Journal of Korea Multimedia Society Vol. 11, No. 4, April 2008(pp.525-543)

메타모델 기반 사용자 인터페이스 계층적 모델링 프로세스

송치양[†], 조은숙[††], 김철진[†††]

요 약

최근 들어 소프트웨어 개발에 있어서 사용자 인터페이스가 차지하는 비중이 급증하고 있다. 이로 인해 스윙, MFC, Web 2.0 등과 같은 다양한 사용자 인터페이스 개발 관련 기술들이 소개되고 있다. 그러나 현재 대부분의 소프트웨어 개발에 있어서는 사용자 인터페이스 부분과 비즈니스 부분을 별도로 개발하는 프로세스로 진행되고 있다. 이로 인해 통합 과정에서의 어려움, 개발 기간의 연장, 개발 모델의 재사용성의 저하 등과 같은 문제점들이 발생하고 있다. 즉, UI 모델링이 체계적이고 계층적이지 못하며, UI 모델링과 비즈니스 모델링간의 일관성있는 통합 기법을 제공치 않아, 구축 모델의 확장성과 재사용성의 저하를 초래하고 있다. 본 논문은 이를 해결하기 위해 개발 단계의 추상화 수준에 따른 계층적 메타모델을 사용해서 단일화되고 체계적인 UML 기반의 사용자 인터페이스 모델링 프로세스를 제시한다. 이를 위해, 개발의 성숙도에 준거하여 PIM/PSM으로 UI 모델의 모델링 요소를 계층화하여 메타모델을 제시한다. 3 단계 모델링(개념/ 명세/ 상세)에 의해 UI 및 비즈니스 메타모델을 적용해서 UI 모델링과 비즈니스 모델링이 통합된 계층적인 모델링 프로세스를 정립한다. 제시한 프로세스를 인터넷 쇼핑몰에 적용해 봄으로써 실효성을 제시한다. 실험 결과를 통해 계층적 UI 메타모델 및 프로세스가 체계적이고 계층적인 UI 모델을 구축할 수 있다. 이는 모델의 품질과 재사용성을 향상시킬 수 있었다.

An User Interface hierarchical modeling process based on Metamodel

Chee-Yang Song[†], Eun-Sook Cho[††], Chul-Jin Kim[†††]

ABSTRACT

Recently, the scope of user interface is increasing the relative importance in software development dramatically. As a result, there are various relative technologies like as SWING, MFC, Web 2.0, and etc. However, most current software developments are progressed in separate development process with user interface part and business part respectively. This causes the problems, like as a difficulty in the integration process, an development period's delay, and a poor reusability for the constructed models. That is, the extendability and reusability of the created models is being decreased because UI modeling is not systematic and hierarchical, and the consistent integration technique between UI modeling and business modeling does not supported. To solve these problems, this paper proposes an unified and systematic UI modeling process based on UML, using the hierarchical metamodel according to the abstraction levels of development phase. We suggest an UI metamodel, which contains a hierarchy by layering the modeling elements in PIM and PSM based on maturity degree of the development. An hierarchical modeling process combined UI modeling and business modeling is built by applying the UI and business metamodel in terms of three modeling phases(concept/ specification/ concrete). The effectiveness of the modeling process is shown by applying the proposed process into an Internet Shopping Mall System. Through the exploratory results, the hierarchical UI metamodel and process can produce systematic and layered UI models. This can improve the quality and reusability of models.

Key words: User Interface(사용자 인터페이스), Hierarchical modeling(계층적 모델링), UML(유엠엘), Metamodel(메타모델)

※ 교신저자(Corresponding Author) : 조은숙, 주소 : 서울시 중랑구 면목동 49-3 (131-702), 전화 : 02)490-7562, FAX : 02)490-7396, E-mail : cscho@scoil.ac.kr
접수일 : 2007년 9월 13일, 완료일 : 2008년 2월 13일
[†] 정회원, 경북대학교 소프트웨어공학과 조교수
(E-mail : cysong@knu.ac.kr)
[††] 정회원, 서일대학 소프트웨어과 조교수
[†††] 정회원, 삼성전자 디지털 솔루션 센터
(E-mail : cjkim777@gmail.com)

1. 서 론

인터넷의 초고속화에 힘입어 이질적 산업간, 미디어간 혹은 서비스들간의 융합화로 이러한 비즈니스의 응용 서비스들을 제공하기 위한 소프트웨어는 점차 복잡화, 대형화 및 다양화되어 가고 있다. 이에, 사용이 편리하고 고급스러운 사용자 인터페이스(UI : User Interface)를 추구함으로서 UI가 소프트웨어 개발의 핵심 이슈로 부각되고 있다. 또한, UI의 수요는 증가되고, 복잡한 디자인과 우수한 품질을 요구하는 실정이다. 이를 위한, SWING, MFC 및 Web 2.0 등과 같은 UI를 지원하기 위한 관련된 기술들이 현존하고 있다.

UI를 개발하기 위해서 UI 개발방법이 필요하고, 정립된 개발절차에 따라 다양한 응용분야(가령, 웹 응용, 모바일 응용, 유비쿼터스 응용 등)에 적용할 수 있는 것이다. UI는 정보를 사용자에게 제공하거나 혹은 사용자로부터의 지시를 받는 소프트웨어의 일부로 정의한다[1]. 즉, 애플리케이션의 개발은 체계적인 UI 모델링 방법 그리고 UI 모델링과 비즈니스 모델링이 통합된 개발방법이 필요하다.

UML(Unified Modeling Language)[2] 기반으로 비즈니스 모델링 및 데이터 모델링을 수행하는 개발방법론들에 대한 연구는 체계화되어 있고 구체적이고 계층적인 프로세스를 갖고 있다[3]. 반면, UI 모델링에 대한 체계적인 방법은 아직 연구가 미약하다. 왜냐하면, UML은 UI 모델링에 대한 표기법을 제공하지 않기 때문이다. 그래서, 기존의 UI 모델링 프로세스에 대한 연구는 UML의 확장 프로파일(profile)을 통한 UI 접근[4] 혹은 UI가 다른 응용분야보다 중요시되는 웹 응용의 개발방법론[5-7]에서 모델링 가이드를 제시하고 있다. 그러나, 이들은 초기 UI 아키텍쳐 구성에서 상세 UI widget(위젯)에 이르는 UI 모델링의 전체 개발단계를 아우르는 조직적이고 구체적인 일원화된 개발 방법론을 지원하지 못하고 있다. 또한, 개발단계의 추상화 수준에 따라 UI를 모델링 할 수 있는 계층적 모델링을 명확하게 제공치 않아 개발 접근이 용이치 못하고 추상화 수준별 구축된 UI 모델의 재사용성을 저하시킨다. 더우기, 제시하는 방법별에 따라 UI 모델의 구조물 및 개발 프로세스가 상이하여 사용에 어려움을 수반한다. 특히, 소프트웨어 시스템을 모델링하기 위해서는 UI 모델링, 비즈니스 모델링(DB 모델링 포함)이 상호 결합, 조직화되어 소프트웨어 모델링 작업이 수행되어야 한다. 그러나, 기존 연구는 UI 모델링과 비즈니스 모델링의 작업이 각각 분리된 개발 프로세스로 되어 있다. 이로서, 통합 모델링시에 발생하는 어려움, 생성 모델들간의 일관성 보장이 어렵고, 구축된 개발 모델의 재사용성의 저하 및 개발 기간의 연장 등과 같은 문제점들이 발생하고 있다[8]. 결국, UI 모델링이 체계적이고, 계층적이지 못하며, UI 모델링과 비즈니스 모델링간의 일원화된 통합 모델링 프로세스를 제공치 않으므로서, 구축되는 생성 모델의 확장성과 재사용성의 저하를 초래하게 되고, 나아가, UI의 품질 및 수요 요구를 충족시키지 못하고 있다.

본 논문의 목적은 UML 기반의 일원화된 계층적 UI 모델링 프로세스와 UI-비즈니스 통합 모델링 프로세스를 제공하는 것이다. 즉, 개발 단계별로 요구되는 UI 모델에 대한 UI 메타모델(metamodel)을 정의하고, UI 모델링 프로세스를 정립하여 비즈니스 개발 프로세스와 통합하여 메타모델 기반의 계층적 모델링 프로세스를 제시하는 것이다. 개발 단계는 추상화의 성숙도에 따라 개념적/ 명세적/ 상세적의 3개 모델링 단계로 구성한다. 계층적 UI 메타모델은 MDA(Model-Driven Architecture)[9] 방식에 의해 PIM(Platform Independent Model) 개념에 입각한 명세적 단계 지원의 Specific UI Metamodel 그리고 PSM(Platform Specific Model) 방식에 기반한 상세적 단계 지원의 Concrete UI metamodel(자바 플랫폼 기반)로 정의한다. 이것은 추상화 수준에 따라 명세적 단계와 상세적 단계로 UI 모델링 요소를 계층화하여 구축되어 진다. 또한, 이 두 개 PIM-PSM UI 메타모델간의 변환 프로파일을 정의한다. 제시한 UI 메타모델과 [3]에서 제시된 비즈니스 메타모델에 기반하여, 개발단계에 따라서 비즈니스 모델(제어 클래스: Controller와 DB 클래스: Model)과 UI 모델(UI 클래스: View)이 분할과 결합으로 구성된 통합 모델링 프로세스를 정립한다. 비즈니스 모델과 UI 모델간의 분할은 MVC(Model View Controller) 모델을 적용한다. 사례 연구로는 범용적 사용의 인터넷 쇼핑몰 시스템의 애플리케이션을 대상으로 제시 개발 프로세스를 적용한다.

본 논문의 구성은 II장에서 UI 메타모델, 개발 프로세스 그리고 비즈니스-UI간 통합 모델링 프로세

스에 대한 현황 문제점을 분석한다. III장에서는 UI 모델링의 접근방향과 UI 모델을 계층적 메타모델로 정의하고 PIM UI 메타모델과 PSM UI 메타모델간의 변환 기법을 다루며, IV장에서는 III장의 계층적 UI 메타모델과 비즈니스 메타모델에 기반하여 3 계층의 모델링 단계에 의한 UI-비지니스 통합 계층적 모델링 프로세스를 기술한다. V장에서는 실제 적용 사례로서 인터넷 쇼핑몰 응용시스템에 대한 UI 중심의 모델링을 보인다. VI장에서는 UI 모델, UI 모델링 프로세스 및 메타모델 측면에서 기존 연구와 비교 평가를 기술한다.

2. 관련 연구

일반적으로 UI 모델링에 관한 기법 및 절차를 소프트웨어 개발방법론내의 일부분으로 포함하여 제시하고 있는데, 본 연구는 메타모델 기반의 UI 모델링 프로세스와 UI-비즈니스 통합 모델링 프로세스에 초점을 두고 관련된 현존 연구를 분석한다.

2.1 계층적 UI 모델링 프로세스

UI 모델링을 하기 위해서, 개발 단계별로 수행해야 할 UI 모델과 이 모델에 대한 모델링 요소를 정의한 UI 메타모델이 있어야 하고, 이에 기반한 개발 프로세스가 필요하다. UI 모델링은 그간 주로 웹 애플리케이션 개발 분야에서 연구되어 왔다. 초기의 화면 설계 기법으로 하이퍼미디어 애플리케이션 개발방법(프리젠테이션 모델을 제시)인 Hydev[10]가 있으며, 또한 항해 중심과 객체지향의 개념에 입각한 디자인 방법으로 OOHDM(Object -Oriented Hypermedia Design Methods)[11]이 있다.

대표적인 웹 개발방법론인 UWE(UML-based Web Engineering)[5,6,8,12]는 웹 특성인 항해에 초점을 두고, 초기 도메인 분석에서 상세한 프리젠테이션 모델을 생성하기까지의 전체 모델링 과정을 제시하고 있다. UI를 포함해서 웹 응용 개발에 필요한 메타모델[6]을 정의하고, UML을 사용토록 UWE 프로파일화 하였다. 개발 프로세스는 초기 UI의 스케치(sketch)와 스토리보딩(storyboarding)을 통한 모델링 과정[5]에서 점차 세분화되어 분석 단계에서 개념 모델과 프로세스 모델을 작성하고, 설계 단계에서 항해 모델과 프리젠테이션 모델을 설계하는 프로세스로 진화하였다[7]. 그러나, UI 모델링 요소에 의한 계층적 모델링이 미약하며, UI의 모델링 요소가 다소 추상적이고 범용이지 않아 실용적인 적용이 어렵다.

한편, 비 웹용 애플리케이션을 위한 UI 모델링[4, 13]은 UML을 확장한 UI 프로파일을 통해 UI 모델과 그 모델링 요소를 정의하고 있다. UMLi[4]는 모델 기반의 UI 프리젠테이션 모델을 구축하기 위해 추상적 모델과 상세적 모델로 모델링 요소를 계층화하여 제시하고 활동 모델을 이용해서 프리젠테이션 모델(presentation model)을 구축한다. 그러나, UI 요소간의 상호 협력 프로세스에 초점을 두고 있어, 개발 환경의 구체적인 UI 플랫폼(platform)에 대한 UI 모델 및 모델링 요소를 제시하고 있지 않다.

GUI_Layout[1]은 좀더 UI 디자인의 관점이 강조된 체계적인 UI 모델링을 제시한다. 이것은 표 1과 같이, GUI 모델링의 메타모델과 프로세스를 정의하고 있다. UI 모델링을 위해 스크린(screen)과 screen area로 구성된 계층적 GUI 레이아웃 모델과 프리젠테이션 모델을 제시하고 있다. 그러나, 상세한 UI 설계를 위해 구현하는 UI 플랫폼에 기반한 UI 모델, 모델링 요소 그리고 상세한 모델링 지침이 없으며, 계층적 모델링을 제공하지 않는다.

한편, 구축된 모델의 재사용성 향상을 위해서, MDD(Model Driven Development) 기반의 UI 개발 방법은 구현 환경에 독립적인 모델인 PIM과 구현 플랫폼 환경을 고려한 다수의 PSM 모델을 작성할 수 있도록 제공해야 한다. [14]에서 MB-UID(Model Based User Interface Design)의 계층적 개발 방법은 PIM 모델로 context model/ Application model/ presentation model을 작성하고, PSM 모델로 UI Deployment Model/ Presentation model의 디자인 방법을 제시한다. 또한, [15]에서는 개략 PIM 모델에서 상세 PSM 모델에 이르는 UI 디자인 방법과 모델간 변환방법을 제시한다. 생성되는 모델은 information model, abstract user interface model, view composition model 그리고 concrete user interface model이다. 그러나, 이들은 범용적인 UI 모델링 요소와 거리가 있으며, 목표 플랫폼에 부합한 구체적인 UI 모델간의 변환방법이 미약하다.

2.2 UI-비지니스 통합 모델링 프로세스

애플리케이션의 개발은 추상화 수준에 따라 MVC

표 1. GUI-Layout(1)의 모델링 체계

개발 모델 및 절차	모델링 방법	모델링 요소
1. Abstract GUI Layout Model	ㅇ연관된 유스케이스(use case)들을사용, 작성 (screen area와 유스케이스 연결) ㅇ개략적 레이아웃 구성도 ㅇ클래스도/ 복합구조도 이용	- screen, screenarea - navigation, content, container screenarea functional screenarea - UI Functionality : activated, static - form, navigation, link, workspace, text, - heading, image, logo, point, dimension
2. Screen Model	ㅇ추상화된 스크린 모델	- (Abstract) screenarea
3. Screenarea Model	ㅇ스크린영역 명확화(상속)	- (concrete) screenarea
4. Activity Model	ㅇ스크린과 screen area간 연계	- backward/presentation/user partition
5. Navigation Model	ㅇ스크린간의 흐름 및 링크 참조(link reference) 연계 표현	- reference, Linkreference, - screenflow
6. Sitemap Model	ㅇ항해도와 linkreference 사용한 순차적 스크린 구성	- storyboard
7. Storyboard Model	ㅇ전체사이트 구조와 스크린 흐름 기반의 링크화	- sitemap
8. Concrete GUI Layout Model	ㅇ윈도우로 구성되는 상세적 레이아웃 구성도	- workspace, app. 및 floating window

에 근간하여 UI 모델과 비즈니스 모델이 상호 분리와 통합을 통해 모델링 되어야 모델의 재사용성을 높일 수 있다. 현존 연구에서는 UML을 사용하여 비즈니스 모델링을 수행하고 UML을 확장하여 UI 모델링을 수행한다. [7]은 기존의 OO-H와 UWE를 통합한 프로세스를 가진다. [7]에서의 모델링 과정은 개념 모델, 프로세스 모델, 항해 모델 및 프리젠테이션 모델로 이루어진다. 프로세스 모델을 통해 비즈니스를 모델링하고 항해 모델을 통해 UI를 모델링하는데, 이때 프로세스 모델과 항해 모델이 결합되어 모델링이 수행된다. 이렇게, 통합된 항해 모델은 프리젠테이션 모델로 구체화 된다. 또 다른 통합 연구로 [16]에서, 요구사항 정의 단계에서 화면 흐름 모델을 작성하고, 분석단계에서 경계(UI) 클래스, 제어(비지니스 로직: business logic) 클래스와 DB(엔터티: entity) 클래스로 객체 모델과 실현 모델을 작성하고, 설계 단계에서 개발 플랫폼과 구현 언어 기반의 객체 모델을 작성한다. 이 외, MVC 모델[17]을 이용한 통합에 대한 연구가 있다. 그러나, 이들은 비즈니스와 UI와의 통합 모델링에 있어서, 그 통합 모델링 프로세스가 체계적으로 구체화되어 있지 않으며, 각각 별도의 개발 프로세스를 가지고 있으며, 자체 UML 만으로 UI 모델링을 다루고 있기 때문에 충분한 UI 모델링 표현에 한계를 갖고 있다.

지금까지 살펴본 UI 관련 메타모델, UI 개발 프로세스 그리고 UI와 비즈니스간의 연계 모델링 방법이 소프트웨어 개발 방법론[18,19]에 포함되어 제공하게 된다. UI-RUP[19]는 유스케이스 명세서(use case description)의 흐름(주요/ 서브/ 대안/ 예외)에 기반하여 UI 요소들과의 메핑을 통해 RUP내에 UI 모델링 프로세스를 제공하나, UI 메타모델 및 그 프로세스의 세부적 지침이 미약하여 명확한 UI 모델링 작업이 어렵다.

결국, UML 기반의 UI 모델링은 UML을 확장하여 웹 응용 및 범용 응용 분야의 개발에 적용될 수 있도록 UI 모델, UI 메타모델 그리고 개발 프로세스를 제공하고 있다. 그러나, 제시하는 방법에 따라 UI 구조물이 상이하고, 계층적 모델링이 부분적이고, 전 개발과정을 지원하지 못한다. 또한, UI, 비즈니스 및 DB 클래스들간의 상호 통합 연계 모델화 그리고 변환과정에 구체성과 체계성이 결여되어 있다. 특히, PIM과 PSM에 입각한 실제 구현 환경까지 포함하는 전체적 UI 모델링 체계가 미흡하고 이들간 변환기법이 완전치 못하다. 이로서, 일관성있는 UI 모델의 생성이 어렵고, 추상화 수준별 모델의 생성에 저해가 됨으로서, UI 모델의 품질과 재사용성의 저하를 야기한다.

3. 계층적 UI 메타모델

UI-비지니스 통합 모델링 프로세스를 구축하기 위해, 먼저, UI 모델링 프로세스가 정립되어야 한다. 즉,

개발 단계에 따른 UI 모델을 정하고, 이를 위한 UI 계층적 메타모델로 정의하고, 이 메타모델에 기반한 UI 모델링 프로세스를 정립해야 한다. 본 장에서는 계층적 UI 메타모델을 정립하기 위한 UI 설계 원리, 이 원리에 입각한 계층적 UI 메타모델 그리고 PIM-UI 모델의 PSM-UI 모델로의 변환기법을 제시한다.

3.1 UI 설계 원리

본 연구는 소프트웨어 설계자에게 UI 모델과 비지니스 모델의 분리와 통합의 모델링 과정을 거쳐 애플리케이션을 디자인할 수 있도록 제공하는 것이다. 목표는 다양한 응용분야 적용이 가능한 범용적이고 일원화된 계층적 UI 모델링 프로세스 및 UI-비지니스 통합 모델링 프로세스의 제공에 초점을 둔다. 이 프로세스는 UI 모델과 UI 모델링 요소가 개발 단계에 따라 상이하게 적용되는 계층적 모델링 접근을 제시한다. 본 연구에서는 비즈니스 모델링을 위해서 [3]에서 제시한 계층적 메타모델과 프로세스를 이용하고, UI 모델링을 위해서 계층적 UI 메타모델을 제시하고 비즈니스 모델링 프로세스와 통합된 모델링 프로세스를 구축하는 것이다. 이를 위한, 연구의 접근 모델은 그림 1과 같이, 대상 UI 모델을 계층화하여 UI 메타모델로 정의하고, 이를 비지니스 메타모델과 결합하여 UI 모델링 프로세스를 정립한다.

이를 위한 주요 연구내용은 다음과 같다.

- UI 모델에 대한 ***계층성***을 가진 개별 UI ***메타모델*** 정의
- 메타모델 사용의 ***MDD***(Model Driven Design) 기반 계층적 UI 모델링 프로세스 정립
- ***MVC*** 기반의 UI 모델과 비즈니스 모델간의 통합 모델 프로세스 정립

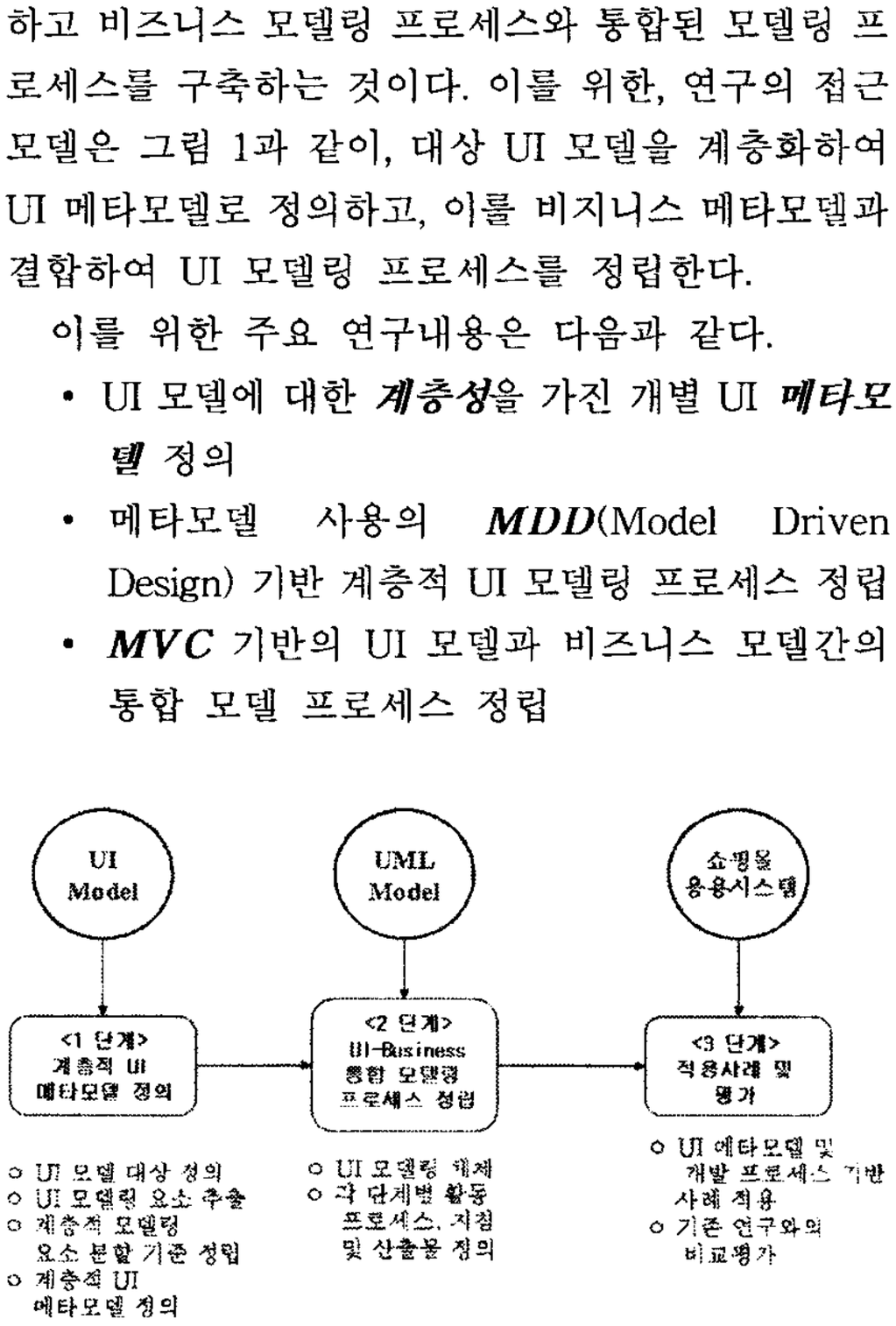

그림 1. UI 모델링의 연구 모델

먼저, ***메타모델***은 주로 모델링 언어를 정의하기 위해 사용되는데, 본 논문에서는 UI 모델의 추상적 구문(abstract syntax)을 정의하기 위해 사용한다. 메타모델 접근은 정형적 명세와 자동화 툴 구축에 용이하다. ***계층성***은 어떠한 대상물들이 계층 구조로 조직되어 있는 것이다. 계층성은 양질의 소프트웨어 재사용을 위하여 디자인 모델을 분리한다는 것에 주목해야 한다. 이 개념은 모델링의 대상 크기(granularity)와 추상화(abstraction)에 의해 분류할 수 있다. 따라서, 크기에 의한 계층성의 층화(layered)는 컴포넌트, 프레임워크, 스크린(form 등의 UI 요소로 구성)의 3계층으로 정의한다. 반면, 추상화에 의한 계층의 분류는 개발의 생명주기에 따라 개념적/ 명세적/ 상세적의 3개 모델링 단계로 분류한다. 이러한 분류는 메타모델을 계층화하고 개발 단계의 프로세스를 계층화하는데 적용된다. ***MDD***는 아키텍쳐 모델에 기반하여 설계 모델과 구현 모델을 분리하여 접근하는 PIM과 PSM 기술을 의미한다. ***MVC***는 UI를 효과적으로 데이터 모형에 관련시키기 위한 설계 방식이다. MVC의 M은 실체 클래스, V는 UI클래스, C는 제어클래스를 나타내며, 이것은 UI 객체, 비즈니스 및 DB 객체를 식별하고 분리하기 위해 사용한다.

UI 모델의 구축을 위한 설계 원리는 아래의 개념에 입각한다.

- UI 모델링 추상화 수준에 부합한 PIM 및 PSM 개념 기반의 UI 모델 작성
- UI 모델별 계층적 UI 메타모델의 구축은 실용적이고 범용적인 핵심 모델링 요소와 그들 간의 관계를 가지고 모델링 개발 단계에 준거하여 명확하게 정의
- 3 계층에 기반을 둔 계층적 UI 메타모델 및 비즈니스 모델과의 통합 모델링 프로세스를 구축
- UI 메타모델내 정의된 UI 모델링 요소에 기반한 UI 모델링
- MVC에 입각한 UI/ Business Logic/ DB 객체의 분리 및 통합 반복적 모델링

3.2 계층적 UI 메타모델

계층적인 UI 모델링 프로세스를 제공하기 위해, 개별 계층적 UI 모델이 필요하다. UI 모델 차원의 계층적 분할 기준 및 분할, UI 메타모델 차원의 UI

모델링 요소들간의 계층화 분할 기준 및 관계의 생성 규칙이 필요하다. 이때, 개별 UI 메타모델은 모델링 단계의 추상화 정도에 부합토록 구성되어야 한다. 다시 말해, 계층적 UI 메타모델은 개발 생명주기의 구체화에 따라 일치하는 확장적 모델링 요소들이 추가됨에 의해 계층적으로 분할되어 구성되어야 한다. 모델링 단계에 의한 개별 계층적 UI 메타모델의 분할 기준은 다음과 같다.

- UI 모델의 모델링 요소 자체가 지닌 본질적 용도 또는 추상화 수준의 적용
  예로, PIM 모델을 작성하는 Specific UI Metamodel은 구현 플랫폼 환경에 무관하게 범용적 적용의 필수적 UI 모델링 요소로 구성되어야 한다.
- 개발 단계별에 따라 UI 모델의 모델링 요소가 다른 모델의 구축 진행 수준과 동일화
  UI 모델링이 기존 UML 모델들(비지니스 모델)과의 연계 모델링이 이루어져야 하므로, 이들과의 상호 산출물 입출력 관계 및 전체적 추상화 수준에 부합한 모델링 요소들 이어야 한다.

UI의 모델들은 실제적으로 다양한 모델링 요소들과 그들 간의 관계들을 가지고 있다. 따라서, 일관성과 명료성을 유지하기 위해서는 구축 모델들간의 모델링 요소들과 관계들을 추출하기 위한 공통적 적용의 생성 규칙이 필요한데, [3]에 제시한 규칙을 준용한다.

이러한 계층적 분할 기준과 메타모델의 생성규칙에 따라, UI 모델을 2 단계로 계층화하여 명세적 단계 적용의 Specific UI Metamodel과 상세적 단계 적용의 Concrete UI Metamodel로 계층적 UI 메타모델을 구축한다. 따라서, 계층적 UI 메타모델은 모델링 단계의 그 추상화의 정도에 따라 적합한 모델링 요소들을 계층적으로 나타낸다. UI 메타모델을 2개의 계층으로 정의한 이유는 MDD 기반의 PIM/PSM 개념을 적용함으로서, 그 추상화 정도에 따른 UI 모델의 재사용성을 증진시키는데 효과적이기 때문이다. 또 다른 이유로 두 개의 메타모델로 이원화함으로서 UI 모델의 복잡한 모델링 요소들의 식별을 용이토록 하는데 있다. 메타모델은 UML의 클래스 모델(class model)로 표현한다.

따라서, 개발 단계의 3 계층에 의한 UI 모델링 작업은 개념적 모델링 단계에서 유스케이스 명세서 모델내에 UI 스케치 수준으로 모델링하며, 이 산출물을 활용하여 명세적 모델링 단계와 상세적 모델링 단계에서 두 개의 계층적 UI 메타모델을 각각 적용해서 실질적인 UI 모델링 작업이 수행된다.

3.2.1 Specific UI Metamodel

명세적 모델링 단계에서 UI 모델의 목적은 명세적 추상화 수준에서 시스템의 단위 기능을 대상으로 UI를 프레임워크 크기에서 UI 요소(element)에 이르는 구현 플랫폼과 독립된 PIM 형태로 논리적인 UI 기술(technology)을 명세하는 것이다. 즉, 한 개의 유스케이스를 하나의 프레임워크로 간주하여, UI 관점(view)에 따라 하나의 프레임워크를 구성하는 여러 스크린 레이아웃(screen layout)들을 스케치한 후, 각 레이아웃별 구성되는 여러 스크린들을 추출하고, 스크린내 다수의 form들을 명세하고, 객체 수준의 form내 UI widget들(가령, 버튼, 메뉴 등)을 생성한다. 이 후, 이들 간의 상호 연관 및 포함 관계에 의해 UI 모델로 구조화한다. 이러한 구조물을 가지는 specific UI model을 정의하는 specific UI metamodel은 스크린/ form/ widget 등을 기반으로 모델링 요소들을 추출하고, 이들 간의 구조와 관계에 대해 클래스 모델로 표현한다. 한편, 이 단계의 모델링 수준은 프레임워크(framework) 크기이므로 이 프레임워크를 구성하는 UI widget들까지 명세하여 이들로 UI 클래스들을 정립한다. 이때, PIM 개념에 준거하여 구현 플랫폼과 무관한 모델링 요소를 추출해야 하므로 특정 UI의 widget들을 포함시키지 않고 다양한 응용분야 적용의 범용적 모델링 요소들로 구성하여야 한다. 즉, 단순성과 범용성을 갖기 위해 최적의 모델링 요소로 구성한다. 그림 2는 명세적 단계의 PIM 수준상에서 수행하여야 할 UI 모델링 요소들을 가지고 생성된 specific UI metamodel을 나타낸다.

그림 2에서, specific UI metamodel의 구성 모델링 요소를 살펴보면, framework는 UI_view와 sitemap 요소로 표현되며, UI_view는 스크린과 screen_flow로 구성되며, form은 static과 active의 UI_element로 나타낸다. UI_view 모델링 요소는 유스케이스 명세서 내 정상/서브/대안/예외 시나리오의 view에 따른 필요한 화면과 화면 흐름을 나타낸다. 그 외, UI 요소에 따라 이벤트를 발생하므로 이를 위한 event listener

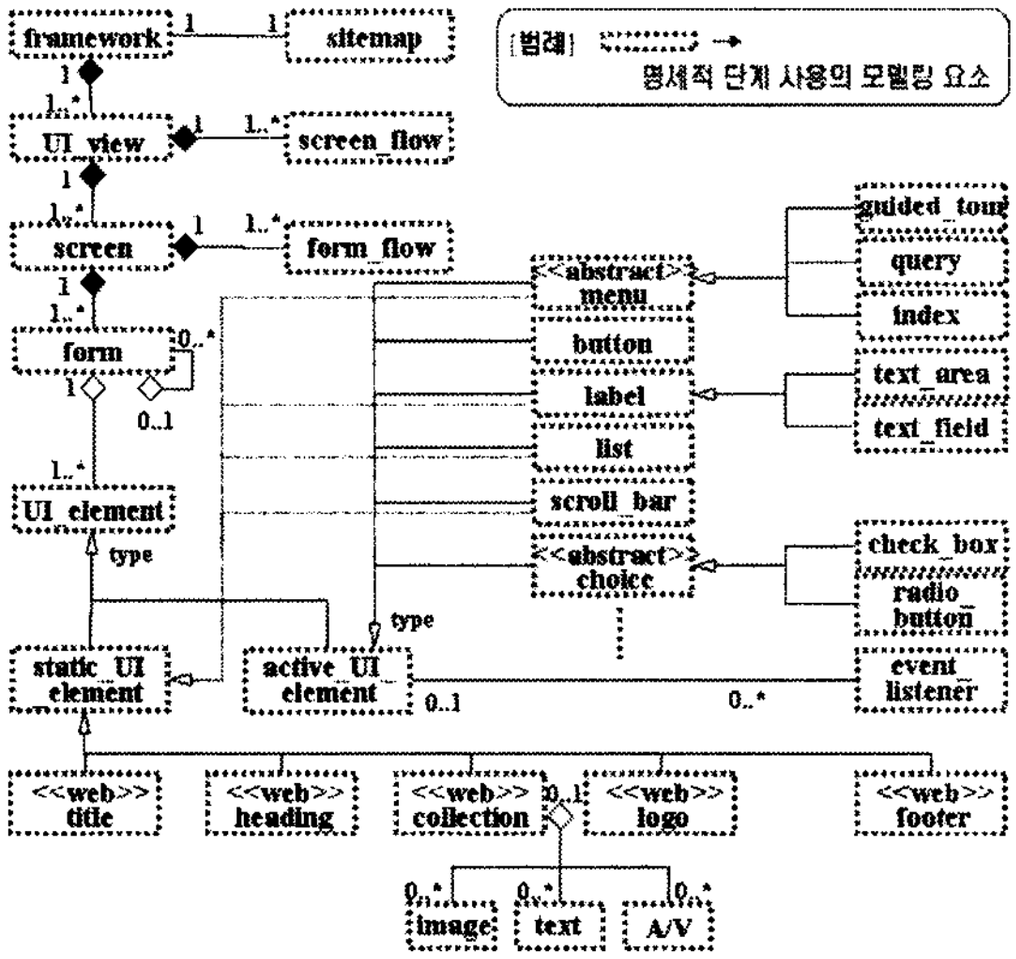

그림 2. 명세적 단계 적용의 계층적 Specific UI Metamodel

요소가 있다. 모델링 요소간의 관계를 보면, 예로 form과 form flow는 특정 스크린에 고유한 요소이기에 포함(composition) 관계로 표현한다. 반면, form에 적재되는 UI widget들인 버튼, 텍스트 등은 타 form들과 공유하여 사용될 수 있기에 집합(aggregation)의 관계로 표현한다.

3.2.2 Concrete UI Metamodel

상세적 모델링 단계에서 UI 모델의 목적은 이전 specific UI model의 PIM 모델을 상세적 수준에서 concrete UI model의 PSM 모델로 변환, 구체화하여 명세하는 것이다. 즉, 구현 환경의 특정 플랫폼에 종속된 UI 패키지의 타입에 맞게 UI 요소들을 변경하고, 구체화 해주는 물리적 기술(physical technology)로 UI 모델을 생성한다. 먼저, 이 모델을 작성하기 위한 입력물로 이전 개발단계에서 생성된 specific UI model의 UI 클래스들과 상세적 단계의 이전 작업 활동의 산출물인 프리젠테이션 모델을 이용해서 concrete UI model을 구축한다. 왜냐하면, 프리젠테이션 모델은 스크린내 UI 요소들간의 레이아웃 및 그들 간의 상호 관계성을 보여주는데, 이것을 이용해서 concrete UI model을 구성하는 UI 클래스내에 메소드들을 추가할 수 있기 때문이다. PIM UI 모델에서 PSM UI 모델로의 변환 과정(단원 3.3 참조)은 specific UI model에 특정 구현 UI 플랫폼 타입에 맞게 UI 클래스 타입을 변경하고, UI 클래스에 메소드를 추가하고, 필요시 이벤트(예, 버튼)를 위한 인터페이스를 추가함으로서 concrete UI model로 변환, 생성할 수 있다. 결국, concrete UI model의 모델링 요소는 특정 플랫폼의 UI 요소 그리고 좀더 UI 디자인을 구체적으로 설계해주는 추가적 상세 UI 요소들로 구성되어야 한다. 본 논문에서는 범용적 사용이 많은 자바를 대상으로, 자바에서 지원하는 UI 플랫폼 즉, 패키지의 UI widget들을 가지고 concrete UI metamodel을 정의한다. 자바의 UI 지원은 기본적인 UI를 제공하는 AWT와 고급스런 UI를 지원하는 SWING 패키지를 가지고 있는데, 이 모델링 요소들을 기반으로 concrete UI metamodel을 정의한 것이 그림 3이다. 그림 3에 나타낸 concrete UI metamodel의 모델링 요소를 살펴보면, 특정 플랫폼인 자바 UI 관련의 AWT와 SWING[20]의 다양한 UI widget들(가령, card_layout, JPannel 등), UI의 상세화 모델링 요소로서 position/ size/ method 등이 있으며, 그리고 특정 이벤트 관련 인터페이스들을 추가하여 메타모델을 구성하였다. 이 메타모델을 사용해서 플랫폼 종속토록 concrete UI model을 생성하는 것이다.

이러한 계층적 UI 메타모델을 통해, UI 설계자는 메타모델에 명세된 UI 모델링 요소와 관계들을 사용해서 쉽고, 정확하게 애플리케이션에 대한 UI 모델을 추상화의 수준에 따라 계층적으로 구축할 수 있다. 즉, 통합 UI 모델링 프로세스에서 이 메타모델에서 정의한 메타모델의 모델링 요소 및 관계에 기반해서, 모델링 활동을 포함하고 있기 때문이다.

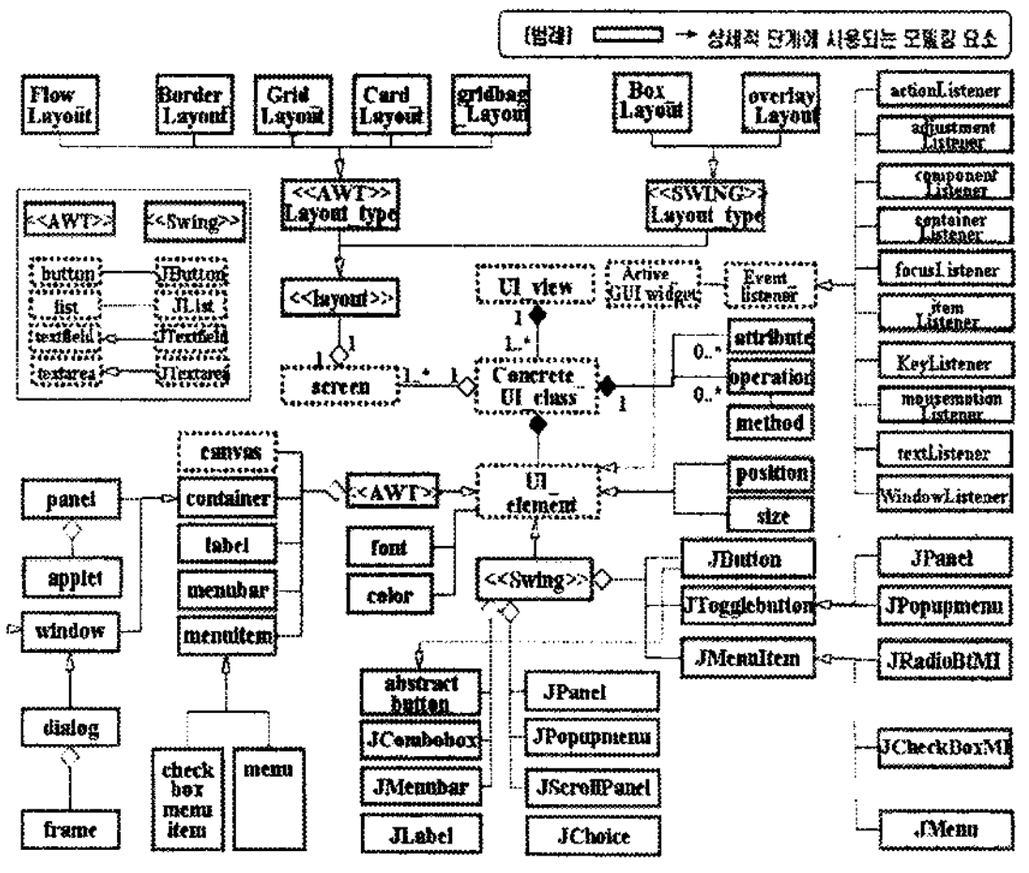

그림 3. 상세적 단계 적용의 계층적 Concrete UI Metamodel

표 2. PIM-UI 모델의 명세를 위한 UML 프로파일

PIM UI 요소	UML 명세		요소 설명
	메핑 UML 요소	UML 표현	
화면	class	<<screen>>	어느 시점에 디스플레이 되는UI widget들을 가진 응용 윈도우 혹은 웹 페이지
화면 흐름	relationship	<<screenflow>>	사용자의 일련의 활동에 의한 스크린 간의 흐름 순서
정적 UI widget	class	<<static UIelement>>	행위 혹은 상호작용을 제공하지 않는 UI 요소
동적 UI widget	class	<<active UIelement>>	행위 혹은 상호작용을 제공하는 UI 요소
.....	.....	.....	.....

3.3 PIM-UI 모델의 PSM-UI 모델로의 변환 기법

UI 모델링 과정은 명세적 단계와 상세적 단계를 따라, 구현 플랫폼에 독립적인 PIM-UI 모델과 구현 환경에 의존적인 PSM-UI 모델을 생성하면서 수행한다. 따라서, PIM-UI 모델을 PSM-UI 모델로 변환할 수 있는 기법이 요구된다. 이에 앞서, PIM-UI 모델은 UML의 확장 기법을 사용해서 클래스 모델로 표현해야 한다. 즉, PIM-UI model을 UML로 명세하기 위해, 확장된 UML-UI 프로파일이 필요하다. PIM-UI 모델의 명세를 위한 UML 확장 프로파일을 나타낸 것이 표 2이다. 표 2는 PIM-UI 모델의 구성 요소들에 대비되는 UML 클래스 모델의 모델링 요소들과의 메핑 관계를 보여준다. 이 표 2의 메핑 프로파일에 따라 PIM-UI 모델을 클래스 모델로 표현할 수가 있다.

UML의 클래스 모델로 작성된 PIM-UI 모델을 PSM-UI 모델로의 변환방법은 구현 UI 플랫폼 환경에 부합토록 클래스 타입을 변경(예로, SWING 경우 예, button → Jbutton)하고, UI 패키지(package)로부터 상속받는 인터페이스를 추가하고, 이벤트 처리를 위한 이벤트 핸들러와 UI widget별 메소드들의 추가함으로서 수행된다. 여기서, PIM-UI 모델에서 PSM-UI 모델로 클래스 타입의 변경을 위해서 타입간 메핑 프로파일이 필요하다. 이를 위해, 표 3은 자바 SWING의 경우, PIM-UI 요소들을 PSM-UI 요소들로 클래스 타입(class type)의 변환을 위한 메핑 프로파일을 나타낸다.

이러한 변환 프로파일을 적용한 PIM-UI 모델에서 PSM-UI 모델로의 변환 예로서, 그림 4는 "고객 등록" 화면에 대한 PIM-UI 모델에서 PSM-UI 모델로의 변환을 보여준다.

표 3. PIM-UI 요소의 PSM-UI 요소로의 메핑 프로파일

PIM-UI 요소	PSM-UI 요소	메핑 UML 요소
Screen	Window	class
Form	JPanel	class
MenuBar	JMenuBar	class
MenuItem	JMenuItem	class
Button	JBotton	class
RadioButton	JRadioBtMI	class
.....	.....	.....

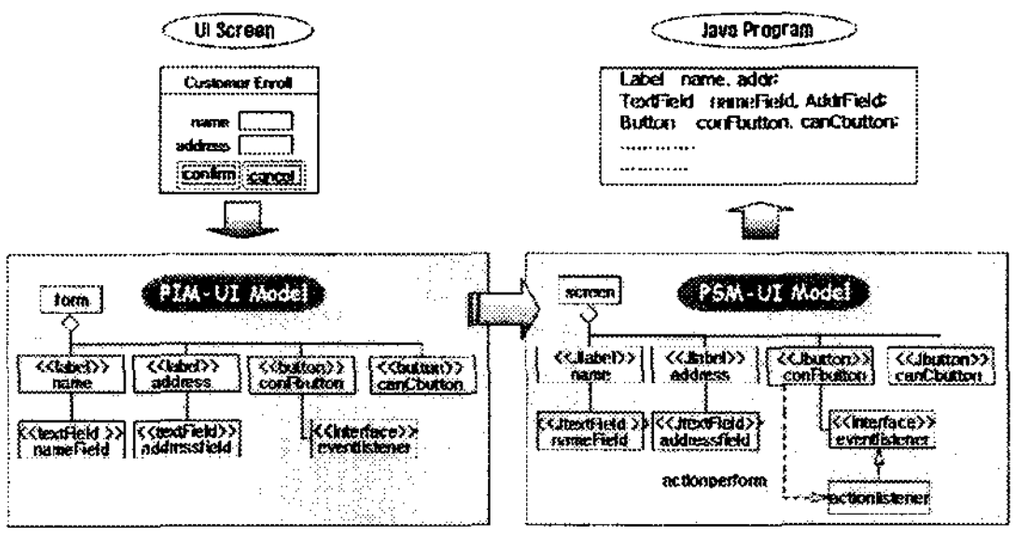

그림 4. PIM-UI 모델의 PSM-UI 모델로의 변환 사례

4. UI-비지니스 통합 계층적 모델링 프로세스

설계 모델의 일관성 유지, 품질과 재사용성의 향상을 위해서, 계층적 메타모델을 이용한 UI 모델링 작업과 비즈니스 모델링 작업이 통합된 계층적 소프트웨어 모델링 프로세스가 필요하다. 이를 위해, UI 모델링 수행 프로세스와 비즈니스 모델링 활동이 통합 조직화된 개발 단계별 상세한 모델링 작업의 수행 체계, 활동 지침 및 산출물을 기술한다.

4.1 계층적 UI 모델링 프로세스 체계

계층적 모델링 프로세스는 개발의 생명주기의 추상화 수준에 따라 단계별로 상위 추상적 모델링에서 하위 구체적 모델링에 이르기까지 점진적으로 디자인 해나가는 것이다. 개발 단계 간에는 활동의 구분이 있고, 결과물이 생성되고, 결과물간의 연계성을 가지고 있어야 한다[21,22]. 단계별 산출물은 MDD에 기반하여 개별 구축 모델로서 재사용이 가능해야 한다. 본 논문에서 제시하는 방법은 UI 모델이 개발의 3 단계에 의해 단원 3.2절에 정의된 계층적 UI 메타모델의 모델링 요소들을 가지고 계층적으로 분할된 UI 모델을 디자인할 수 있도록 제공하는 것이다. 즉, UI 모델이 명세적 단계와 상세적 단계의 2 단계에 걸쳐서 디자인되는 생성자(constructor) 혹은 모델링 요소에 사용의 제약을 주어서 계층적으로 접근토록 제공하는 것이다. 이를 통하여, 전체 모델링 작업을 혼돈없이 쉽게 수행토록 제공하며, 구축 모델의 재사용을 극대화시킬 수 있도록 하는데 있다.

통합 UI-비지니스 개발 프로세스 체계는 그림 5에 나타내었다. 모델링의 단계는 개념적 모델링 단계, 명세적 모델링 단계, 그리고 상세적 모델링의 3개 단계로 구성한다.

각 단계는 관련된 활동들의 집합체이다. 활동은 목표 모델의 모델링 행위를 나타내며, 프로세스 내에서 활동 상태(activity state)로 표현한다. 이 체계는 추상화의 성숙 정도에 따라 계층적 개발을 지원하며, 제시된 메타모델을 사용해서 구축된 모델의 정확성에 대한 검사과정을 수행토록 하며, 반복적인 피드백 기능을 제공한다. 사용되는 계층적 메타모델로서, UI 모델링을 위한 UI 메타모델 그리고 비즈니스 모델링을 위한 UML 메타모델이 사용된다. UML 메타모델은 [3]에서 제시한 use case metamodel, use case description metamodel, class metamodel 및 sequence metamodel을 이용한다. 따라서, 제시하는 통합 모델링 체계는 UI와 비즈니스 모델링을 포함한다. 그림 5에서, UI 모델링 작업은 개념적 단계에서 최상위 수준의 UI 스케치 수준으로 모델링하고, 명세적 및 상세적 단계에서 실질적인 UI 모델링 작업이 이루어진다. 명세적 단계에서 PIM 형태의 UI 모델을 생성하고, 이를 기반으로 상세적 단계에서 특정 자바 플랫폼 환경에 부합하여 PSM 형태의 구체적인 UI 모델이 생성된다.

이 계층적 모델링 프로세스의 특징은 다음과 같다. 첫째, 각 개발 단계별로 계층적 모델링이 이루어지도록 현 단계에서 요구하는 수준에 맞는 모델링 요소들을 정의한 메타모델을 이용한 개발 프로세스를 들 수 있다. 즉, 그림 5에서, "5. 명세적 UI 클래스 모델" 작성시 그림 2상의 specific UI metamodel에서 정의한 명세적 수준의 모델링 요소와 관계를 사용해서 specific UI 모델을 작성한다는 것이다. 둘째로, 설계 오류를 최소화하기 위해 추상화 수준에 따라 일원화되고 체계화된 UI 모델링 프로세스를 들 수 있다. 왜냐하면, 초기 UI 스케치에서 관심의 분리 패러다임을 적용해서 MVC를 통한 UI 객체와 비즈니스 객체를 분리하여 디자인하고, PIM과 PSM에 의한 조직적인 분할 설계를 지원하기 때문이다. 그림 5상의 명세적 모델링 활동에서, 통상 클래스 모델을 생성한 후, 객체 순차 모델을 작성하나, 본 모델링 프로세스에서는 UI 객체의 추출을 위해 상위의 작업 활동으로 두었다. 셋째로, MVC 모델에 의한 UI 모델과 비지니스 모델과의 명확한 통합 디자인을 제공한다.

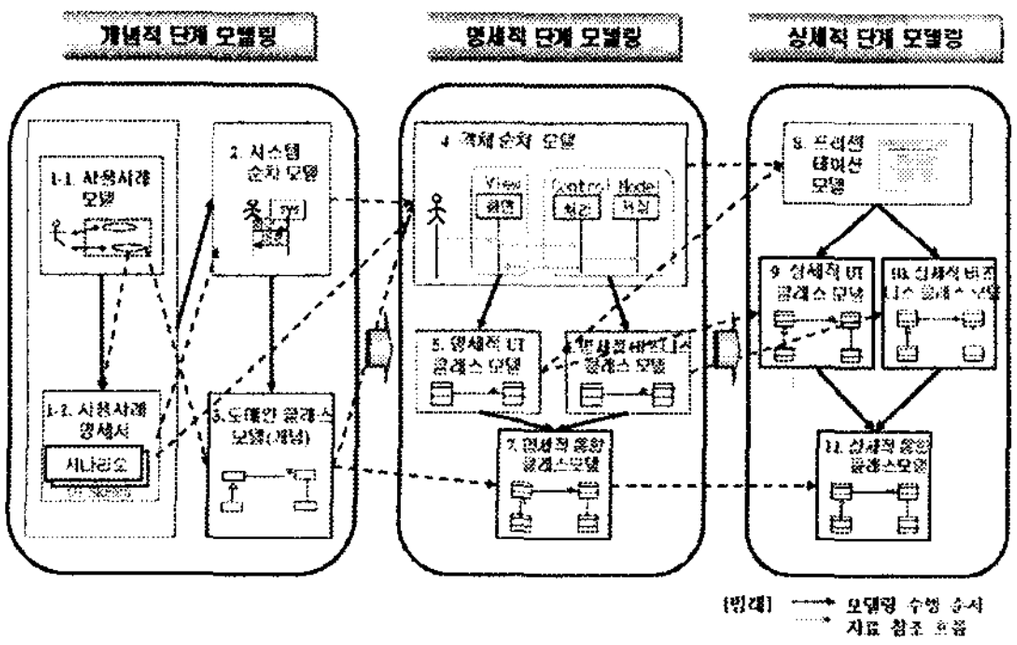

그림 5. 계층적 UI-비지니스 통합 모델링 프로세스 체계

4.2 개발 단계별 모델링 프로세스

4.2.1 개념적 모델링 단계

가장 최상위 수준의 아키텍쳐/컴포넌트 차원에서 목표 도메인에 대한 요구사항들을 분석한다. 이 단계의 주요한 활동은 요구 도메인에 대해 개념적 수준에서 관련된 모델들을 생성한다. 이 단계에서는 use case metamodel, use case description metamodel과 conceptual class metamodel[3]의 구조물을 사용한 모델링을 통해서 산출물로 유스케이스(사용 사례)

모델, 유스케이스 명세서, 시스템 순차 모델, 도메인 클래스 모델을 생성한다. 이 단계에서 UI 모델링 작업 활동은 미약하나마, 유스케이스 명세서내에 시나리오(정상/예외 등) 흐름에 따른 초기 스케치 수준의 UI 모델링을 수행한다. 또한, 사용자와 시스템간의 인터페이스를 모델링하는 시스템 순차 모델을 통해 후속 단계에서 UI 요소 모델링을 위한 기본적인 산출물을 생성한다.

4.2.2 명세적 모델링 단계

명세적 단계의 모델링은 프레임워크 수준에서 목표 도메인에 대해 PIM 형태로 UI 모델링과 비즈니스 모델링을 수행한다. 실질적인 UI 작업이 이루어지는 단계로 MVC 모델에 근간하여 추상적 수준의 UI 모델(UI 클래스)과 비즈니스 모델(제어 클래스와 DB 클래스로 구성)을 분리[16]하여 모델링한 후, 이들을 결합하여 통합 클래스 모델을 생성한다. UI 모델과 비즈니스 모델을 분리하는 이유는 "관심의 분리"(separation of concern)의 원리에 입각해 특성 영역별(UI와 비즈니스) 모델의 재사용성을 높이는데 있다. 즉, UI 혹은 비즈니스 모델중 어느 한쪽 변경시, 모델의 변경을 최소화하여 모델의 재사용을 높일 수 있다. 한편, 이들을 통합하는 이유는 응용 서비스에 대한 전체 시스템의 구조 및 상호관계를 한 눈에 파악하기 위함이다. 즉, UI 클래스를 통해 입력된 값들이 제어 클래스를 통해 처리가 되고, 이것이 최종 DB 클래스를 통해 DB에 저장되는 일련의 과정이 하나의 전체 클래스 모델에 표현되어야 하기 때문이다. 이 단계에서는 입력물로 개념적 단계의 산출물인 유스케이스 모델, 유스케이스 명세서, 시스템 순차 모델 및 도메인 클래스 모델을 사용해서, 출력물로 객체 순차 모델, specific_level UI 클래스 모델, specific_level 비즈니스 클래스 모델 및 specific_level 통합 클래스 모델을 산출한다. 이때, 이 단계에 적용되는 메타모델은 sequence metamodel, specific UI metamodel 그리고 specific class metamodel이다. 명세적 단계의 모델링 과정을 그림 6에 나타내었다.

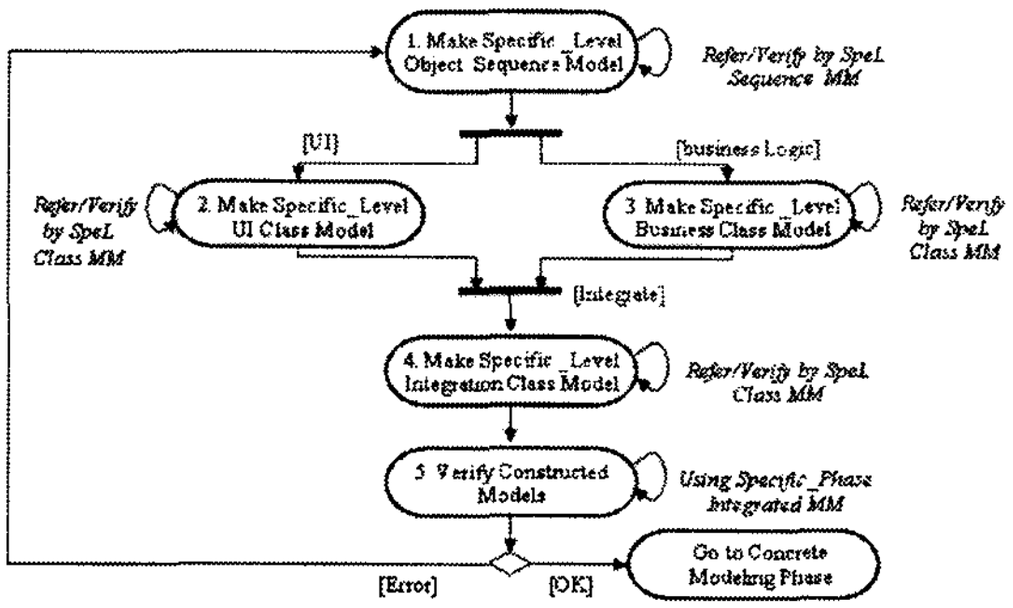

그림 6. 명세적 단계의 모델링 프로세스

표 4. 명세적 단계의 모델링 활동 과정

순번	모델링 활동 내용
1	유스케이스 명세서과 시스템 순차 모델에 기반하여 한 개 프레임워크(1개 유스케이스)를 대상으로 MVC 모델에 의해 한 개의 객체 순차 모델을 생성한다. 이때, 명세적 수준(specific_level)의 sequence metamodel([3] 참조: 명세적 단계에서 수행되는 순차 모델의 모델링 요소를 정의)에 정의한 모델링 요소를 사용해서 모델을 작성하며, 또한, 이 메타모델에 정의된 모델링 요소간의 관계에 의거해서 생성 모델의 일관성을 검사한다.
2	유스케이스 명세서의 UI 스케치, 객체 순차 모델의 UI 부문(MVC상의 'view') 그리고 specific UI metamodel을 사용해서 구현 플랫폼에 독립적인 명세적 수준의 UI 클래스 모델을 디자인한다. 이 모델은 specific UI metamodel 메타모델에 의거하여 정확성을 검사한다.
3	순번 2와 병행적으로 도메인 클래스 모델과 객체 순차 모델의 비즈니스 부문(MVC 모델상의 'controller'와 'model') 기반으로 specific class metamodel을 사용해서 비즈니스 클래스와 DB 클래스로 구성된 명세적(specific_level) 비즈니스 클래스 모델을 생성한다.
4	객체 순차 모델에 명세한 그들 간의 관계성에 기반하여, 순번 2와 순번 3에서 생성된 UI 클래스 모델과 비즈니스 클래스 모델을 결합한 통합 클래스 모델을 생성한다.
5	구축된 모델들에 대한 상호 관계성의 일관성을 검사하기 위해, 명세적 단계 적용의 통합 메타모델([3] 참조: 3계층에 의해 UML 모든 모델들에 대한 한 개의 메타모델을 정의)에 의거하여 생성 모델을 검사한다.

그림 6에서 명세적 단계의 개발 프로세스를 상세하게 그 활동을 설명한 것이 표 4로서, 그림 6상의 활동별 순번과 표 4의 순번은 일치한다.

특별히, UI 클래스 모델은 한 개 유스케이스를 대상으로 하나, 비즈니스 클래스 모델은 전체 시스템을 대상으로 한다. 왜냐하면, 통합 클래스 모델의 구축은 전체 시스템을 대상으로 표현해야 하기 때문이다. 한편, UI 클래스 모델은 2 레벨로 모델링한다. 첫 레벨은 한 개 유스케이스를 대상으로 하고, 두 번째 레

벨은 한 개 화면(스크린)을 대상으로 모델링한다. 이 때, 두 번째 레벨로 작성된 각 UI 클래스 모델은 후속 상세적 모델링 단계에서 각각 한 개의 프리젠테이션 모델로 화면 설계를 한다.

4.2.3 상세적 모델링 단계

상세적 모델링 단계는 객체 수준의 응용시스템 크기를 대상으로 목표 도메인에 대한 구체적인 디자인과 모델의 정제화 작업을 수행한다. PSM 형태로 자바 특정 플랫폼(AWT와 SWING 등)에 부합하도록 UI 모델링과 비즈니스 모델링을 수행한다. 각 스크린별 화면을 설계(프리젠테이션 모델로 표현)하고, 이 모델과 단원 3.3절의 변환기법을 적용해서 구현 플랫폼 환경을 가미하여 상세한 UI 클래스 모델을 설계하고, 또한 비즈니스 클래스 모델을 구체화한 후에, 다시 UI 모델과 비즈니스 모델을 통합하여 상세한 통합 클래스 모델을 생성한다. 여기서, 프리젠테이션 모델링 작업이 UI 상세화 모델링 작업이 보다 선행 작업으로 먼저하는 이유는 프리젠테이션 모델의 한 화면(스크린)내 구성하는 UI 요소들간의 레이아웃과 상호 협력 관계의 식별을 통해서, 상세 UI 클래스 모델내 UI 클래스에 객체간의 통신 메시지에 의한 메소드를 추가하여 구체화할 수 있기 때문이다. 이 단계의 세부적 모델링 활동의 과정은 표 5와 같다.

결국, 메타모델 기반의 UI와 비즈니스 객체가 결합된 계층적 모델링 프로세스는 UI 모델이 어떻게 PIM/PSM의 추상화 수준에 따라 계층적으로 모델링 되는지 그리고 UI 모델과 비즈니스 모델의 분리/통합되어 모델링 되는지에 대한 체계적이고 상세한 가이드를 보여준다. 이를 통해 소프트웨어 설계자는 정확하고 재사용성이 높은 품질의 모델을 생성할 수 있어 모델링에 소요되는 노력 즉, 비용을 경감시킬 수 있다.

표 5. 상세적 단계의 모델링 활동 과정

순번	모델링 활동 내용
1	스크린 크기 대상의 명세적 UI 모델을 기반으로 한 개 스크린별 UI 프리젠테이션 화면 모델을 생성한다. 화면은 UI widget들의 컨텐츠 구성 및 배치 등 레이아웃을 설계한다.
2	specific UI model과 프리젠테이션 모델을 기반으로 concrete UI metamodel을 사용해서 상세적 UI 클래스 모델을 생성한다. 구현 플랫폼 UI 환경에 맞게 클래스의 타입을 변경하고, 인터페이스를 추가하고, 클래스내 메소드를 추가한 후, 관계 재정립에 따른 모델을 재구조화한다.
3	명세적 비즈니스 클래스 모델에 대해 구현 환경(예, EJB, .NET, CCM 등)에 일치토록 모델을 구체화하여 상세적 비즈니스 클래스 모델을 생성한다.
4	상세 UI 클래스 모델과 비즈니스 클래스 모델을 통합하여 최종 서비스 시스템에 대한 통합 클래스 모델을 생성한다.

5. 적용 사례

제시한 계층적 UI 및 비즈니스 통합 모델링 프로세스의 실효성을 검증하기 위하여, 인터넷 쇼핑몰 시스템(ISMS: Internet Shopping Mall System)을 대상으로 UI 모델 및 UI-비지니스 통합 프로세스를 적용한다. 적용 방법은 제 IV장에서 명세된 모델링 프로세스에 따라 제 III장의 단계별로 UI 메타모델에 정의된 계층별 요소들을 가지고 모델링을 수행한다. 그림 5에 따른 모든 산출물중 UI 관련 모델을 일부 발췌하여 기술하였다.

5.1 개념적 모델링 단계

ISMS 시스템에 대한 개략적 도메인 분석과 UI 스케치를 수행한다. 이 단계는 단원 4.2 절 4.2.1에 기술한 개념적 단계의 모델링 활동 절차에 따라 유스케이스 모델과 유스케이스 명세서, 시스템 순차 모델 그리고 도메인 클래스 모델을 생성한다. 이 단계의 UI 작업으로는 유스케이스 명세서내에 기술되는 UI 스케치를 들 수 있다. 먼저, 기능 분석을 위한 유스케이스 모델은 유스케이스 계층적 메타모델을 사용해서 시스템 수준의 크기를 대상으로 유스케이스 모델과 유스케이스 명세서를 작성한다. 유스케이스 모델은 "고객"외 2개의 행위자, "상품" 주문외 21개의 유스케이스를 추출하여 모델화하였다. 예로, "상품 주문" 유스케이스에 대한 유스케이스 명세서는 그림 7과 같다.

그림 7에서 "상품주문" 유스케이스에서 필요한 UI 스케치를 볼 수 있다. "주문 목록", "지불" 및 "배송"에 관련된 사용자와 시스템간의 UI 화면을 나타낸다. 이것이 초기의 UI 스케치 작업이다. 다음은 유스케이스 명세서의 시나리오와 UI 스케치를 기반으로 [3]에서 제시한 계층적 sequence metamodel을 이

유즈케이스 명칭	상품 주문
유즈케이스 명세	고객이 상품을 쇼핑카드에 추가한 후, 결제수단과 배송정보를 고객으로부터 받아 주문 처리
행위자 / 유즈케이스 타입 / 관계 유즈케이스	고객 / Primary 쇼핑카드검색, 상품정보검색, 고객정보검색
정상 흐름	[고객] [N1] 고객이 주문할 상품들을 쇼핑카드에 하나 이상 추가 한 후, 주문을 요청한다. [시스템] [N2] 시스템이 고객에게 주문서 내역을 화면에 보여 준다.
대안 흐름	[A1] 신용카드 결제 - 시스템은 카드번호, 유효기간,할부 개월수를 고객에게 요청한다.
예외 흐름	[E1] 시스템은 고객이 배송/결제/비밀번호에 대해 잘못된 정보가 있음을 확인하였다.
비 스케치	

그림 7. "상품 주문" 유스케이스에 대한 유스케이스 명세서

용해서 "상품 주문" 유스케이스에 대해 고객과 시스템간의 시스템 순차 모델을 구축한다. 이때, 행위자와 시스템간의 주고 받는 메시지를 가능한 화면 중심으로 표현한다.

5.2 명세적 모델링 단계

ISMS에 대한 기본적 UI 설계 작업으로서, 한 개의 유스케이스를 한 개의 프레임워크로 메핑하여 PIM 수준의 UI 클래스 모델을 작성한다. 먼저, MVC 모델에 입각하여 UI 클래스와 비즈니스 클래스를 분리하고 이들 간의 관계성을 중심으로 그림 8과 같이 객체 순차 모델을 작성한다.

그림 8에 보여주듯이, UI 화면 클래스들(MVC중 View)로서 "쇼핑카드 화면", "주문 화면", "결재 등록 화면", "결재 화면" 및 "주문 결과 화면"을 생성하였고, 비즈니스 클래스들(MVC중 Controller와 Model)로서 "주문 관리", "결재" 등의 클래스들을 추출하였다. 이 객체 순차 모델을 통해 사용자가 입력한 것이 UI 객체에서, 제어 객체를 통해 DB 객체로 저장 처리되는 과정을 알 수 있다.

다음 활동으로 객체 순차 모델의 UI 클래스들을 대상으로 두 레벨로 분할된 한 개 유스케이스 대상의 그림 9와 한 개 화면 대상의 그림 10과 같은 명세적 UI 클래스 모델들을 생성하였다. 먼저, 레벨-1인 그림 9의 UI 클래스 모델에서, UI 클래스 명칭과 오퍼레이션은 객체 순차 모델의 객체 명칭과 객체간의 주고 받는 메시지에 근간하여 추출한다. 속성은 개념적 단계의 도메인 클래스 모델에 기반하여 생성한다. 이어서, 그림 9상의 각 화면 클래스별로 specific UI metamodel을 사용해서 레벨 1-1의 명세적 UI 클래스 모델을 작성한 것이 그림 10이다.

예로서, 그림 10은 그림 9의 "결재 등록 화면" (한 개 스크린) UI 클래스를 대상으로 작성한 것이다. 스크린 요소로서 "결재/배송 등록" 클래스를 생성하였고, form 요소로서 "타이틀", "결재" 및 "배송"의 3개를 추출하였다. 또한, 정적(static)인 UI 요소인 라벨(label) 등의 요소를 사용해 각 form의 타입에 맞게 구성하였다.

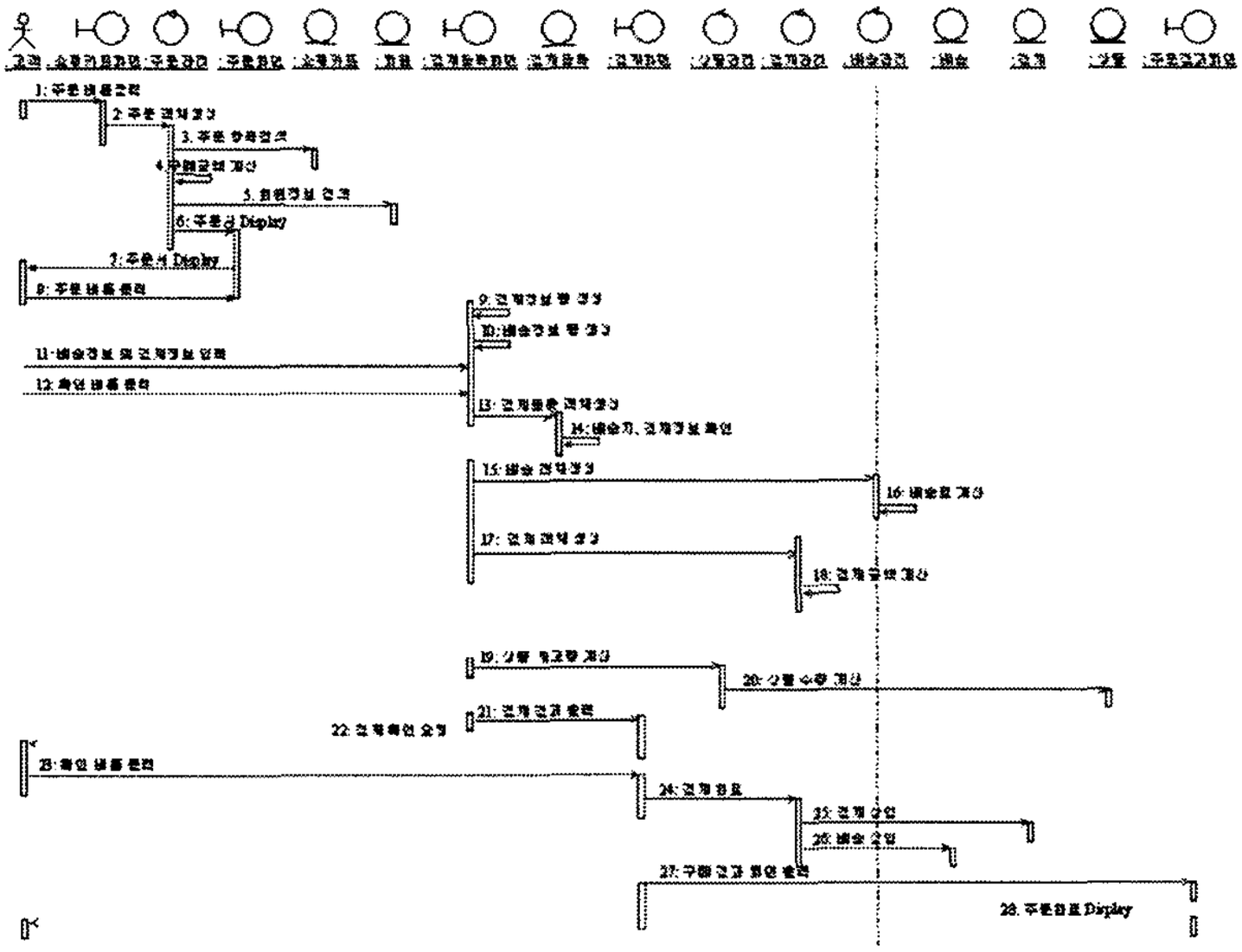

그림 8. 명세적 모델링 단계의 객체 순차 모델

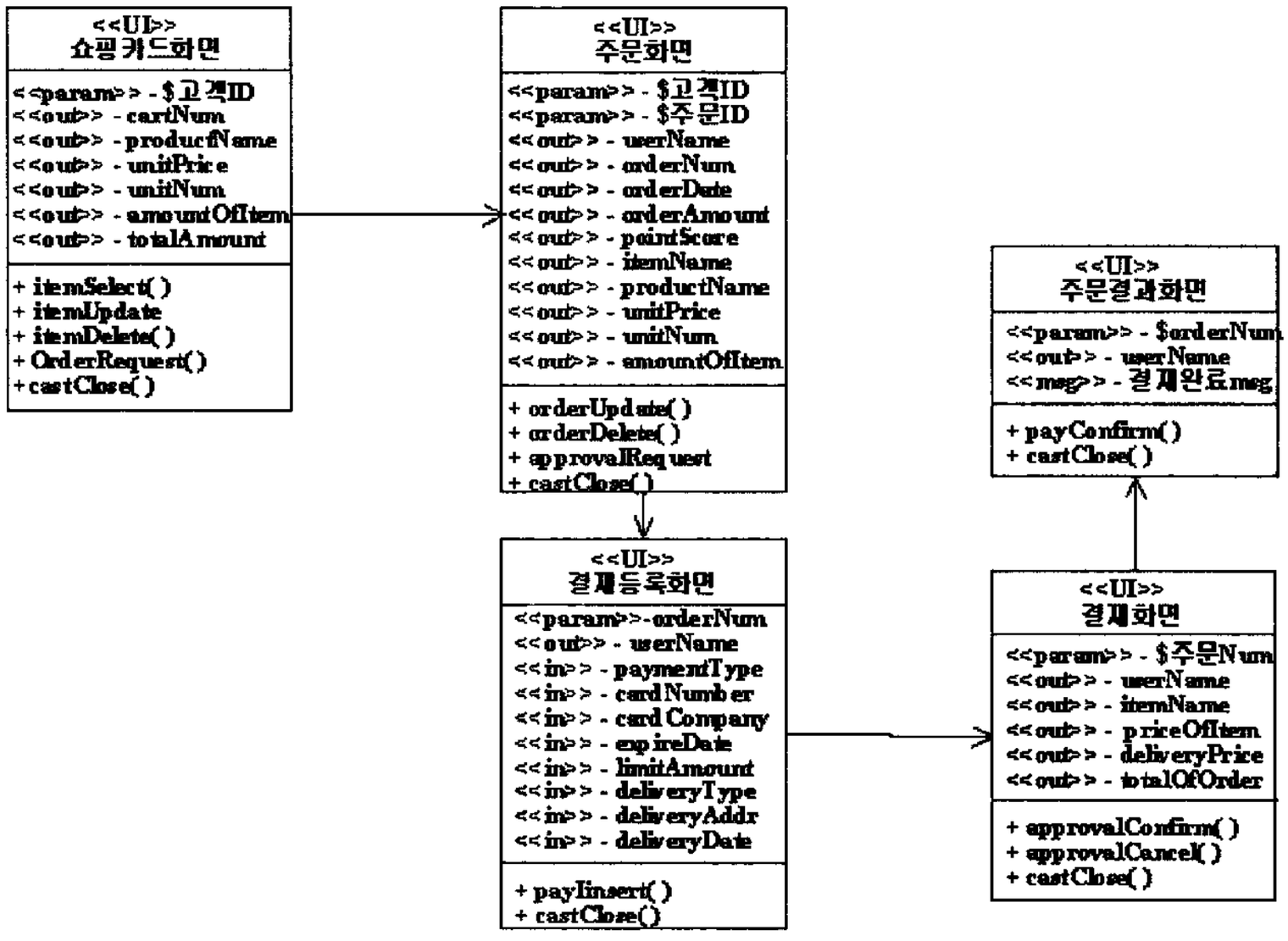

그림 9. 명세적 모델링 단계의 레벨-1 수준의 명세적 UI 클래스 모델

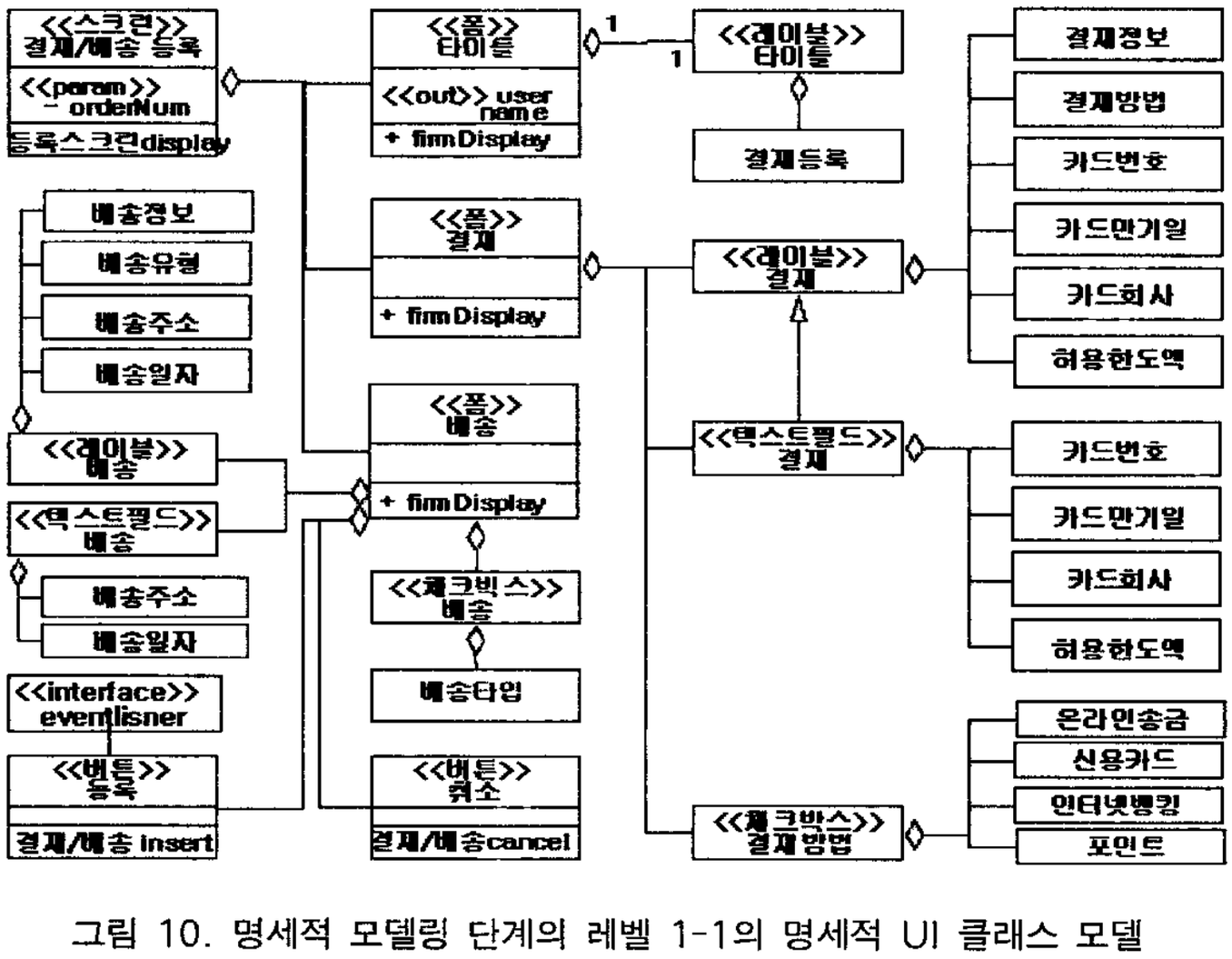

그림 10. 명세적 모델링 단계의 레벨 1-1의 명세적 UI 클래스 모델

한편, ISMS 시스템의 비즈니스 모델은 시스템의 전체적 구조 및 관계를 파악하기 위해 한 개의 유스케이스를 대상으로 하지 않고, 전체 시스템을 범주로 클래스 모델을 작성한다. 명세적 비즈니스 클래스 모델은 객체 순차 모델의 MVC중 C와 M 객체를 이용해서 비즈니스 클래스와 DB 클래스로 구성한다. DB 클래스로 "고객정보"외 7개, 제어 클래스로 "고객관리"외 10개의 클래스를 추출하여 디자인하였다. 전체적 객체 구조를 표현하기 위해 UI 클래스와 비즈니스 클래스를 결합한 통합 클래스 모델이 그림 11이다.

5.3 상세적 모델링 단계

ISMS에 대한 상세 설계로서, 구현 언어 및 UI 플랫폼의 환경에 부합토록 UI 클래스 모델과 비즈니스 클래스 모델을 구체화한다. UI 모델링 작업으로 레벨 1-1 명세적 UI 클래스 모델을 대상으로 프리젠테이

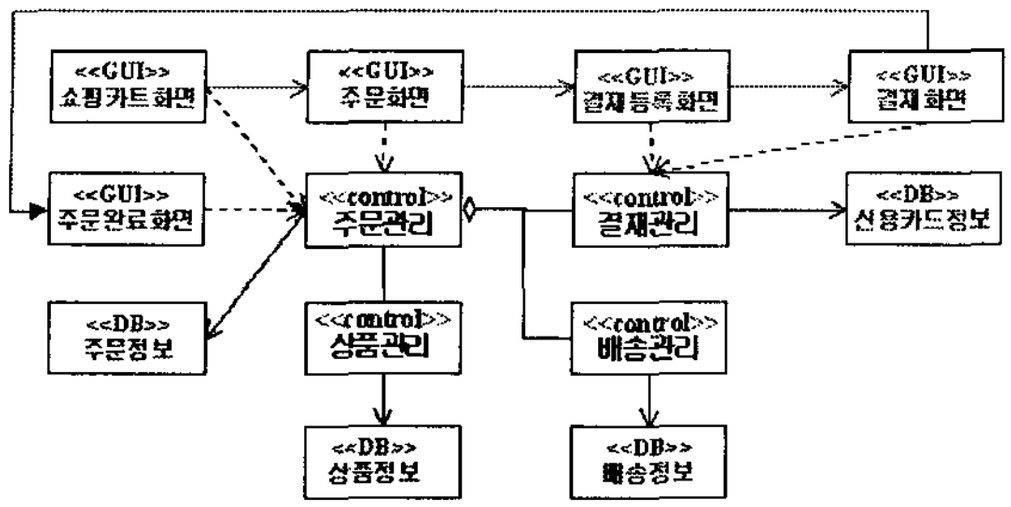

그림 11. 명세적 모델링 단계의 통합 클래스 모델

션 모델을 생성하고, 자바 UI 플랫폼에 맞게 단원 3.3절의 변환기법을 적용해서 PSM 형태의 2 레벨의 상세 UI 클래스 모델을 작성한다. 그리고, 명세적 단계에서 생성된 비즈니스 모델을 구현 언어 등에 부합토록 상세적 비즈니스 클래스 모델을 생성한다. 끝으로 이들을 결합한 통합 클래스 모델을 작성한다.

프리젠테이션 모델은 한 개 스크린 크기 대상으로 이전 단계에서 작성된 레벨 1-1 UI 모델을 기반으로 한 개 스크린별로 한 개의 프리젠테이션 화면 모델을 생성한다. 화면은 UI widget들의 컨텐츠 구성 및 배치 등의 레이아웃을 설계한다. 예로서, "결재 등록"에 유스케이스에 대한 생성된 화면 모델은 그림 12와 같다.

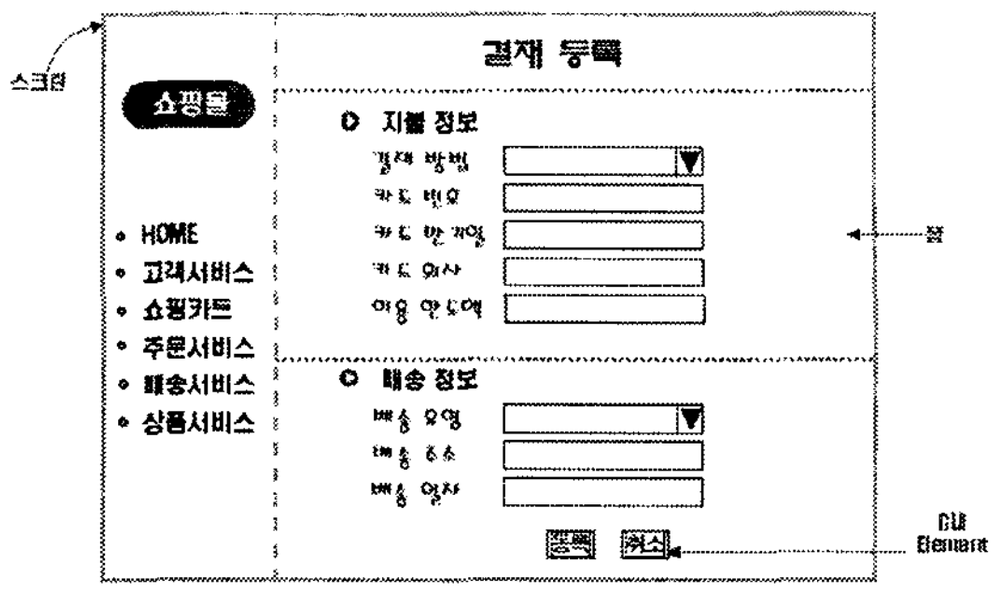

그림 12. 상세적 모델링 단계의 프리젠테이션 모델

이어서, 상세적 UI 클래스 모델의 구축은 명세적 UI 클래스 모델과 프리젠테이션 모델을 입력물로 concrete UI metamodel을 사용해서 상세적 UI 클래스 모델을 생성한다. 상세적 UI 클래스 모델은 명세적 단계와 마찬가지로, 두 레벨의 UI 클래스 모델을 생성한다. 레벨-1의 UI 클래스 모델("상품 주문"의 한개 유스케이스 대상)이 그림 13이다. 그림 13은 명세적 단계의 UI 클래스 모델그림 9을 구체화한 것으로 구현 언어 등을 스테레오타입(stereotype)으로 해당 UI 클래스내에 추가하여 작성하였다.

이를 상세화한 한 개 화면 대상의 레벨 1-1 상세적 UI 클래스 모델이 그림 14이다. 이 모델은 명세적 UI

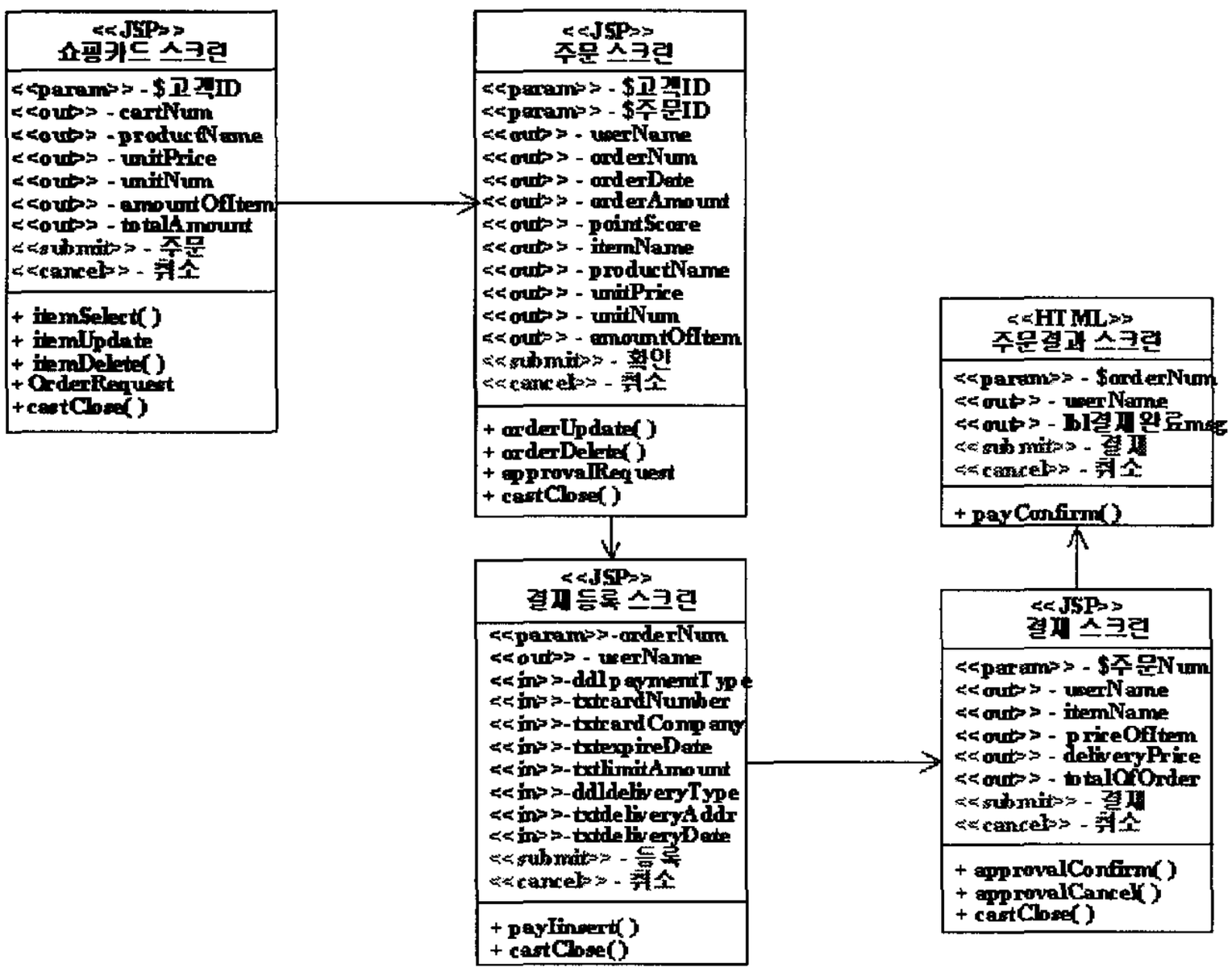

그림 13. 상세적 모델링 단계의 레벨-1의 상세적 UI 클래스 모델

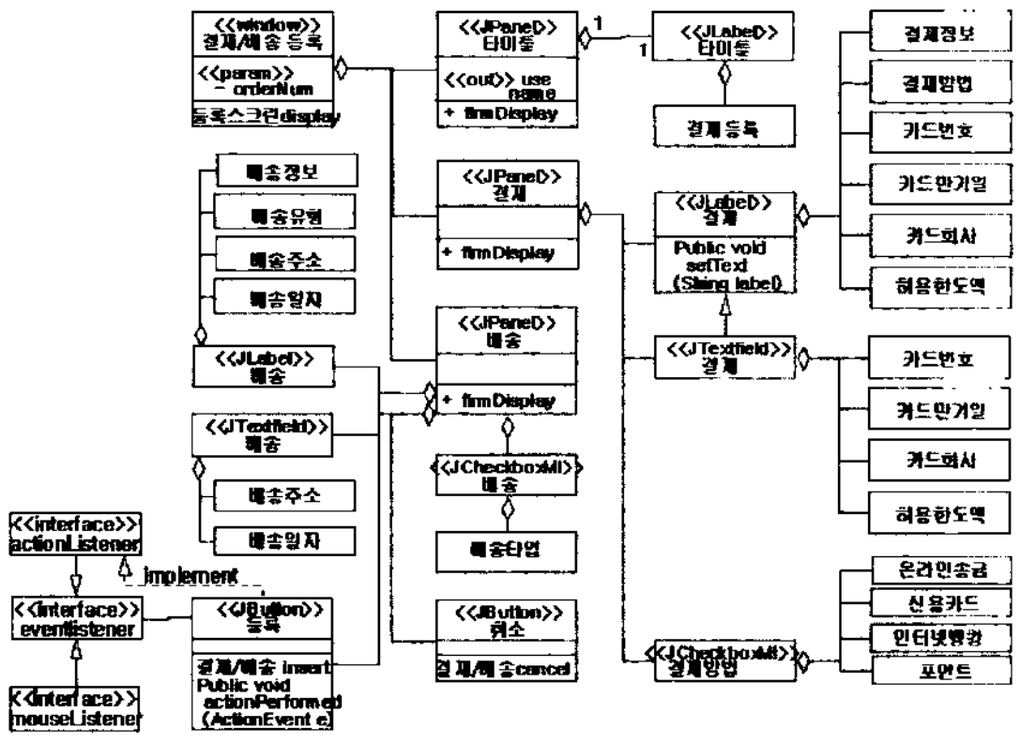

그림 14. 상세적 모델링 단계의 레벨 1-1의 상세적 UI 클래스 모델

클래스 모델 그림 10에 특정 자바 SWING UI 플랫폼에 맞는 UI 모델링 요소를 추가적으로 사용해서 구축한다. 즉, 그림 14의 모델은 그림 13의 "결재 화면" 클래스의 한 개 스크린에 대해 자바의 "Swing" UI 패키지를 적용하였을 경우를 보여준다. 그래서, UI의 클래스 타입을 Swing의 타입으로 변경하고(가령, "form" UI 클래스 타입을 "JPannel" UI 클래스 타입으로 변환), 인터페이스 타입을 추가하고(예, mouseListener), 각 UI widget별 메소드를 추가한다. (예, 결재 label 클래스내 "setText" 메소드)

또한, 후속 모델링 활동으로 명세적 비즈니스 클래스 모델을 입력물로 자바 구현 언어의 플랫폼에 맞게 상세적 비즈니스 클래스 모델을 생성한다. 마지막으로 상세적 UI 클래스 모델과 상세적 비즈니스 클래스 모델을 통합한다. 통합 클래스 모델은 명세적 단계에서 생성된 그림 11의 통합 클래스 모델을 구현 플랫폼에 부합토록 UI 클래스에 대해서는 <<JSP>>/<<ASP>> 등, 제어 클래스는 <<SesionBean>>(J2EE 플랫폼 경우), DB 클래스는 <<EntityBean>>으로 스테레오타입으로 표현하여 모델화한다.

6. 비교 평가

ISMS 시스템 모델링의 적용사례를 통해서, 계층별(PIM/PSM) UI 메타모델 및 UI-비지니스 통합 모델링 프로세스를 적용해서 개념적 모델링 단계에서 상세적 모델링 단계에 이르기까지 명확하고, 쉽게 계층적으로 UI 및 비즈니스 모델링이 수행됨을 알 수 있었다. 이로서, 소프트웨어 개발을 위해, UI 모델링과 비즈니스 모델링이 별개의 개발 프로세스를 가지는 것을 통합하였고, [3]에서 계층적 비즈니스 개발 프로세스에 국한된 것을 계층적 UI 모델링 프로세스 부문까지 포함해서 완전한 UML 기반의 계층적 통합 모델링 방법을 구축하였다. 제시한 기법에 대해 UI 모델, UI 모델링 프로세스 및 메타모델 측면에서 현존 연구와 비교 평가를 한다.

UI 모델 측면에서, 표 6은 기존 연구과의 비교를 나타낸 것으로 제공하는 UI 관련 모델, 이를 정의한 UI 메타모델 그리고 그 모델링 요소들을 분석한 것이다. 표 6에서, UIM_UMLi, GUI_Layout, 그리고 본 연구는 UI 모델링 기법에 초점을 둔 연구이다. UI 모델링을 위한 UI 모델의 수준 측면에서, UWE는 항해 모델과 프리젠테이션 모델을 제시하고, 특히 항해 모델은 모델링의 추상화 수준에 따라 2 단계로 계층화 하였다. UIM_UMLi 연구는 UI 모델과 2 계층화된 프리젠테이션 모델을 제시했다. GUI_Layout은 UI 모델들의 개발체계가 체계적으로 정립되어 있는데, 2 계층화된 GUI 레이아웃 모델, 스크린 모델, 항해 모델, sitemap 모델 및 스토리보드 모델로 구성되어 있다. 본 논문은 2 계층화된 UI 클래스 모델과 프리젠테이션 모델을 제시하였다. 따라서, 기존의 연구에서 UI 모델은 부분적인 계층적 모델링을 제공하나, 각 연구별로 상이한 UI 접근 모델들로 구성되어 있어 사용에 어려움이 있으며, 완전한 UML 기반의 UI 모델을 제공하지 못한다. 즉, GUI_Layout 경우, 항해 모델은 UML에서 제시하는 모델을 사용치 않는 표기법을 사용한다. 이에 비해, 본 논문은 UML의 클래스 모델을 사용해서 UI 모델을 체계적이고, 계층적으로 표현하고, 소프트웨어 아키텍쳐의 3계층(UI, 비즈니스 로직, DB)에 부합하여 UI 클래스를 분리, 추출할 수 있도록 제시하였다.

이어서, 제시한 UI 모델들의 모델링 요소를 가지고 명확한 구문적 구조를 정의하는 메타모델 측면에서, 각 연구별로 표 6과 같이 메타모델을 정의하고 있다. 그러나, 이들의 메타모델의 구조가 전체적 아키텍처에 초점을 두어, 이해와 사용이 어렵다. 반면, 본 논문의 제시하는 계층적 UI 메타모델은 개발의 추상화 수준에 따라 단계별 specific UI 모델과 concrete UI 모델에 대해 각각 메타모델을 정의하고 있어 개별적 사용이 용이하다. 한편, UI 모델링을 위해

표 6. 기존연구의 제시 UI 모델들과의 비교

구 분	UWE	UIM_UMLi	GUI_Layout	제시 방법
Target application	Web application Web modeling focus	General application UI Modeling Focus	General application UI modeling Focus	General application UI modeling Focus
UI Model	Use case model Conceptual model **Abstract Navigation model I** **Concrete Navigation model II** Process model (structure, flow) **Presentation model**	**User interface model** **Abstract Presentation model** Task model **Concrete presentation model**	**Conceptual GUI Layout Diagram** **Screen model** Activity model **Navigation model** **Sitemap model** **Storyboard model** **Concrete GUI Layout model**	Use case Model/ Description **Object Sequence model** **Specific UI class model** Specific Integrated class model **Presentation model** **Concrete UI class model** Concrete Integrated class model
UI Metamodel	UWE metamodel Conceptual metamodel Navigation metamodel Abstract user interface metamodel Presentation metamodel	Abstract Presentation metamodel Concrete presentation metamodel	UI Layout metamodel UI Layout: Reference metamodel Screen metamodel	**Specific UI metamodel** **Concrete UI metamodel**
Number of UI Modeling element	22 modeling elements (form 등)	19 modeling elements (window etc)	31 modeling elements (screenarea etc)	82 modeling elements (form etc)

[범례] 굵은 글자 : UI 관련 모델

필요하고 충분한 UI 모델링 요소를 제공하는지에 대해 살펴보면, 본 논문의 경우, 82개의 UI 모델링 요소를 제공한다. 이 수는 specific UI 메타모델과 concrete UI 메타모델에서 UI 관련 핵심 모델링 요소의 수를 대상으로 산정한 것이다. 그 모델링 요소의 수가 의미하듯이, 기존 연구의 UI 모델링 요소는 다양한 애플리케이션 적용의 UI 개발을 위해 그 모델링 요소가 충분치 못하고, 실용적이고, 범용적이지 못하다. 또한, 일부 연구에서 부분적으로 모델링 요소를 계층화하여 나타내고 있으나, PIM과 PSM에 의한 구분이 명확치 않으며, 특히 특정 플랫폼과 연계된 UI의 PSM에 의한 구축방법이 구체적이지 않다.

기존 연구(RUP[19] 등)와의 비교로서, 표 7은 UI 모델링 프로세스에 의한 비교를 나타낸다.

표 7에서, 기존 연구는 일원화된 UI 모델링 프로세스 측면에서 초기 유스케이스 수준에서 객체 수준에 이르는 UI 모델 및 비즈니스 모델 기반하의 모델링 체계가 상세적이고 조직적이지 못하다. 또한, 객체 대상의 UI 모델과 비즈니스 모델간의 분리와 통합의 계층적 개발 프로세스가 미약하다. 그로 인해, UI 및 비즈니스 모델의 품질에 대한 일관성을 보장하기가 어렵다. 반면, 제시 기법은 범용적 사용의 UI 모델링 요소 및 구조물, PIM/PSM 기반의 계층적 모델링 그리고 UI-비지니스 모델간의 구체적인 결합 모델링

표 7. UI 모델링 프로세스에 의한 비교

구 분		UWE	UIM_UMLi	GUI_Layout	RUP	CBD Bible	제시 방법
UI 모델링	UI 모델	+	+	+	-	o	+
	UI 모델링 요소 수준	+	o	+	-	o	++
	UI 모델링 요소 계층화 (PIM/PSM)	+	+	+	-	-	++
	UI 계층적 모델링 프로세스	+	+	+	--	+	++
UI 모델과 비즈니스 모델간 통합 프로세스		-	-	-	-	+	+

[범례] ++: 아주 양호, +: 양호,
o: 보통, -: 부족, --: 아주 부족

프로세스를 제시하고 있다. 또한, UI와 비즈니스 간을 분리하여 모델화함으로서 객체 단위의 UI 모델의 재사용을 향상시키고 UI 컴포넌트화를 기할 수 있다. 또한, 모델의 변경 발생시, 해당 모델만(UI 모델 혹은 비즈니스 모델)을 수정함으로서 영향을 최소화 할 수 있다. 아울러, UI 모델과 비즈니스 모델의 통합을 통해 전체 시스템의 구조를 일관성있게 구축할 수 있다.

본 방법은 메타모델을 기반으로 모델링 프로세스 및 세부 지침 등을 구축한 것이다. 이에, 표 8은 표 7의 비교 자료를 토대로, 제시 방법과 기존 방법과의 메타모델에 의한 품질비교를 나타낸다. 각 연구 방법별 품질 측정을 위한 기준은 다음과 같다.

- 레벨 1 : UI 메타모델이 정의
- 레벨 2 : UI 메타모델이 정의되고, 개발단계별 UI 모델링 요소를 계층화(MDD기반)
- 레벨 3 : UI 메타모델이 정의되고, 개발 단계별 UI 모델링 요소를 계층화되어 있고, 개발 프로세스를 제공
- 레벨 4 : UI 메타모델이 정의되고, 개발 단계별 UI 모델링 요소를 계층화되어 있고, 개발 프로세스를 제공하며, 세부 개발활동별 지침(instruction)을 제공
- 레벨 5 : UI 메타모델이 정의되고, 개발 단계별 UI 모델링 요소를 계층화되어 있고, 개발 프로세스를 제공하며, 세부 개발활동별 지침(instruction)을 제공하고, PIM-PSM간 변환기법을 제공하고, UI 모델과 비즈니스-UI 모델간 통합 프로세스를 제공

표 8에서 각 연구별 품질 매트릭스에 따른 숫자 값의 의미는 위에 언급한 레벨의 등급을 나타낸다. 예로, UWE는 메타모델을 정의하고 세부 UI 개발지침 등을 제공하기에 통합성, 확장성, 독립성 및 재사용성의 등급이 4가 된다. 메타모델상에서 통합성이 의미하는 것은 타 모델과의 정합을 잘 이루어짐을 나타낸다. 이유는 UI 모델을 메타모델로 정의하였기에 메타모델간 통합이 용이하기 때문이다. 또한, 메타모델로 표현하고 있기 때문에, 추가적 모델링 요소의 확장이 쉬우며, MDD 기반의 모델링 요소를 계층화하여 PIM 모델과 PSM 모델간 독립성을 제공할 수 있다. 또한, 메타모델의 재사용성은 모델링요소 추가의 경우, 기존 메타모델을 재사용해서 추가로 첨가하여 구성하면 된다. 따라서, 메타모델을 이용한 모델링 프로세스는 사용이 용이하고 일관된 모델의 구축에 따른 모델의 품질을 보장하고, 추상화 수준에 따른 계층화에 따른 재사용성을 향상시킬 수 있다.

표 8. 메타모델에 의한 품질 비교

구 분	UWE	UIM_ UMLi	GUI_ Layout	RUP	CBD Bible	제시 방법
통 합 성	4	4	4	3	3	5
확 장 성	4	4	4	3	3	5
독 립 성	4	4	4	3	3	5
재사용성	4	4	4	3	3	5

7. 결 론

소프트웨어 개발에 있어 기존에는 UI 모델링이 체계적이지 못하고, UI 모델링과 비즈니스 모델링이 별도의 프로세스를 가지고 있어, 생성 모델의 품질과 재사용성의 저하를 야기하였다. 본 논문에서는 UML 확장을 통해 UI 모델을 정의하고 모델링 프로세스, UI와 비즈니스 모델과의 통합에 의한 메타모델 사용의 계층적 모델링 프로세스를 제시하였다. UI 모델을 PIM과 PSM에 준거하여 계층화된 UI 메타모델을 정의했다. 제시 UI 메타모델과 기존 비즈니스 모델링을 위한 UML 메타모델을 기반으로 3 단계의 MDD형 계층적 통합 모델링 프로세스를 정립했다. 이때, MVC 모델에 준거하여 UI 모델과 비즈니스 모델을 분리와 통합의 프로세스를 구축하였다. 인터넷 쇼핑몰 응용시스템을 가지고 계층적 UI-비지니스 통합 모델링 프로세스에 따라 이 기법의 실효성을 사례연구를 통하여 입증하였다. 또한, 기존 UI 모델과 개발방법론들과의 비교평가를 하였다.

기대효과로 계층적 UI 모델링 프로세스에 의해 UI 모델들을 구축함으로서, UI 모델의 품질과 재사용성을 증진시킬 수 있다. 즉, UI 메타모델의 모델링 구조물을 메타모델로 정형화해서 사용함에 따른 일관성 있는 모델을 구축할 수 있으며, 추상화 수준별 그리고 UI 모델과 비즈니스 모델의 특성별 재사용성을 증진시킬 수 있다. 또한, 애플리케이션의 완전한 계층적 모델링을 수행할 수 있다. 이것은 기존의 비즈니스 부문의 계층적 모델링에 대해 UI 모델링을 추가

하고 UI와 비즈니스간의 통합 모델링에 의해 가능해졌다. UML 표준으로의 활용 측면에서 OMG UML 4-layer 아키텍처상의 M2 계층에 참조모델로서 체화될 수 있으며, RUP와 같은 현존 개발방법론에 적용할 수 있다. 특히, 계층적 모델링 기법은 복잡하고 대형적인 시스템의 개발에 유용하다.

향후 연구로서, UI 모델의 컴포넌트 모델링 기법, PIM UI 모델을 PSM UI 모델로의 자동변환 알고리즘 및 툴 개발 그리고 제시 메타모델의 정형적 명세 및 검사가 필요하다. 또한, UML 기반의 계층적으로 응용 모델들을 편집할 수 있고, 자동적으로 모델의 일관성 검사를 할 수 있는 Meta-CASE Tool의 개발이 이루어져야 한다.

참 고 문 헌

[1] K. Blankenhorn, "A UML Profile for GUI Layout," *Master's Thesis University of Applied Sciences Furtwangen Department of Digital Media*, 2004, http://www.bitfolge.de/pubs/thesis/Thesis_GUILayout.pdf.

[2] Object Management Group, Unified Modeling Language: Infrastructure V2.1.1, 2007, http://www.omg.org/docs/formal/07-02-04.pdf.

[3] C.Y. Song and D.K. Baik, "A Layered Metamodel for Hierarchical Modeling UML," *International Journal of Software Engineering and Knowledge Engineering*, Vol.13, No.2, pp. 191-214, 2003.

[4] P. P. Silva and N.W. Paton, "User Interface Modeling in UMLi," *IEEE Software*, pp. 62-69, 2003, http://www.ksl.stanford.edu/people/pp/papers/PinheirodaSilva_SOFTWARE_2003.pdf.

[5] R. Hennicker and N. Koch, "Modeling the user interface of web applications," 2001, http://www.pst.informatik.uni-muenchen.de/~kochn/pUML2001-Hen-Koch.pdf.

[6] A. Kraus and Nora Koch, "A Metamodel for UWE," Technical Report 0301, Institut für Informatik, Ludwig-Maximilians-Universität München, Jan. 2003, http://citeseer.ist.psu.edu/cache/papers/cs/27445/http:zSzzSzwww.pst.informatik.uni-muenchen.dezSzpublicationszSzTR0301_UWE.pdf/a-metamodel-for-uwe.pdf.

[7] N. KOCH, A. KRAUS, C. CACHERO, and S. MELI "Integration of business process in web application models," *Journal of Web Engineering*, Vol.3, No.1, pp. 22-49, 2004.

[8] A. Knapp, N. Koch, G. Zhang, and H.M. Hassler, "Modeling business process in web applications withArgoUWE," 2004. http://www.pst.ifi.lmu.de/veroeffentlichungen/knapp-et-al:uml:2004.pdf.

[9] Object Management Group, MDA Guide Version 1.0.1, 2003, http://www.omg.org/docs/omg/03-06-01.pdf.

[10] P. Pauen and J. Voss, "The HyDev Approach to Model-based Development of Hypermedia Applications," *First International Workshop on Hypermedia Development*, 1998.

[11] D. Schwabe and G. Rossi, "The Object-Oriented Hypermedia Design Model," *Comm. ACM*, Vol.38, No.8, pp. 45-46, 1995.

[12] J. Conallen, *Building Web Applications with UML Second Edition.* Addison-Wesley, Boston. San Francisco, 2003.

[13] P.P. Silva and N.W. Paton, "User Interface Modeling with UML," *Information Modeling and Knowledge Bases XII*, IOS Press, pp. 203-217, 2001, http://www.cs.utep.edu/paulo/papers/PinheirodaSilva_IMKB_2000.pdf

[14] J.V. Bergh and K. Coninx, "Using UML 2.0 and Profile for Modeling Context-Sensitive User Interface," *Workshop on Model Driven Development of Advanced User Interfaces (MDDAUI 2005)*, Montego Bay, Jamaica. Vol.159, Paper 7, 2005.

[15] T. Schattkowsky and M. Lohmann, "UML Model mappings for platform independent user interface design," Springer-Verlag Berlin Heidelberg, LNCS 3844, pp. 201-209, 2006.

[16] 채흥석, *객체지향 CBD 개발 Bible*, 한빛미디어

출판사, 서울, 2003.

[17] J. Sadd, "DEFINING THE OPENEDGE® REFERENCE ARCHITECTURE - PRESENTATION: MODEL-VIEW-CONTROLLER PATTERN," PROGRAM SOFTWARE, 2006, http://www.progress.com.

[18] C. Phillips and E. Kemp, "In support of User Interface Design in the Rational Unified Process," *The Third Australasian User Interface Conference (AUIC 2002)*, Vol.7, pp. 21-27, 2002.

[19] P. Kruchten. *The Rational Unified Process - An Introduction 2nd edition.* Addison-Wesley, Boston.San Francisco, 2000.

[20] H.M. Deitel and P.J. Deitel, *JAVA HOW TO PROGRAM*, Prentice Hall, London, 2002.

[21] M.E.C. Hull, P.S. Tayler, J.R.P. Hanna, and R.J. Millar, "Software development processes - an assessment," *Information and Software Technology*, Vol.41, No.1, pp. 1-12, 2002.

[22] E.S. Cho, S.D. Kim, and S.Y. Rhew, "A Domain Analysis and Modeling Methodology for Component Development," *International Journal of Software Engineering and Knowledge Engineering*, Vol.14, No.2, 2004.

송 치 양

1985년 한남대학교 전산학과 학사
1987년 중앙대학교 전산학과 석사
2003년 고려대학교 컴퓨터학과 박사
1990년~2005년 한국통신 중앙연구소 책임연구원
2005년 10월~2008년2월 상주대학교 소프트웨어공학과 조교수
2008년 3월~현재 경북대학교 소프트웨어공학과 조교수
관심분야 : UML 모델링 기술, 컴포넌트 기반 개발방법, IP-TV 서비스

조 은 숙

1993년 동의대학교 전산통계학과 졸업(학사)
1996년 숭실대학교 대학원 컴퓨터학과 졸업(석사)
2000년 숭실대학교 대학원 컴퓨터학과 졸업(박사)
2002년~2003년 한국전자통신연구원 초빙연구원
2000년~2004년 동덕여자대학교 정보학부 전임강사
2005년~현재 서일대학 소프트웨어과 조교수
관심분야 : 컴포넌트 기반 소프트웨어 개발, 임베디드 소프트웨어개발, 유비쿼터스 컴퓨팅

김 철 진

1996년 경기대학교 전자계산학과 졸업(학사)
1998년 숭실대학교 대학원 컴퓨터학과 졸업(석사)
2004년 숭실대학교 대학원 컴퓨터학과 졸업(박사)
2004년~2005 가톨릭대학교 컴퓨터 정보 공학부 교수
2005년~현재 삼성전자 디지털 솔루션 센터
관심분야 : 컴포넌트 기반 소프트웨어 공학, 임베디드 소프트웨어 개발 방법론