문 시스템의 요구 사항은 다음과 같다.

고객인 회원은 인터넷을 사용하여 온라인 서점에서 도서 목록을 검색하고 원하는 도서를 장바구니에 담아 주문하는 시스템이다. 도서 주문을 원할 때 회원인 경우는 시스템에서 고객의 기본 정보를 제공하며 배송 주소의 변경을 원하는 경우 배송지를 수정하여 입력할 수 있고 입력이 완료되면 결제 방법을 선택하여 대금 결제를 한다. 대금 결제의 방법에는 신용카드, 마일리지, 무통장 입금, 인터넷 계좌이체 등의 방법이 있다. 입금이 완료되면 배송업체에 배송을 요청한다. 무통장 입금인 경우에는 관리자가 입금을 확인 후 배송업체에 요청한다. 주문 취소를 원하는 고객은 주문 상태를 확인하고 주문을 취소할 수 있다. 이와 함께 도서를 주문한 고객에게 현재의 배송 상태를 추적하는 배송 추적 시스템 서비스를 제공 한다.

3.2.2 엔티티 클래스 식별

RUP 기반의 액터(Actor) 및 비즈니스 자산(business assets)이 될 수 있는 엔티티(entity) 클래스의 유형을 본 연구에서는 네 가지로 분류한다. 첫째는 시스템을 사용하는 액터(Actor) 객체를 정의하는 클래스이고, 둘째는 액터의 요청에 응답하기 위해 기본적으로 시스템이 소유해야 할 재산 객체를 정의하는 재산 클래스이다. 셋째는 액터의 요청에 응답하기 위해서 2 개 이상의 액터나 재산 객체들의 조합에 의해서 생성되어지는 거래 객체를 정의하는 재산 클래스이고, 넷째는 액터의 요청을 처리하기 위해서 반드시 존재해야할 지원정보 객체를 정의하는 클래스이다. 따라서 이들을 찾아내고 각각의 속성을 파악하여 엔티티 클래스를 식별을 한다. 따라서 먼저 이러한 네 가지 유형의 엔티티 클래스를 요구사항 분석을 통하여 식별한다.

온라인 도서 주문 시스템의 엔티티 클래스의 유형을 요구사항 분석을 통해서 대략적으로 식별한다. 요구사항을 분석하면 다음과 같은 대략적인 엔티티 클래스가 식별된다.

액터 객체 : 회원, 관리자, 배송 시스템
재산 객체 : 도서
거래 객체 : 주문
지원정보 객체 : 결제정보, 배송정보

3.2.3 액터별 이벤트 기반의 클래스 식별

이벤트 기반의 시스템 분석은 액터가 시스템에 요청하는 서비스를 중심으로 이벤트를 식별하고 식별된 각각의 이벤트 처리를 위하여 이벤트의 입력 데이터, 이벤트 처리를 위한 프로세스, 그리고 이벤트 처리에 의한 출력 데이터의 형태로 시스템 분석을 진행한다. 따라서 이러한 실행 흐름에 의한 시스템 분석 과정은 다음 <표 1>과 같이 액터별 이벤트, 이벤트를 처리하기 위한 입력 데이터, 이벤트 처리를 위한 기능의 흐름, 이벤트 처리의 결과를 나타내는 출력의 형태를 테이블 형태로 제시한다. 따라서 UML 모

델에 익숙하지 않은 업무 담당자나 개발자라 해도 시스템을 용이하게 분석하고 이해할 수 있다. 또한 액터 기반의 이벤트 기준에 의하여 시스템 분석을 진행하므로 어렵지 않게 접근할 수 있다.

<표 1> 이벤트 기반의 입력-처리-출력 표

액터	입력		처리	출력
	이벤트	입력 자료	처리 기능	목표
회원	회원정보 조회	회원ID	로그인	회원정보조회
		비번	회원 정보 조회	

<표 1>의 분석을 통해서 1차 클래스와 그들의 관계를 식별하기 위한 가이드라인을 다음에서 제시 한다.

가이드라인 C_1. 입력 데이터에 의해서 이벤트를 처리한 후에 산출되는 출력 데이터는 대부분 클래스 단위의 정보가 출력 된다. 또한 이벤트를 실현하기 위하여 필요한 입력 데이터는 대부분 속성 단위이지만 부분적으로는 클래스 단위의 데이터가 입력될 수 있다.

가이드라인 C_2. 이벤트 처리를 위하여 임의의 클래스가 부분적으로 다른 클래스의 속성을 참조하는 관계이면 연관관계이다.

가이드라인 C_3. 이벤트 처리를 위하여 임의의 클래스가 다른 클래스 전체를 완전히 포함하여 참조한다면 그들의 관계는 포함관계이다.

가이드라인 C_4. 이벤트 처리를 위하여 입력 데이터와 출력데이터가 다르지만 처리 프로세스가 일치한다면 처리과정이 공통 속성이기 때문에 그들 간의 관계는 상속관계로 확장시킬 수 있다.

다음 <표 2>에서 가이드라인 C_1을 적용하면 대략적으로 식별된 엔티티 클래스인 회원, 관리자, 배송시스템, 도서정보, 주문정보, 결제정보, 배송정보를 포함하여 우편번호, 장바구니, 주문도서정보등의 클래스가 더 식별되어짐을 확인할 수 있다. 따라서 다음 <표 2>에서 명확히 보여주듯이 필요한 입력 데이터에 의해서 이벤트를 처리한 후에 산출되어지는 출력 데이터는 대부분 클래스 단위의 정보가 출력된다는 것을 확인할 수 있다. 또한 이벤트를 실현하기 위하여 필요한 입력 데이터는 대부분 속성 단위이지만 부분적으로는 클래스 단위의 데이터가 입력되어짐도 확인할 수 있다.

가이드라인 C_2, C_3, C_4에 의해서 이벤트 처리를 위한 입력 데이터와 출력데이터의 관계를 보면 클래스 간의 연관관계와 포함관계 그리고 상속관계가 식별되어짐을 알 수 있다.

예를 들면 가이드라인 2에 의해서 이벤트 처리를 위하여 임의의 클래스가 부분적으로 다른 클래스의 속성을 참조하는 관계이면 연관관계이다. 예를 들면 장바구니와 도서정보 클래스와의 관계를 보면 장바구니 클래스는 장바구니 담기 이벤트를 통하여 도서정보 클래스의 도서명과 가격을 참조하는 관계이다. 따라서 도서정보 클래스와 장바구니 클래스는 연관관계이다. 또한 가이드라인 3에 의해서 이벤트 처리를 위하여 임의의 클래스가 다른 클래스 전체를 완전히 포함하여 참조한다면 그들의 관계는 포함관계이다. 예를 들면 주문 정보 클래스는 배송정보, 결제정보, 주문도서 정보를 완전히 포함하여 생성되고 주문도서 정보는 장바구니 정보를 완전히 포함하여 참조하므로 그들 간의 관계는 포함관계이다. 또한 상속관계는 이벤트 처리를 위하여 입력 데이터와 출력데이터가 다르지만 처리 프로세스가 일치한다면 처리과정이 공통 속성이기 때문에 그들 간의 관계는 상속관계로 확장시킬 수 있다.

따라서 본 연구에서 제안한 클래스 간의 관계 식별은 이벤트 처리를 위한 입력 데이터와 출력 데이터와의 관계 또는 처리 프로세스의 일치 등을 분석해 보면 클래스 간의 관계를 용이하게 식별할 수 있다.

본 논문에서 제시한 1차 클래스 식별 가이드라인 1,2,3,4를 적용하기 위하여 다음 <표 2>는 온라인 도서 주문 시스템에 대하여 액터별 이벤트들 및 각 이벤트의 입력-처리-출력을 통한 실현의 과정을 제시한다.

<표 2> 클래스 식별을 위한 이벤트 기반의 입력-처리 프로세스-출력 표

액터	입력		처리	출력
	이벤트	입력자료	기술	목표
회원	회원정보 조회	회원ID 비번	**로그인** 회원 정보 조회	회원정보조회
회원	기등록 회원체크	이름 주번	기등록 회원체크	회원(T/F)
회원	중복아이 디체크	회원ID	중복아이 디체크	회원ID(T/F)
회원	회원정보 저장	회원정보	회원정보 저장	회원정보저장
회원	회원가입	회원정보	기등록회원체크 중복아이 디체크 **우편번호 검색** 회원정보 저장	회원정보저장
회원	회원정보 변경	회원ID 비번	**로그인** 회원 정보 조회 **우편번호 검색** 회원정보 저장	회원정보저장
회원, 직원	회원탈퇴	회원ID 비번	**로그인** 회원 정보 조회 회원정보 삭제	회원정보삭제
회원	우편번호 검색	동	**우편번호 검색**	우편번호
회원	로그인	회원ID 비번	**로그인**	로그인
회원	장바구니 담기	도서명 수량 가격	도서목록 조회 장바구니 정보 저장	**장바구니정보** 저장(도서명,**수량,,가격,총액, 총구매가격**) + **주문도서정보**
회원	장바구니 조회	회원ID 비번	로그인 장바구니 정보 조회	장바구니정보 조회
회원	장바구니 변경	회원ID 비번	로그인 장바구니 조회 장바구니 변경 장바구니 저장	장바구니정보 변경
회원	장바구니 비우기	회원ID 비번	로그인 장바구니 조회 장바구니 삭제 장바구니 저장	장바구니정보 삭제
회원	도서목록 조회	도서명, 핵심단어, 저자, 출판사, ISBN	도서목록 조회	도서목록정보 출력
회원	도서상세 정보조회	도서명	도서상세 목록 조회	도서정보 출력 (도서목록조회에 의한 정보에 의존)
회원	도서주문	회원ID, 비번	장바구니 조회	장바구니정보 출력
		회원ID, 비번	회원정보 조회	회원정보출력
		배송정보	배송정보 저장	배송정보저장
		결제정보	결제정보 입력	결제정보저장
			결제요청/ 처리	
			결제확인	결제처리(T/F)
		장바구니 정보	도서정보 저장	도서정보저장
		배송정보	배송요청 (배송시스템)	배송요청번호 출력
		회원정보_ 이름, 전화번호 도서정보_ 번호	주문정보 저장	주문번호출력

		배송정보_번호		
		배송요청_번호		
		결제정보_번호		
직원	결제확인	주문정보	미결제주문정보조회	미결제주문정보
		장바구니정보	도서정보저장	도서정보저장
		배송정보	배송요청(배송시스템)	배송요청번호출력
		회원정보_이름, 전화번호	주문정보저장	주문번호출력
		도서정보_번호		
		배송정보_번호		
		배송요청_번호		
		결제정보_번호		
회원	주문조회	회원ID	로그인	주문정보
		비번	주문정보조회	
회원	주문취소	회원ID	로그인	주문정보
			주문조회	
		비번	주문정보삭제	
		배송정보번호	배송취소	배송시스템

따라서 본 연구의 3.2.2절에서 제시된 방법에 의하여 식별된 클래스들은 회원정보, 관리자정보, 배송 시스템, 도서정보, 주문정보, 결제정보, 배송정보 클래스이고 본 절에서 식별된 클래스들은 회원정보, 도서정보, 주문정보, 배송정보, 결제정보, 장바구니정보, 주문도서정보, 우편번호, 로그인이다. 따라서 3.2.2절에서 식별되지 않은 클래스들은 본 절에서 더욱 상세히 클래스들이 도출되어짐을 확인할 수 있다. 또한 3.2.2절에서 식별된 액터들이 이벤트 처리 프로세스에 사용되어지지 않을 경우 이벤트 기반의 식별 방법에 의해서는 클래스로서 식별되지 않음을 알 수 있다. 따라서 3.2.2 절에서 제안한 방법과 본 절에서 제안한 방법에 의해서 식별된 클래스를 다시 정제하면 회원정보, 도서정보, 주문정보, 배송정보, 결제정보, 장바구니정보, 주문도서정보, 우편번호, 로그인, 배송시스템 클래스가 식별된다.

본 연구에서 제안한 1차 클래스 후보에 대한 식별 방법은 액터 기반의 이벤트를 중심으로 입력 데이터-처리 프로세스-출력 데이터를 중심으로 분석하므로 클래스 식별이 좀 더 효율적이다. 또한 기존의 클래스 식별 방법에서 제시하지 않았던 방법으로, 비즈니스 도메인에서 이벤트 처리에 의한 출력 데이터는 대부분 클래스 단위라는 것을 본 연구에서 제시함으로 해서 기존의 클래스 식별의 어려움을 보완하였다. 또한 식별된 1차 클래스에 대해서 참조 데이터의 특성을 분석하므로 좀 더 용이하게 클래스 간의 관계를 도출할 수 있다.

3.2.4 유스케이스 식별

본 절은 액터 기반의 이벤트를 중심으로 입력 데이터, 처리 프로세스 그리고 출력 데이터와의 관계를 통하여 효율적으로 유스케이스를 식별하는 가이드라인을 제시한다.

> 가이드라인 UI_1. 각각의 이벤트가 같은 데이터를 가지고 처리하지만 처리 프로세스가 각각 다르다면 각각의 이벤트는 각각의 유스케이스로 식별한다.

> 가이드라인 UI_2. 각각의 다른 이벤트들이 하나의 이벤트에 포함되어 처리되어질 경우 이 때 각각의 이벤트를 포함한 이벤트를 하나의 유스케이스로 식별한다.

> 가이드라인UI_3. 임의의 이벤트가 다른 이벤트의 출력 데이터에 의존해서 기능이 실현되어질 경우 그들을 합하여 하나의 유스케이스로 식별한다.

> 가이드라인UI_4. 가이드라인UI_1에 의해서

식별된 각각의 이벤트에 대한 유스케이스가 부분적으로 똑같은 프로세스의 흐름을 갖는다면 그들은 그룹화 되어 포함 유스케이스나 선택 유스케이스로 식별한다. 유스케이스가 반드시 포함해야할 프로세스 흐름이라면 포함 유스케이스이고 선택적으로 실행해야할 유스케이스라면 선택 유스케이스로 식별한다.

가이드라인UI_5. 각각의 이벤트에 대한 유스케이스들이 데이터는 다르지만 공통으로 참조하는 이벤트가 존재한다면 공통 유스케이스로 식별한다.

가이드라인UI_6. 각각의 이벤트가 처리 프로세스가 같지만 입력 데이터와 출력 데이터가 다를 경우 각각의 이벤트는 각각의 유스케이스로 식별한다.

다음 <표 3>은 유스케이스 식별 가이드라인에 의한 유스케이스 식별 결과를 제시하고 있다.

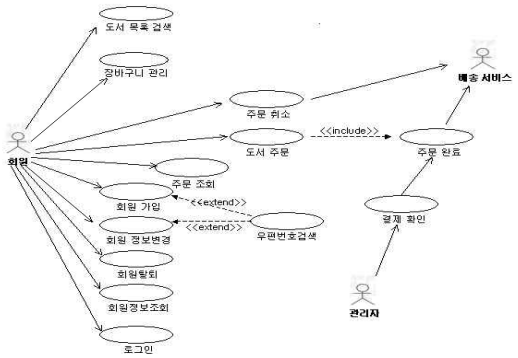
<표 3> 유스케이스 식별을 위한 이벤트 기반의 입력-처리 프로세스-출력 표

유스케이스	액터	입력					처리			출력	
		이벤트	이벤트 정의	입력자료	입력 속성 정의	입력 데이터 정의	기술	세부 정의	처리 정의	목표	출력 정의
U2	회원	**회원정보 조회**	E1	회원ID	C1_A1	D1	**로그인**	S1	P1	회원정보조회	C1
				비번	C1_A2		회원 정보 조회	S2	P2		
U3	회원	기등록 회원체크	E2	이름	C1_A3	D2	기등록 회원체크	S3	P3	회원(T/F)	C1
				주번	C1_A4						
U3	회원	중복아이디체크	E3	회원ID	C1_A1	D3	중복아이디체크	S4	P4	회원ID(T/F)	C1
U3	회원	회원정보저장	E4	회원정보	C1	C1	회원정보저장	S5	P5	회원정보저장	C1
U3	회원	**회원가입**	E5	회원정보	C1	C1	기등록회원체크	S3	P3	회원정보저장	C1
							중복아이디체크	S4	P4		
							우편번호검색	S0	P0		
							회원정보저장	S5	P5		
U4	회원	**회원정보변경**	E6	회원ID	C1_A1	D1	**로그인**	S1	P1	회원정보저장	C1
							회원 정보 조회	S2	P2		
				비번	C1_A2		**우편번호검색**	S0	P0		
							회원정보저장	S5	P5		
U5	회원, 직원	**회원탈퇴**	E7	회원ID	C1_A1	D1	**로그인**	S1	P1	회원정보삭제	C1
							회원 정보 조회	S2	P2		
				비번	C1_A2		회원정보 삭제	S6	P6		
U0	회원	**우편번호 검색**	E8	동	C1_A5	D0	**우편번호검색**	S0	P0	우편번호	C0
U1	회원	**로그인**	E9	회원ID	C1_A1	D1	**로그인**	S1	P1	로그인	C1
				비번	C1_A2						
U6	회원	**장바구니담기**	E10	도서명	C3_A1	D3	도서목록 조회	S11	P11	**장바구니정보**	C2

				수량	C2_A1		장바구니 정보 저장	S7	P7	저장(도서명, **수량,,가격,총액, 총구매가격**) + **도서정보**	**(참조 C3)**
				가격	C2_A2						
U7	회원	**장바구니조회**	**E11**	회원ID	C1_A1	**D1**	로그인	S1	P1	장바구니정보 조회	**C2**
				비번	C1_A2		장바구니정보 조회	S8	P8		
U8	회원	**장바구니변경**	**E12**	회원ID	C1_A1	**D1**	로그인	S1	P1	장바구니정보 변경	**C2**
				비번	C1_A2		장바구니조회	S8	P8		
							장바구니 변경	S9	P9		
							장바구니 저장	S7	P7		
U9	회원	**장바구니 비우기**	**E13**	회원ID	C1_A1	**D1**	로그인	S1	P1	장바구니정보 삭제	**C2**
				비번	C1_A2		장바구니조회	S8	P8		
							장바구니 삭제	S10	P10		
							장바구니 저장	S7	P7		
U10	회원	**도서목록조회**	**E14**	도서명,	C3_A1	**D4**	도서목록조회	S11	P11	도서목록정보 출력	**C3**
				핵심단어,	C3_A2						
				저자,	C3_A3						
				출판사,	C3_A4						
				ISBN	C3_A5						
U10	회원	도서상세정보조회	**E15**	도서명	C3_A1	**D5**	도서상세목록 조회	S12	P12	도서정보 출력 (도서목록조회에 의한 정보에 의존)	**C3**
U11	회원	**도서주문**	**E16**	**D1**	C1_A1 C1_A2	**C1**	장바구니조회	S8	P8	장바구니정보 출력	**C2**
				D1	C1_A1 C1_A2	**C1**	회원정보조회	S2	P2	회원정보출력	**C1**
				배송정보	**C4**	**C4**	배송정보저장	S13	P13	배송정보저장	**C4**
				결제정보	**C5**	**C5**	결제정보입력	S14	P14	결제정보저장	**C5_A1**
							결제요청/처리	S15	P15		
							결제확인	S16	P16	결제처리(T/F)	**C5**
				장바구니 정보	**C2**	**C2**	도서정보저장	S17	P17	도서정보저장	**C6_A1**
				배송정보	**C4**	**C4**	배송요청 (배송시스템)	S18	P18	배송요청번호 출력	**C7_A1**
				회원정보_ 이름, 전화번호	C1_A3 C1_A4	**C1**	주문정보저장	S19	P19	주문번호출력	**C8_A1**
				도서정보_ 번호	C6_A1	**C6**					
				배송정보_	C4_A1	**C4**					

				번호							
				배송요청_번호	C7_A1	C7					
				결제정보_번호	C5_A1	C5					
U12	직원	**결제확인**	E17	주문정보	C8	C8	미결제주문정보조회	S20	P20	미결제주문정보	C8
				장바구니정보	C2	C2	도서정보저장	S17	P17	도서정보저장	C7_A1
				배송정보	C4	C4	배송요청(배송시스템)	S18	P18	배송요청번호출력	C7_A1
				회원정보_이름,전화번호	C1_A3 C1_A4	C1	주문정보저장	S19	P19	주문번호출력	C8_A1
				도서정보_번호	C6_A1	C6					
				배송정보_번호	C4_A1	C4					
				배송요청_번호	C7_A1	C7					
				결제정보_번호	C5_A1	C5					
U13	회원	**주문 조회**	E17	회원ID	C1_A1	D1	로그인	S1	P1	주문정보	C8
				비번	C1_A2		주문정보조회	S20	P20		
U14	회원	**주문 취소**	E18	회원ID	C1_A1	D1	로그인	S1	P1	주문정보	C8
				비번	C1_A2		주문조회	S20	P20		
							주문정보삭제	S21	P21		
				배송정보번호	C4_A1	C4	배송취소	S22	P22	배송시스템	
U15	회원	**주문완료**	**U11과 U12에 공통인 내부 유스케이스**								

본 연구의 유스케이스 식별 가이드라인에 의하여 식별된 유스케이스 결과를 <표 3>에 제시해 놓았다. 가이드라인UI_2에 의하여 식별된 유스케이스는 U3에 해당하고 가이드라인UI_3에 의하여 식별된 유스케이스는 U10에 해당한다. 가이드라인UI_4에 의하여 식별된 유스케이스는 U15에 해당하고 가이드라인UI_5에 의하여 식별된 유스케이스는 U0에 해당한다. 그 다음 가이드라인UI_6에 의하여 식별된 유스케이스는 본 연구의 예에서는 제시되지 않았고 그 외 나머지 유스케이스는 가이드라인UI_1에 의한 것이다. 따라서, 다음 (그림 2)는 본 연구의 유스케이스 식별 가이드라인에 의하여 산출된 유스케이스 다이어그램이다.



(그림 2) 식별된 유스케이스 다이어그램

본 연구에서는 다른 방법론에서 제안하지 않았던 액터 기반의 이벤트를 중심으로 입력 데이터, 처리 프로세스, 출력 데이터를 고려하여 유스케이스 식별을 위한 명확한 가이드라인을 제시하므로, 추상화 레벨이 분석가 별로 다르게 기술되고 개발자 경험에 많이 의존되고 있는 유스케이스 모델링의 문제점을 보완할 수 있다.

3.2.5 시퀀스다이어그램(Sequence Diagram)을 통한 클래스 정제

시스템 분석을 위하여 UML 모델링 방법을 적용할 경우, 가장 큰 어려움은 유스케이스 식별과 클래스 식별에 있다. 그러나 본 연구에서 제안한 클래스 식별 방법과 유스케이스 식별 방법에 의하여 효율적으로 시스템이 분석되므로 그 이후에 과정은 UML 모델링 방법을 용이하게 적용할 수 있다. 따라서 유스케이스의 시나리오를 기반한 시퀀스 다이어그램을 통해서 클래스의 상세한 부분, 즉, 클래스의 속성, 클래스의 메소드 그리고 메소드 호출에 의한 클래스와의 상세한 관계를 식별하고 클래스 다이어그램을 정제해 가는 과정을 통해서 완전한 클래스 다이어그램을 산출한다.

4. 결론

본 연구에서는 액터 관점에서 이벤트 기반의 클래스 식별, 클래스 간의 관계 그리고 유스케이스 식별 방법을 개발 담당자와 업무 담당자가 친숙하게 의사소통할 수 있는 입력, 처리, 출력에 관한 표를 통해서 제안하였다. 기존의 UML 모델링 기법은 유스케이스 식별 기준이나 클래스 식별 기준이 명확히 제시되지 않아 개발자 각자의 경험에 의존하는 문제점과 개발 담당자와 업무 담당자가 공통으로 이해하기 힘든 표기법과 모델을 사용하기 때문에 의사소통이 어려운 점이 존재하였다.

본 연구에서는 이러한 문제점을 해소하여 분석 초기에 UML 모델에 익숙하지 않은 개발 담당자와 업무 담당자가 경험이 없어도 이해할 수 있는 다이어그램을 사용하고, 명확하고 이해하기 쉬운 클래스와 유스케이스 식별 방법 제시에 의하여 유스케이스와 클래스 다이어그램을 효율적으로 도출할 수 있다.

참 고 문 헌

[1] Martin Fowler, Kendall, "UML DISTILLED", 홍릉과학출판사, 1999.

[2] Hans-Erik Eriksson and Magnus Penker, UML Toolkit, Wiley, 1998.

[3] Susan Lilly, "Use Case Pitfalls: Top 10 Problems from Real Projects Using Use Cases," Proceedings of Tools USA, 1999.

[4] Hermann Kaindl, "Difficulties in the Transition from OO Analysis to Design," IEEE Software, 1999.

[5] 한/카네기멜론대학 기술교류협회, 최신 소프트웨어 공학기법, V.I.LAND Co. Ltd, 2002.

[6] Frank Armour, Granville Miller, Advanced Use Case Modeling Software Systems, Addison Wesley, 2002.

[7] Pan-Wei Ng, "Business Process Modeling and Simulation with UML," Part I: Defining a UML Transition Model That Maps to RUP Business Models, the Rational Edge, http://www.therationaledge.com/content/apr_02/t_businessProcessModeling_pn.jsp, 2002.

[8] Ivar Jacobson, Maria Ercisson, Agneta Jacobson, The Object Advantage: Business Process Reengineering With Object Technology, Addidon Wesley, 1995.

최 미 숙

1990년 : 전북대학교 수학과 졸업 (이학사)
1994년 : 숙명여자대학교 컴퓨터과학과 석사과정 졸업 (이학석사)
2002년 : 숙명여자대학교 컴퓨터과학과 박사과정 졸업 (이학박사)
1995년～1999년 : 나주대학 소프트웨어 개발과 전임교수
2004년～2006년 : 우석대학교 컴퓨터공학과 초빙교수
2006년～현 재 : 우석대학교 기초및자연과학연구소 연구교수
관심분야 : 소프트웨어 개발 방법론, 컴포넌트 기반 시스템, 소프트웨어 메트릭, SOA

이 종 석

1988년 : 서울대학교 계산통계학과 학사
1990년 : 서울대학교 계산통계학과 석사
2001년 : 서울대학교 컴퓨터공학 전공 박사
1993년～현 재 : 우석대학교 컴퓨터교육과 교수
관심분야 : 소프트웨어 공학, 소프트웨어 복잡도, 컴포넌트 기반 시스템, SOA