# 시나리오 기반 객체 지향 기법을 이용한 인트라넷 하이퍼미디어 시스템 개발

이희석* · 유천수** · 이충석** · 김영환** · 김종호** · 조선형***

# Developing Intranet Hypermedia System Using Scenario-Based Object-Oriented Technique

Heeseok Lee* · Cheonsoo Yoo** · Choongseok Lee** ·
Younghwan Kim** · Jongho Kim** · Seon H. Cho***

## ABSTRACT

Intranet emerges as a key technology for building enterprise information system. This paper proposes a scenario-based object-oriented technique for designing intranet hypermedia information systems. The method consists of six phases such as domain analysis, object modeling, view design, navigational design, implementation design and construction. Users' requirements are analyzed in the form of scenarios by the use of a responsibility-driven object technology. Object-oriented views are generated from the resulting object model and then used for the subsequent navigational and implementation design. Implementation design phase deals with database schema, page structure and flow, and user interface. The method is effective for integrating enterprise databases with distributed hypermedia systems by employing Java language. To demonstrate its usefulness, a real-life bank case is illustrated.

## Ⅰ. 서 론

최근 정보 기술 (IT)의 급진적 변화와 발전은 이 분야에 효과적으로 대처하기 위한 많은 노력을 요구한다. 즉, 객체 지향 기술 (Object-Oriented Technology) [19]의 연구와 이를 현업에 적용하기 위한 시험 평가가 모색되어야 하며 인터넷 (Internet) 기술 응용에 의한 전략적 기회의 포착이 중요한 과제이다. 상기한 정보 기술의 변화에 의거 기업 정보 시스템 구축 환경의 변화가 불가피 하다. 현재의 인터넷 기술이 기업 내 정보 시스템 구축에 그대로 적용되는 인트라넷

* 한국과학기술원 테크노경영대학원 부교수
** 한국과학기술원 테크노경영대학원 경영공학 전공
*** 왕 컴퓨터코리아(주) 대표이사
주 : 본 연구는 한국과학기술원 일반 수탁 연구과제(GI 90020)로 수행되었습니다.

(Intranet) [2] 기술은 기존 정보 시스템 환경에 많은 변화를 미칠 것으로 예상된다.

본 논문에서는 기업 내 정보 자원을 특정 사용자나 사용자 그룹의 정보 요구에 맞게 효과적으로 구축하여 이용할 수 있는 인트라넷 기반 하이퍼미디어 (Hypermedia) 어플리케이션 개발 방법이 소개된다. 또한, 실제 업무에 적용한 프로토타입 시스템을 개발하여 인트라넷의 유효성을 평가하고 발전시켜 현업에서 효과적으로 활용할 수 있도록 하였다. 본 고에서는 편의상 금융 계통 분야 업무를 대상으로 시나리오 기반 객체 지향 기법을 적용하여 하이퍼미디어 어플리케이션의 실제 구현까지 거치는 실 개발 결과를 소개한다.

하이퍼미디어 시스템은 멀티미디어 노드들을 링크를 통해 네트워크를 구성하여 정보를 표현하고 관리하기 위한 새로운 형태의 정보 시스템이다 [14]. 정보의 노드와 링크 및 브라우징이라 불리는 네비게이션 인터페이스의 단순한 구조에 의해 사용자는 원하는 정보를 쉽게 찾고 검색할 수 있다. 하이퍼미디어 시스템의 미래에 대한 상반된 의견이 있으나 하이퍼미디어가 현재 및 미래의 정보 시스템을 위한 매체로서의 역할을 충실히 담당할 것으로 믿어진다.

이미 전 세계적으로 하이퍼미디어 기능의 유용성과 최근의 WWW (World-Wide-Web)에 의한 정보의 체계적인 조직화 및 활용의 용이성이 충분히 알려져 있다. 현재 WWW가 일반적인 개념으로 받아들여지고 있으나 웹 (Web) 기술을 제한된 조직이나 그룹 내에서 폐쇄적으로 활용하는 것에 대한 인식이나 동의는 최근에야 이루어지고 있다. 이의 결과가 바로 조직 차원의 웹 (Corporate-Wide-Web) 또는 인트라넷의 구축이다. 인트라넷은 외부와의 연결 활용보다는 조직 내에서 인터넷 기술을 구현하여 사용하는 수사적인 용어라 할 수 있다. 인트라넷에 의해 저렴한 네트워크 비용과 비교적 간단한 기술을 이용하여 지리적 장벽 없이 조직 내의 시스템과 독자적으로 설치 활용 중인 네트워크를 대체할 수 있을 것으로 기대된다. 예를 들어 공동 저작, 의사결정 지원, 문서관리, 교육 및 훈련, 문서화 작업, 소프트웨어 공학 및 팀 공동작업과 같은 다양한 응용 업무에 활용될 수 있을 것이다 [5]. 특히, 인트라넷에서 조직의 응용 업무 개발을 위한 플랫폼으로서 웹 기술이 이용될 수 있으므로 비용이 많이 들고 사용이 불편했던 기존의 많은 응용 업무들을 효과적으로 대체할 수 있을 것이다.

일반 정보 시스템 구축에서와 마찬가지로 하이퍼미디어 어플리케이션에서도 개발 방법론이 중요하다. 여러 연구자들이 하이퍼미디어 응용 업무 개발에 관한 방법론을 제안하고 있다. 이들 방법론의 요약 비교 분석이 〈표 1〉에 제시되어 있다.

RMM (Relationship Management Methodology) [18]은 개체-관계를 이용한 방법론으로서 노드 (Node) 및 링크 (Link)등과 같은 필수적인 개념들을 처음 제시한 HDM (Hypermedia Design Methodology) [12]이라는 공식적인 하이퍼미디어 모델에 기반을 두고 있다. 인덱스 (Indexed) 및 가이드 (Guided Tour) 방식의 액세스 구조 (Access Structure)를 이용하여 HDM을 보다 향상 시켰으며 7단계의 개발 과정으로 이루어져 있다. EORM (Enhanced Object-Relationship Model) [23]은 하이퍼미디어 응용 분야에 객체 지향 설계 방법론을 최초로 적용한 기법으로서 객체 클래스를 파악하는 클래스 프레임워크(Class Framework), 클래스간 관계를 식별하는 구성 (Composition) 프레임워크 및 그래픽 사용자 인터페이스 (GUI) 프레임워크로 이루어진다. OOHDM (OO Hypermedia Design Method) [30]은 객체 지향 모델을 기초로 하며 4개의 개발단계가 상호 보완되면서 점진적 형태로 또는 프로토타입을 구축해 가는 방식으로 개발이 진행되는 방법론이다. 특히, 추상 인터페이스 설계

〈표 1〉 하이퍼 미디어 어플리케이션 개발 방법론 비교 분석

| 방법론 (저자 및 연도) / 기준 | EORM[22] (Lange 1993) | RMM[18] (Isakowitz et al. 1995) | OOHDM[30] (Schwabe et al. 1995) | VHDM[25] (Lee et al. 1996a) | SOHDM[1][26] (Lee et al. 1996b) |
|---|---|---|---|---|---|
| 기반하이퍼미디어 모델 | OO | E-R | OO | E-R | OO |
| 개발단계 | 1. Class Framework<br>2. Composition Framework<br>3. GUI Framework | 1. E-R Design<br>2. Slice Design<br>3. Navigational Design<br>4. Conversion Protocol Design<br>5. UI screen Design<br>6. Run-tim behavior Design<br>7. Construction | 1. Conceptual Design<br>2. Navigational Design<br>3. Abstract Interface Design<br>4. Implementation | 1. Requirement Analysis<br>2. E-R Design<br>3. View Design<br>4. Navigational Design<br>5. Mapping<br>6. Implementation | 1. Domain Analysis<br>2. OO Modeling<br>3. View Design<br>4. Navigational Design<br>5. Implementation Design<br>6. Construction |
| 설계산출물 | Class Structure<br>GUI Design | E-R Diagram<br>Slice Diagram<br>RMDM Diagram | OMT's OO Model<br>Navigational Class<br>Abstract Interface Design Model | E-R Schema<br>View Schema<br>Navigational Schema. | Scenario Sets<br>CRC Card<br>Class Diagram<br>OO View<br>Navigation Model<br>Page Structure<br>User Interface |
| 네비게이션의 기준 | OO Relationship | E-R Relationship | OO Relationship | E-R Relationship | Scenario and OO Views |
| 사용자 요구사항 파악방법 | 없음 | 없음 | View | View | Scenario and OO Views |
| 의미표현의 충실성 | 충실 | 미흡 | 충실 | 미흡 | 충실 |

(Abstract Interface Design) 단계를 두어 하이퍼미디어 시스템의 개발에 필요한 세부적인 명세를 구현 이전에 구체적으로 정의한다. VHDM (View-based Hypermedia Design Methodology) [25]은 개체-관계 모델을 이용하며 뷰 개념을 도입하여 서로 다른 정보를 요구하는 다양한 사용자들을 뷰를 통해 쉽고도 융통성 있게 제공할 수 있는 방법론이다.

기존에 연구된 상기한 방법론들의 제약 점을 다음과 같이 요약할 수 있다. 첫째, 개념적 데이타베이스 설계 단계가 비교적 단순하고 하이퍼미디어 어플리케이션과의 상관관계에 대한 인식이 미흡하다. 둘째, 네비게이션 형태들이 데이타베이스 스키마에 의해서만 설계되어 있다. 그러나 프로세스 관련의 정보도 유용하게 활용될 수 있을 것으로 판단된다. 세째, 이전의 객체 지향 접근 방법에 있어서는 논리적 단계 보다도 구현의 물리적 요소에 보다 강조점을 두고 있다고 할 수 있다. 하지만 객체 지향 기법의 우수한 표현력은 개념적 설계 단계에서 보다 큰 위력이 발휘될 것이다.

본 논문에서는 이상의 기존 방법들의 제약 점을 개선 및 보완하여 SOHDM (Scenario-based Object-Oriented Hypermedia Development Methodology)을 제안한다 (SOHDM의 기본 개념은 [1] [26]에서 소개 되었다). SOHDM의 주요

특성은 다음과 같다. 첫째, SOHDM은 하이퍼미디어 어플리케이션 시스템 개발 수명 주기 전반을 지원한다. 개발자 측면에서 볼 때 개발 수명 주기 전체를 지원한다는 것이 매우 중요하다. 둘째, SOHDM은 시나리오를 이용해서 개발 초기 단계부터 하이퍼미디어 어플리케이션의 요구 사항을 파악하고자 한다. 따라서 보다 유용한 많은 정보들을 초기에 획득할 수 있다. 끝으로 객체 모델링 기법과 객체 뷰의 개념을 적용하므로 실 세계에 대한 표현 능력이 훨씬 향상된다. 특히, 객체 뷰는 데이타베이스의 사용자 뷰로서 뿐 아니라 네비게이션 단위로서 활용된다.

## II. 시나리오 기반 객체 지향 하이퍼미디어 개발 방법

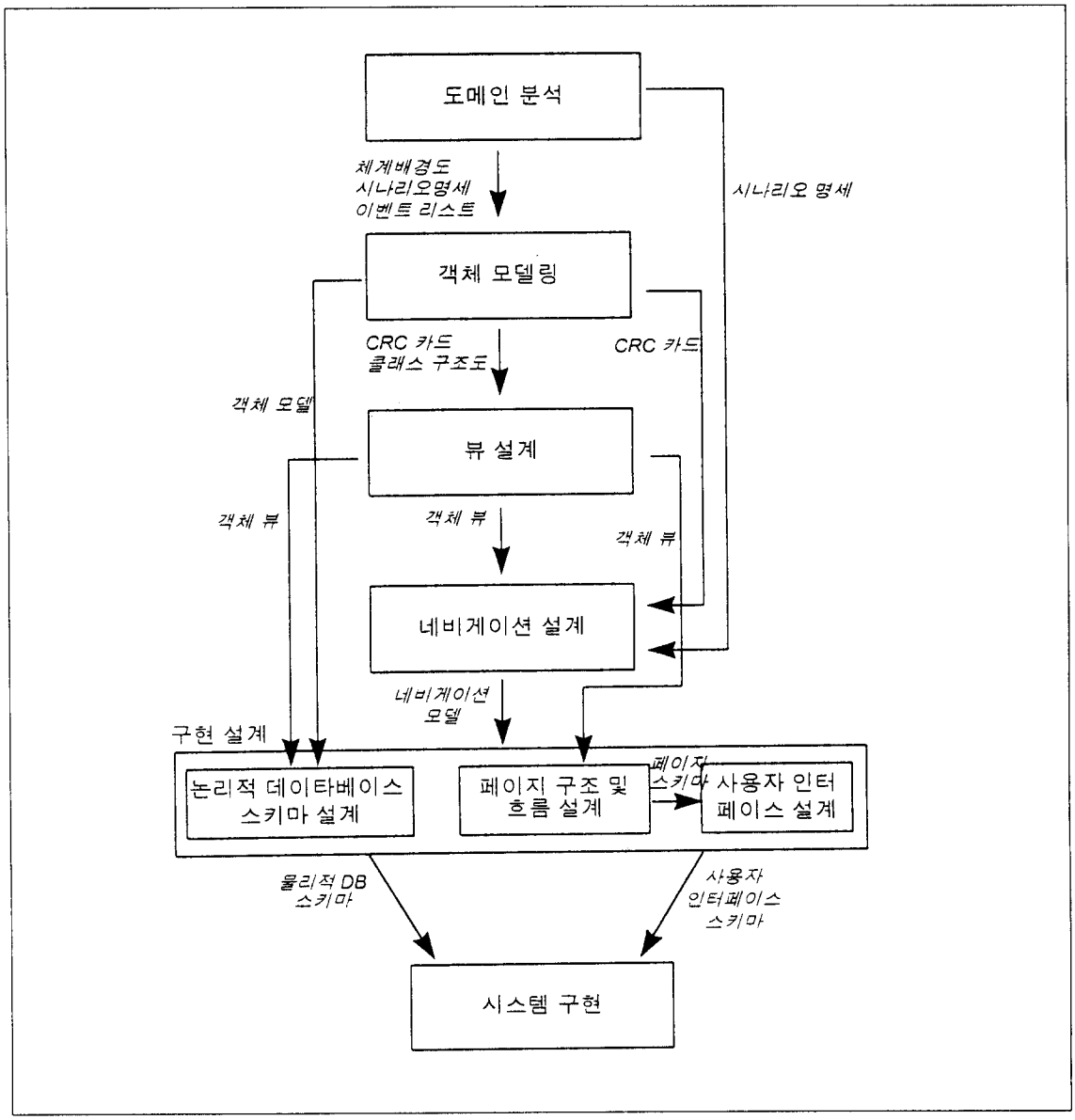
SOHDM의 기본 아키텍쳐는 〈그림 1〉에 나타나 있다. 그림에서 보듯이 SOHDM은 도메인 분석 (Domain Analysis), 객체 모델링 (Object Modeling), 뷰 설계 (View Design), 네비게이션 설계 (Navigational Design), 구현 설계 (Implementation Design) 및 시스템 구현 (Implementation)의 6단계로 이루어진다. 〈그림 1〉에는 표시되지 않았지만 각 단계에서 상위 단계로의 피드백이 존재하며 이를 통해 단계별 산출물의 품질을 제고한다.

각 단계별 개략적 내용을 설명하면 아래와 같으며 보다 상세한 내용은 3장에서 설명된다. 도메인 분석에서는 개발 대상 업무에 대한 관련 자료 조사 및 비용 대 효과를 중심으로 한 개발 타당성 분석을 통해 개발 시스템의 범위를 설정하고 업무 수행 프로세스를 파악한다. 업무 양식 및 컴퓨터로 작업 중인 업무의 스크린 폼과 각종 업무 지침 외에 현업 담당자와의 인터뷰도 업무 파악을 위한 중요한 활동이다.

먼저 시스템과 시스템의 외부를 구분 짓는 체계 배경도 (Context Diagram) [37]를 DFD (Data Flow Diagram)를 이용하여 작성한다. 체계 배경도를 작성한 후 시스템 외부 또는 시스템 자체로부터 시스템을 기동 시키는 이벤트를 파악한다. 더불어 각 이벤트별로 사용자의 요구 사항과 네비게이션 요구를 파악하기 위해 업무의 순서를 기술하는 명세로서 시나리오 (Scenario)를 만든다. 시나리오의 기술은 SAC (Scenario Activities Chart)를 이용하는데 이는 BPR에서 비즈니스 프로세스를 모형화 하기 위해 사용하는 EPC (Event-Process-Chain) [20]를 변형하여 시나리오 표현에 맞게 재구성한 표현 도구이다.

객체 모델링에서는 도메인 분석에서 만들어진 시나리오에 대한 상세 분석을 통해 시나리오에서 핵심이 되는 객체를 파악한다. 시나리오에서 액터 [19]는 대표적인 객체 후보가 되며 시나리오상의 액티비티를 중심으로 유용한 객체들을 발견할 수 있다. Actor와 시스템과의 인터액션을 통해 시스템의 동작을 보여줄 수 있으며 이때 Actor는 시스템을 이용하는 외부 즉, 이벤트의 발생처 이다. 파악된 객체가 담당하는 역할 (Roles)을 기초로 객체의 데이타 속성 (Data Aspects)과 행위적 (Behavioral Aspects) 요소인 연산 및 객체간의 관계 구조 (Relationship Structure)를 정의한다. Taylor [34]의 기법을 기초로 하며 CRC (Class Responsibilities Collaborators) 카드를 이용하여 객체 모델을 명세화하고 동적인 객체를 추출할 수 있도록 RDD (Responsibilities Driven Design) [16] 방식이 접목되었다.

뷰 (View) 설계단계에서는 사용자 관점의 단위 정보로서 관계형 데이타베이스의 논리적 사용자 뷰와 동일한 의미의 뷰를 정의한다. SOHDM에서는 객체 모델을 바탕으로 한 객체 뷰를 유도하여 이용한다. 객체 모델로부터 다양한 조합의 뷰가 유도될 수 있으나 사용자 요구 사항을 충실




〈그림 1〉 SOHDM 방법

히 반영하는 뷰를 정의하는 것이 중요하다. 더불어 하이퍼미디어 어플리케이션에서는 뷰의 내용을 통해 또 다른 정보를 찾아 갈 수 있는 패스(Path)가 존재하는 뷰가 주목된다. 특히, 하이퍼미디어 어플리케이션에서는 뷰가 한 단위 정보로부터 다시 찾아 갈 수 있는 정보가 되는 네비게이션 단위 (Navigation Unit)로 쓰이게 된다.

네비게이션 설계단계에서는 사용자가 하이퍼미디어 응용 업무의 한 부분에서 다른 부분으로 정보를 찾아갈 수 있도록 하는 액세스 구조를 제공한다. 액세스 구조 및 패스를 포함하고 있는 액세스 구조 노드 (Access Structure Node, ASN)와

앞 단계에서 유도된 객체 뷰가 네비게이션 단위로 활용된다. 또한, 개발자는 하이퍼미디어 정보를 추적해 나가는 네비게이션 패스로서 링크를 설계한다. 링크 설계를 위해 클래스 구조도(Class Structure Diagram)상의 클래스간 상호 관계가 분석되며 시나리오를 검토하게 된다. 이때 시나리오에서의 액티비티의 형태에 따라 메뉴 형태, 하향식 접근, 검색 및 색인 방식(Glossary)과 같은 사용자 인터페이스 전략을 수립한다. 링크는 접근 경로에 있어 출발점(Source)과 목적지(Target)간의 관계로 나타내며 양자 모두 객체 뷰나 액세스 구조 노드가 될 수 있다.

구현 설계단계에서는 실제의 동작 환경 및 개발 도구를 고려하여 구현에 필요한 제 요소를 상세하게 결정한다. 구현 설계 단계는 크게 논리적 데이타베이스 설계, 페이지 구조 및 흐름 설계, 사용자 인터페이스 설계의 3가지 하위 단계로 이루어 진다. 먼저 정해진 물리적 조건에 의거 시나리오를 기초로 하여 단위 정보로부터 다음의 단위 정보를 찾아가는 일련의 순환과정을 페이지 구조 (Page Structure)와 이의 흐름으로 정의하게 된다. 다음은 사용자가 시스템을 사용하기 위해 만나게 되는 인터페이스를 구체적으로 정한다. 사용자 인터페이스 설계는 페이지 스키마를 이용하여 이를 Java [3], HTML 등과 같은 개발 도구에서 제공하는 선택 박스(Choice Box), 버튼(Button) 등과 같은 사용자 인터페이스 요소들로 매핑하게 된다. 이때 설계자의 경험과 취향에 따라 다양한 요소들이 활용될 수 있으나 가능하다면 표준적인 기준을 정하여 설계 요소를 구성하는 것이 시스템 전체의 통일성뿐 아니라 사후 유지 보수에도 용이하게 된다. 또한 객체 지향 모델로 설계된 클래스 구조도를 물리적인 개발 환경에 속하는 데이타베이스 관리 시스템 구조에 맞춰 매핑한다. 본 연구에서는 관계형 데이타베이스 시스템 환경으로 매핑하기 위한 방안을 이용한다.

마지막으로 시스템 구현 단계에서는 개발 및 운영 환경상의 제 요소를 고려하여 실제 동작하는 하이퍼미디어 어플리케이션 소프트웨어를 구현한다. 데이타베이스 관리 시스템에서 제공하는 데이타 정의 언어(DDL)를 이용하여 데이타베이스 스키마를 생성한다. 한편, 페이지 구조와 흐름을 명세화 한 페이지 스키마와 입/출력 인터페이스 설계도를 이용하여 Java언어로 어플리케이션 프로그램을 개발한다. 시험 및 보완 작업을 반복적으로 거치면서 전체 시스템 개발을 완료하게 된다.

## III. 인트라넷 하이퍼미디어 시스템 개발

### 3.1 개발 대상 및 범위

ㄷ 은행은 27년의 역사를 가진 지방 은행으로서 현 정보 시스템 환경은 계정계는 IBM 주장비를, 정보계는 워크스테이션을 기반으로 한 클라이언트/서버 시스템을 구축하여 활용 중에 있다. 인터넷 기술을 사내 정보 시스템 구축 기술로 적용코자 하는 인트라넷에 대한 지속적인 관심이 조직 내에 상존하고 있다. 이와 같은 수요에 부응하여 기존의 클라이언트/서버 시스템을 기반으로 인트라넷 활용 방안을 분석 시도하기 위해 KAIST 연구 팀이 구성되었다.

우선, 대상 업무는 금융 계통 분야의 전형적인 업무로서 은행의 정보계 업무를 정하고, 이들 중 구좌가 존재하는 거래 고객과 고객 유치 차원에서 은행에서 접촉하는 섭외 고객 관리로 정했다. 또한, 대 고객 서비스 업무로서 은행의 적금 등 상품 정보 제공 및 고객 계좌에 대한 잔고 확인을 포함했다. 경영층을 위한 정보로서 하이퍼미디

어 어플리케이션 개발의 효과를 가시화할 수 있는 예로서 은행의 계정과목 및 점포별 수신고를 조회할 수 있는 기능을 선정했다. 3개 분야에 대한 세부 내용은 다음과 같다.

(i) 고객 관리:

고객은 현재 은행에 계좌를 가지고 거래하는 거래 고객과 고객을 유치하고자 하는 차원에서 고객의 재산 상태 및 신용 평가를 기초로 고객 정보를 관리하고자 하는 섭외 고객으로 나눌 수 있다. 각 고객에 대해 고객 코드별 조회, 고객 명부 조회 후 세부 사항을 검색하는 기능과 신용 평가 자료를 이용한 우수 고객 조회 및 각급 조건을 부여하여 고객을 조회하는 기능이 있다.

(ii) 고객 지원:

은행에서 고객에게 유용한 기능을 제공하는 대고객 지원 서비스로서 금융 상품 안내가 있다. 즉, 은행에서 취급하는 적금 등에 관해 고객에게 알려 준다. 또한, 계정계 업무이기는 하지만 고객 지원 기능으로서 계좌의 잔고 상태를 확인할 수 있는 기능을 제공한다.

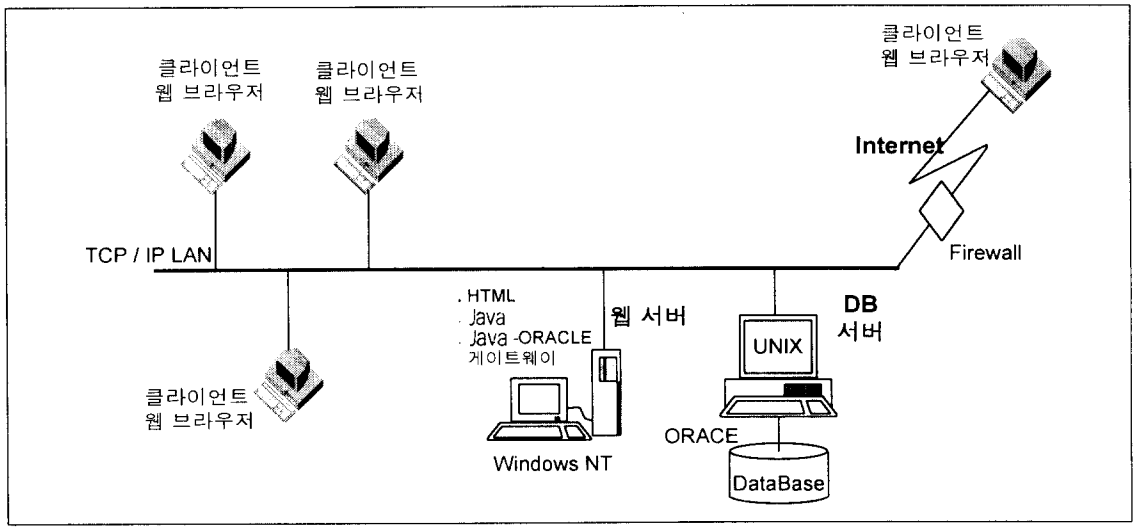
(iii) 경영정보:

은행의 중요 지표가 되는 수신고 현황 정보를 경영정보로서 제공한다. 계정과목별, 점포별, 지점별로 전일 대비 금일에 대한 수신고 정보를 알 수 있도록 한다.

끝으로 개발 단계 및 대상 업무에 대한 기준을 기초로 하이퍼미디어 어플리케이션의 장점과 효과를 가시화할 수 있는 내용에 개발 중점을 두었다.

### 3.2 구현 컴퓨팅 환경

하이퍼미디어 어플리케이션 구현을 위해 사용된 컴퓨팅 환경 및 도구는 〈그림 2〉에서 보는 것과 같다. 특히, Java언어를 사용하고 관계형 DBMS와의 연계를 위해 OCI (Oracle Call Interface) 인터페이스를 사용했으며 모두가 현재 최신 기술로 베타 버전 상태에서 적용되었다. 〈그림 2〉에서 보는 바와 같이 DB서버, Web 서버 및 클라이언트의 웹 브라우저 (Web Browser)를 분리한 3계층 (Tier)으로 아키텍처가 구현되었다.




〈그림 2〉 개발 컴퓨팅 환경

DB 서버에는 Sun/Sparc-20를 이용하고 ORACLE DBMS [27]가 탑재되었다. Web 서버로는 펜티엄 급 PC에 Windows/NT 운영체계를 활용하였다. 웹서버에는 HTML문서, Java언어 및 ORACLE DBMS와 Java언어간의 인터페이스를 위한 게이트웨이가 설치되었다.

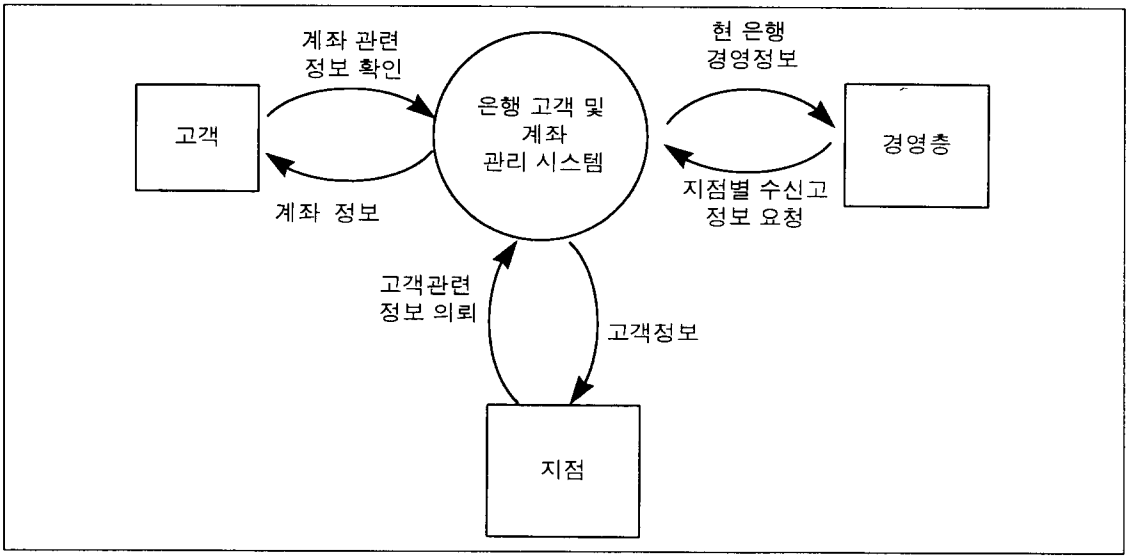
### 3.3 개발 절차

#### 3.3.1 도메인 분석

〈그림 3〉은 도메인 분석 결과 작성된 체계 배경도로서 개발 대상 범위로 정한 부분에 대해 외부와 시스템간의 경계를 보여준다. 고객 및 계좌 관리 시스템의 외부는 고객, 지점 및 경영층이다. 특별히, 경영정보 제공을 위해 경영층과 이와 관련된 자료 흐름을 포함했다.

체계 배경도를 근간으로 시스템을 기동 시키는 역할을 하는 이벤트를 파악하여 다음 〈표 2〉와 같이 이벤트 리스트를 정리했다. 총 4개의 이벤트가 추출되었다. 이외에도 발생처가 시스템 외부인 이벤트도 가능하다. 예로서 재 보고서를 산출하는 주기라는 이벤트를 들 수 있다. 각 이벤트에 대해 시나리오를 작성하게 되는데 본 논문에서는 사례로서 이벤트 1, 2, 3을 선정했다. 다음 〈그림 4〉는 시나리오 작성에 사용되는 표현 기호이다. 시나리오의 처음에 나타나는 이벤트는 둥근 사각형으로, 시나리오상에서의 세부 액티비티는 직사각형으로 나타낸다. 그리고 화살표는 일련의 작업이 시간에 따라 진행되는 흐름을 표시하며, 조건에 따라 분기되는 선택에 대해서는 마름모로 그리고 작업의 종료는 두개로 된 원으로 나타낸다.

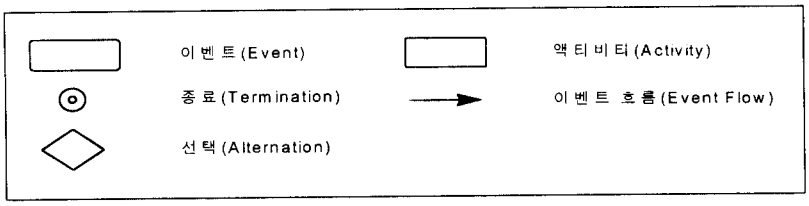
〈그림 5〉는 계좌 상태 확인 이벤트에 의한 시나리오를 보여준다. 고객이 자신의 계좌에 대한 잔고 및 트랜잭션 데이타를 요구할 경우 이에 대한 자료를 제공한다. 〈그림 6〉은 고객 관련 정보 의뢰 이벤트에 의한 시나리오이다. 고객을 거래 고객과 섭외 고객으로 분류하고 고객 형태별로 필요한 정보를 산출하여 제공한다. 같은 형식으로 〈그림 7〉은 지점 또는 경영층에서 지점별 수신고



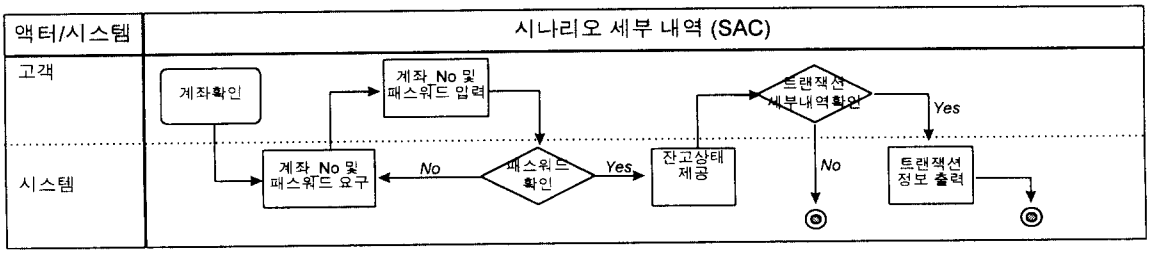

〈그림 3〉 체계 배경도

〈표 2〉 이벤트 리스트

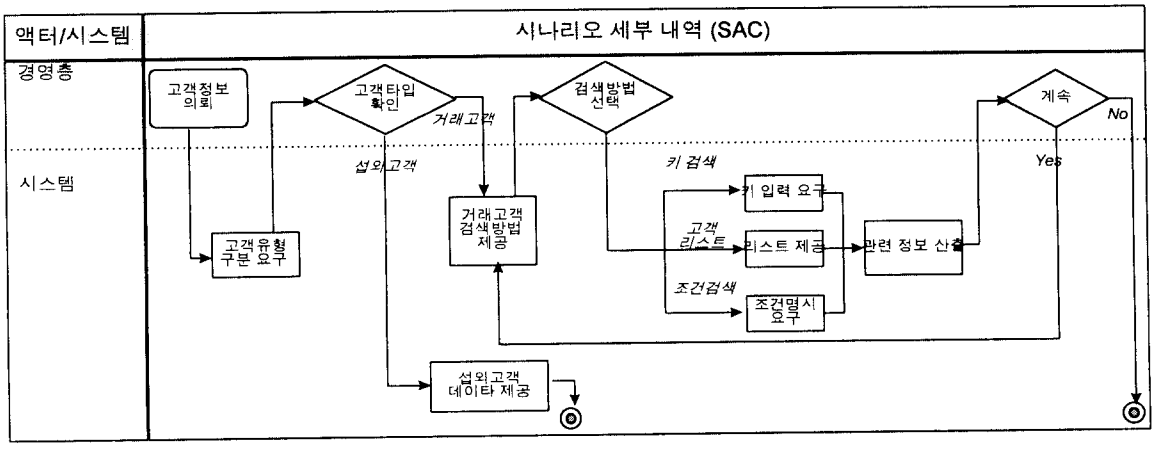
| no | 이벤트 명 | 발생처 |
|---|---|---|
| 1 | 계좌 상태 확인 | 고객 |
| 2 | 고객 관련 정보 의뢰 | 경영층 |
| 3 | 지점별 수신고 확인 요구 | 지점/경영층 |
| 4 | 은행 금융 상품 문의 | 고객 |



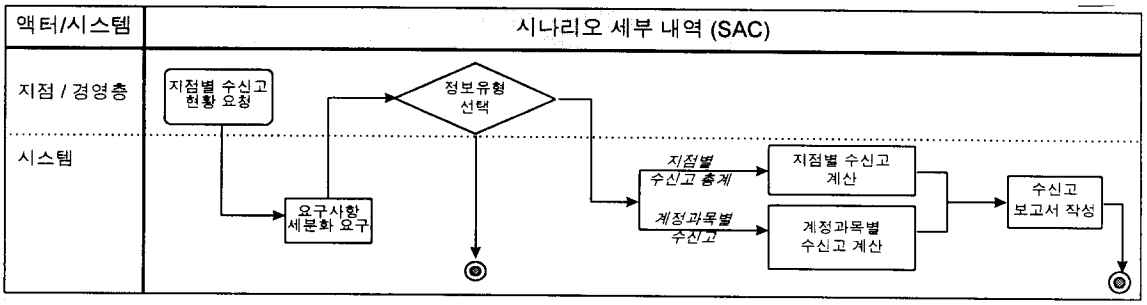

〈그림 4〉 시나리오 표현 기호




〈그림 5〉 계좌 상태 확인 시나리오




〈그림 6〉 고객 관련 정보 의뢰 시나리오



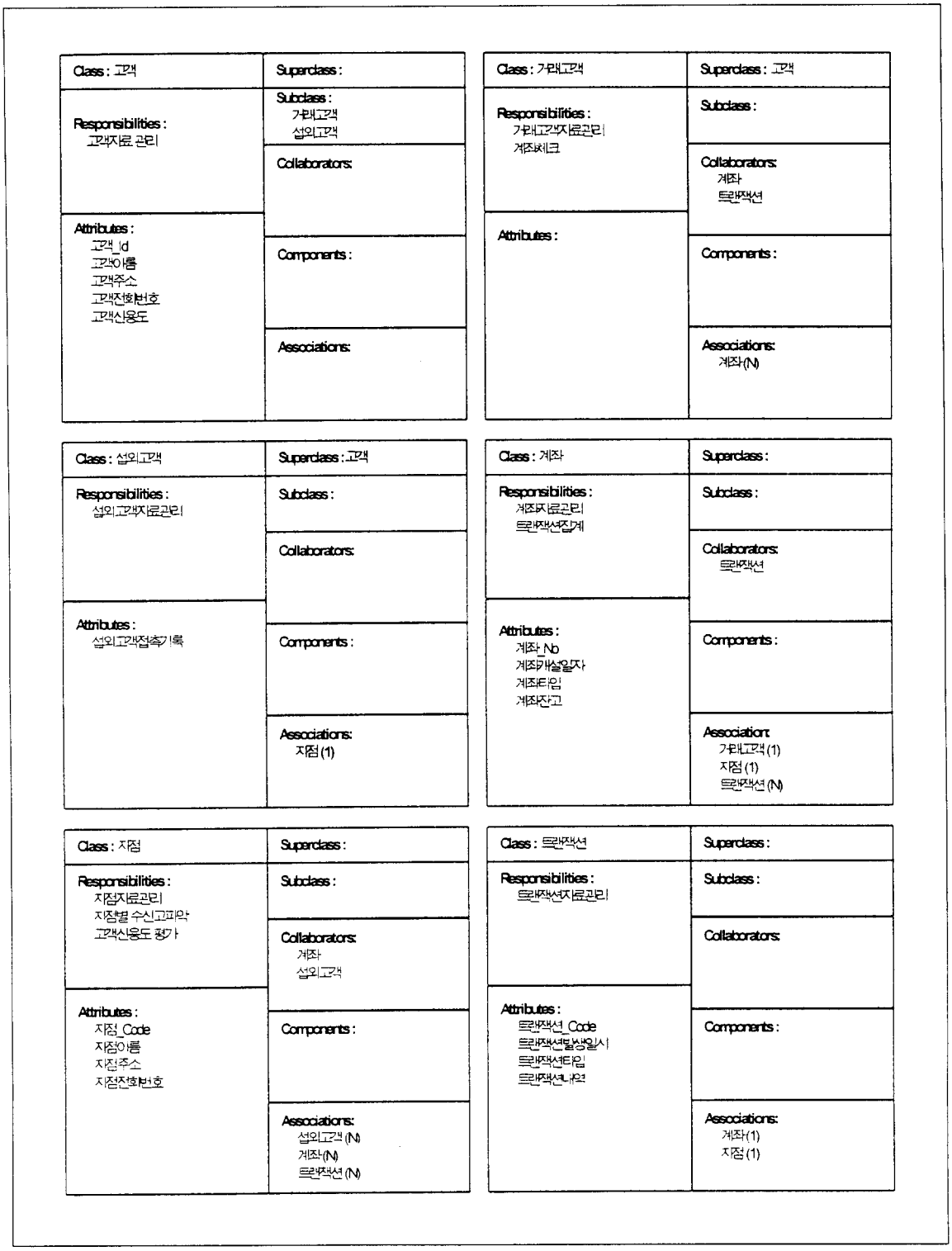

〈그림 7〉 지점별 수신고 확인 요구 시나리오

현황을 파악하고자 할 경우 처리되는 경영정보 제공 시나리오이다. 시나리오에서 액터는 이벤트의 발생처이며 동시에 업무의 세부 액티비티를 수행하는 주체이다.

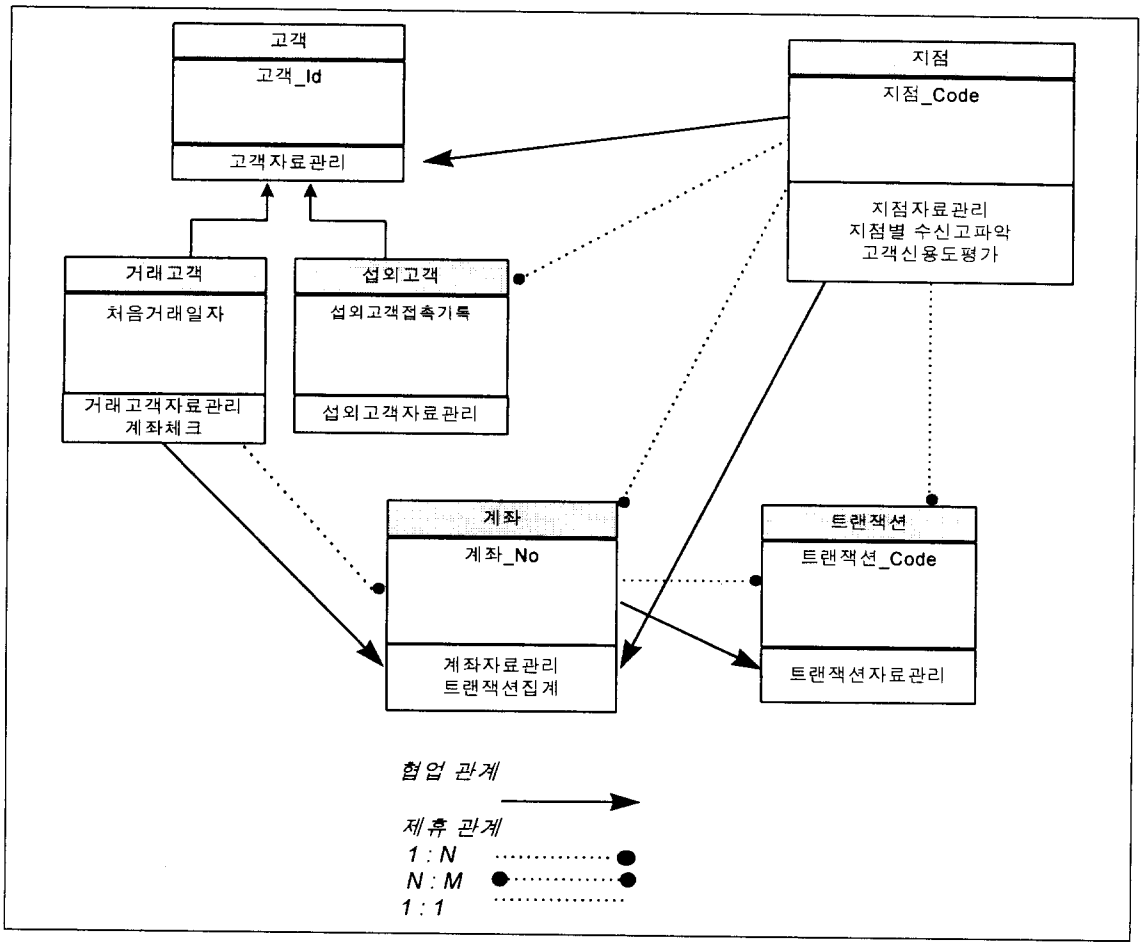
3.3.2 객체 모델링

앞의 〈그림 5〉, 〈그림 6〉 및 〈그림 7〉의 시나리오를 분석하여 객체 모델링을 실시한다. 우선 액터는 1차적으로 객체 후보가 되며 시나리오 상의 액티비티를 중심으로 객체 대상을 발견할 수 있다. 분석 결과 거래 고객, 섭외 고객, 계좌, 지점 및 트랜잭션과 같은 주요 객체를 파악할 수 있었다. 이외에 고객 객체는 거래 고객과 섭외 고객의 공통 요소를 보유한 수퍼 타입으로서 일반화 관계에 따라 추상화한 결과이다. 〈그림 8〉은 이들 객체를 클래스로 정의하여 CRC카드로 명세화 한 것이다. 본 CRC카드는 기존 Wirfs-Brock et al.[36]이 제안한 표현을 본 연구에 맞도록 속성 항목과 제휴 관계(Association)를 추가 보완한 형식을 취하고 있다. 또한 제휴 관계에서 대응하는 클래스의 인스턴스가 발생될 수 있는 최대 수를 카디낼리티 (Cardinality)로 나타내며 제휴 관계의 클래스 옆에 괄호 안의 숫자로 명시한다. 거래 고객 클래스 카드를 예를 들어 설명하면 다음과 같다. 거래 고객의 수퍼 클래스는 고객이며 서브 클래스는 존재하지 않는다. 거래 고객 클래스는 거래 고객 자료의 관리라는 기본적인 책임 외에 계좌의 잔고 상태를 확인하는 책임을 가진다. 따라서 계좌가 협업 클래스가 된다. 또한, 수퍼 클래스인 고객에 존재하는 모든 속성 항목을 계승받으며 최초 거래 일자가 추가적으로 정의된다. 끝으로 거래 고객의 부품을 이루는 클래스가 없으므로 Components 항목은 빈 칸이 된다.

〈그림 9〉는 클래스간의 관계 구조를 다이어그램으로 기술한 클래스 구조도 이다. 클래스 구조도는 클래스간의 관계를 외부적으로 나타내기 위해 사용된 다이어그램이다. 클래스간의 관계는 클래스 카드의 Superclass / Subclass, Components 및 Collaborators 3종류를 기본으로 한다. 거래 고객은 자신의 계좌의 잔고 상태를 확인하기 위해 계좌 클래스와의 협업 관계가 필요하며 거래 고객의 클래스 카드에 계좌 클래스가 협업 대상으로서 명시되었다. 그러나 어떤 클래스와 클래스간에 부품과 완성품으로서의 관계를 나타내는 Components는 본 예에 제시되어 있지 않다. 〈그림 9〉에서 화살표는 협업 관계를 표현하며 점선은 제휴 관계를 나타낸다. 제휴 관계는 카디낼리티에 따라 1:1 (일대일), 1:N (일대다) 및 M:N (다대다)의 관계가 있으며 본 예에는 1:N의 관


Class : 고객
Superclass :
Responsibilities :
고객자료 관리
Subclass :
거래고객
설외고객
Collaborators:
Components :
Attributes :
고객_Id
고객이름
고객주소
고객전화번호
고객신용도
Associations:
Class : 거래고객
Superclass : 고객
Responsibilities :
거래고객자료관리
계좌체크
Subclass :
Collaborators:
계좌
트랜잭션
Attributes :
Components :
Associations:
계좌(N)
Class : 설외고객
Superclass : 고객
Responsibilities :
설외고객자료관리
Subclass :
Collaborators:
Components :
Attributes :
설외고객접촉기록
Associations:
지점 (1)
Class : 계좌
Superclass :
Responsibilities :
계좌자료관리
트랜잭션집계
Subclass :
Collaborators:
트랜잭션
Components :
Attributes :
계좌_No
계좌개설일자
계좌타입
계좌잔고
Association:
거래고객 (1)
지점 (1)
트랜잭션 (N)
Class : 지점
Superclass :
Responsibilities :
지점자료관리
지점별 수신고파악
고객신용도 평가
Subclass :
Collaborators:
계좌
설외고객
Attributes :
지점_Code
지점이름
지점주소
지점전화번호
Components :
Associations:
설외고객 (N)
계좌(N)
트랜잭션 (N)
Class : 트랜잭션
Superclass :
Responsibilities :
트랜잭션자료관리
Subclass :
Collaborators:
Attributes :
트랜잭션_Code
트랜잭션발생일시
트랜잭션타입
트랜잭션내역
Components :
Associations:
계좌(1)
지점 (1)

〈그림 8〉 CRC 카드



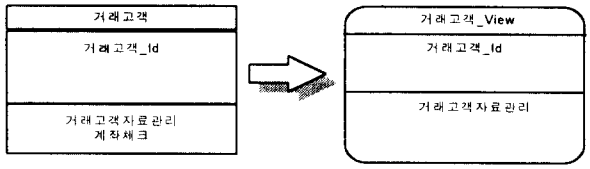

〈그림 9〉 클래스 구조도

계만이 제시되어 있다. 그림에서 계좌는 트랜재션 클래스와 협업 관계를 가지며 거래 고객, 트랜잭션 및 지점 클래스와 제휴 관계가 있음을 보여준다. 또한, 상위 단계에서는 클래스 카드 상의 책임이 클래스의 메쏘드로 변환된다. 메쏘드에는 기본적으로 자신의 내부 자료를 관리하는 메쏘드와 클래스의 책임에 의거 추가적인 기능을 수행하는 메쏘드가 있다.

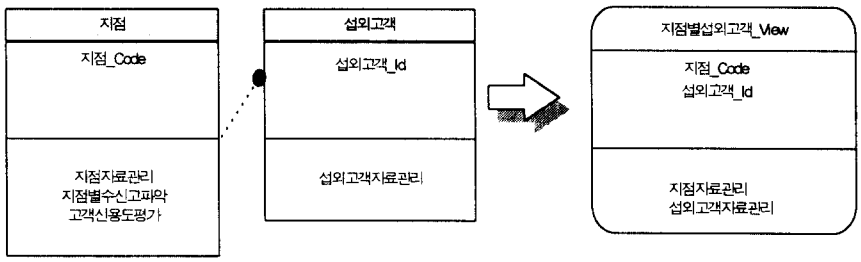
3.3.3 뷰 설계

뷰는 사용자가 바라보는 정보의 단위이다. 여기에서는 하이퍼미디어 어플리케이션에서 데이타베이스로부터 동적으로 교류되는 정보 내용이 객체 뷰로 설계된다. 특히, 뷰를 통해 또 다른 뷰로 가거나 새로운 정보를 찾을 수 있는 통로를 갖는 뷰가 네비게이션 설계단계에서 중요하게 쓰인다. 뷰는 클래스 구조도를 바탕으로 유도되며 한 개의 클래스로부터, 한 클래스로부터 다른 클래스를 참조하는 제휴 관계에 대해 그리고 클래스간 협업 관계에 의해 유도되는 3가지 유형이 있다.

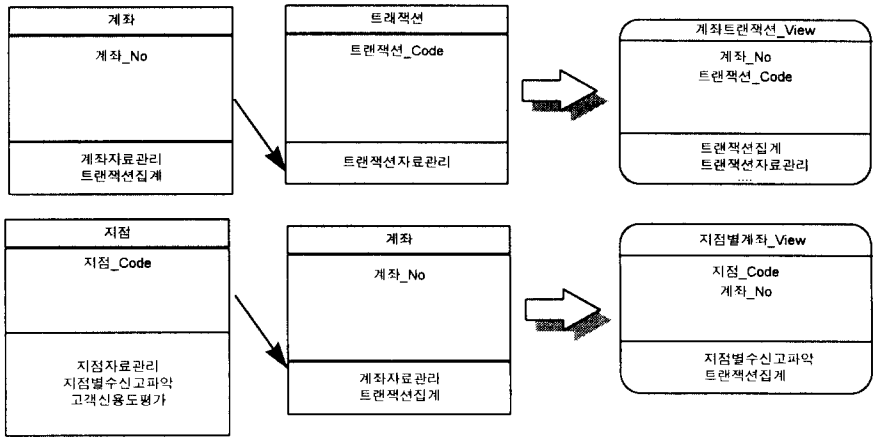
〈그림 10〉 (a)의 고객의 뷰는 단순히 거래 고객 클래스만으로 구성된 베이스 뷰이다. 거래 고객에 관하여 필요한 정보를 거래 고객 클래스의




(a) 베이스 뷰




(b) 제휴 관계에 의한 뷰




(c) 협업 관계에 의한 뷰

〈그림 10〉 뷰 설계 예

속성으로부터 가져 오고 필요한 메쏘드를 이용한다. 거래고객_View에서는 거래 고객 자료 관리 메쏘드가 뷰에 포함된다. (b)는 지점에서 관리하는 섭외 고객과 참조 링크로 연결되는 제휴 관계에 의한 뷰이다. 제휴 관계 뷰에서는 제휴 관계에 참여하고 있는 각 클래스로부터 필요한 모든 속성 항목을 가져오고 참조에 필요한 메쏘드를 활용한다. 예에서 지점별섭외고객_View는 지점에 관한 기본 자료와 해당 지점에서 접촉한 모든 섭외 고객의 정보를 함께 이용하며 제휴에 필요한 메쏘드를 추가하여 뷰를 구성했다. (c)의 계좌트랜잭션_View는 계좌와 트랜잭션간의 협업 관계를 이용하여 계좌에 남아 있는 잔고 상태 계산 및 트랜잭션 내역을 확인하는 협업 관계 뷰이다. 마찬가지로 지점별계좌_View도 지점별 수신고 현황 파악을 위한 협업 관계 뷰로서 본 연구의

프로토타입 개발에 사용되는 뷰이다. 계좌트랜잭션_View에서 계좌의 잔고 상태를 확인하기 위해 계좌 클래스는 트랜잭션 집계 및 관리하는 메쏘드를 활용하여 계좌에서 발생한 모든 트랜잭션을 계산하게 된다.

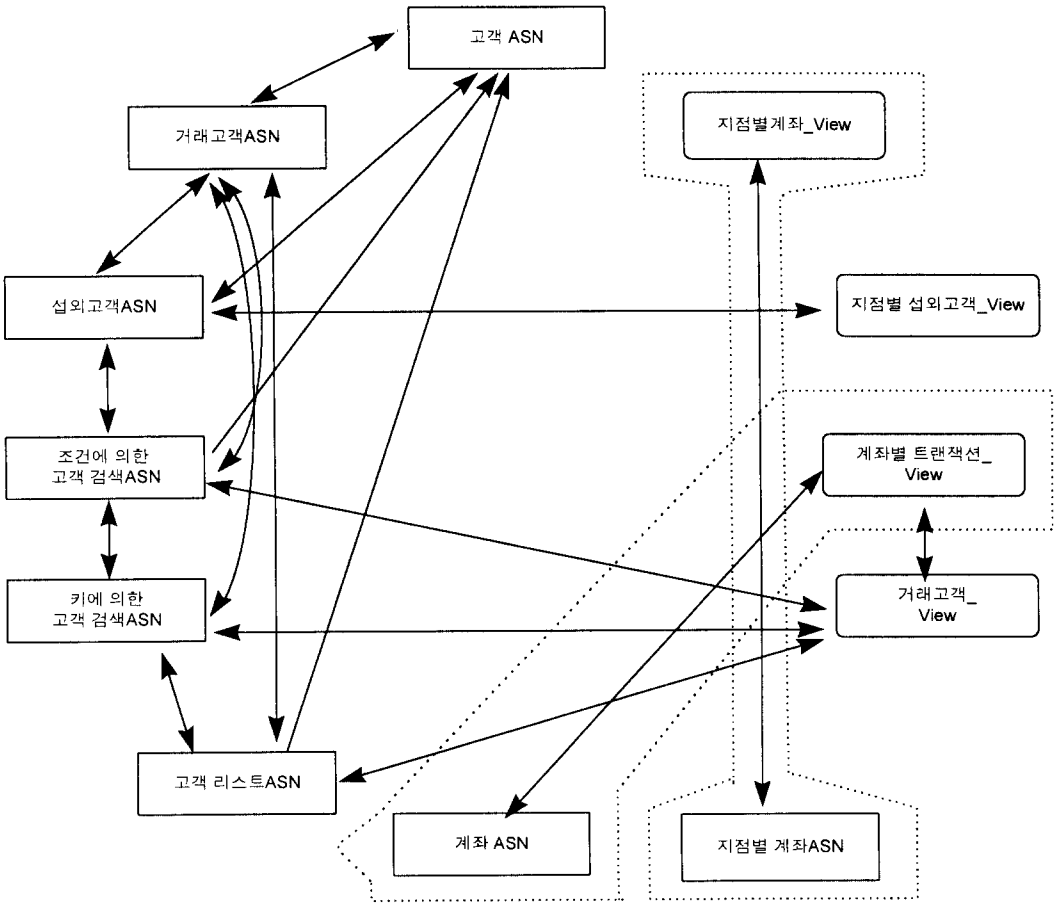
(b)와 (c)의 뷰는 모두 메쏘드를 가지나 뷰에 포함되는 메쏘드의 형태가 서로 다르다. 제휴 관계에서는 제휴하고 있는 각 클래스의 내부 자료를 관리하는 메쏘드가 포함되며 협업 관계에서는 협업을 위해 요구되는 메쏘드가 선택된다. 표기 목적상 그림에서는 각 뷰에 대해 키가 되는 애트리뷰트만을 나타냈으나 실제로는 사용자의 요구사항에 따른 제반 애트리뷰트들도 뷰에 상세하게 명시된다.

### 3.3.4 네비게이션 설계

네비게이션 설계 시 기존 방법에서는 데이타만을 위주로 고려하는 경향이 있다. SOHDM에서는 ASN과 뷰가 네비게이션 단위가 된다. 시나리오로부터 네비게이션에 대한 유용한 정보를 찾아 이를 데이타 모델과 결합하여 보다 양질의 결과를 얻도록 시도한다. 먼저 시나리오 분석을 통해 사용자들이 자신의 업무를 수행하면서 어떠한 방식으로 정보를 찾아 나가고 활용하는 지를 파악한다. 다음에는 이를 메뉴, 하향식, 검색 및 색인에 의한 접근 방식과 같은 4가지의 사용자 인터페이스 전략으로 설정한 후 ASN으로서 정의한다. 여기에 실제 필요한 정보인 뷰를 결합하여 네비게이션을 설계한다. 네비게이션은 주로 시스템과 다른 액터와의 사이에서 발생한다는 것을 본 연구를 통해 발견할 수 있었다. ASN과 뷰의 차이점은 ASN이 정보를 찾아가는 길만을 제시해 주는데 비해 뷰는 사용자에게 실제로 유용한 정보를 나타내는 노드이다. 뷰는 실제 정보를 제공하는 외에 제공된 정보를 이용하여 새로운 정보를 찾아갈 수 있는 링크 역할도 한다.

앞의 시나리오 분석을 통해 8가지의 ASN을 찾아냈으며 이를 뷰 설계 단계에서 유도된 4가지의 뷰와 상호 결합하여 네비게이션 링크를 구성한다. 〈그림 6〉 고객 관련 정보 의뢰 이벤트에 의한 시나리오를 예로 들어 설명하면 다음과 같다. 먼저 고객의 정보가 요구될 경우 고객 유형을 거래 고객과 섭외 고객으로 구분한다. 여기에서 고객 ASN을 설정할 수 있으며 다시 이 ASN으로부터 거래 고객과 섭외 고객의 정보를 찾아갈 수 있는 거래고객ASN과 섭외고객ASN을 정의할 수 있다. 거래 고객의 정보를 원하는 경우 시스템에서는 정보의 검색 방법을 3가지로 구분하여 결정할 것을 요구한다. 이러한 시나리오를 근간으로 키 검색, 고객 리스트에 의한 검색 및 조건 검색과 같은 3 종류의 ASN을 유도할 수 있다. 이와 같은 6가지의 ASN과 거래 고객 정보를 제공하는 거래고객_View와 섭외고객_View가 노드로 이용된다.

〈그림 11〉은 거래 고객에 대한 6가지의 ASN과 뷰 외에 다른 시나리오를 이용하여 지점별 계좌 및 계좌에 대한 트랜잭션을 기초로 각각 만들어진 ASN과 뷰를 첨가하여 구조화한 네비게이션 링크이다. 그림에서 점선으로 표시된 부분은 서로 독립적인 다른 정보를 네비게이션 하는 3개의 그룹이 존재한다는 것을 나타낸다. 독립적이라는 것의 의미는 각 그룹 상호간에 서로 관계 있는 정보가 존재하지 않아 각 그룹으로부터 다른 그룹으로 정보를 찾아 갈 수 없다는 것을 뜻한다. 네비게이션 링크에서 양 방향의 화살표는 본래의 정방향 링크 외에 추가적으로 역방향의 링크를 고려했음을 표시한다. 즉, 노드간의 이동에서 바로 직전 노드로의 역방향 링크를 둠으로써 여기서부터 다시 필요한 노드로 편리하게 이동할 수 있도록 한다. 또한, 이러한 기본적인 네비게이션




〈그림 11〉 네비게이션 링크

링크를 기초로 사용의 용이성을 고려하여 새로운 링크가 추가되었다.

복잡한 링크 관계를 매트릭스로 구성하면 보다 간단하고 이해가 용이할 것이다. 다음 〈그림 12〉는 〈그림 11〉을 매트릭스로 도시하여 단순화한 것이다. 고객 관리, 지점별 계좌 및 계좌 트랜잭션과 같은 3개의 그룹의 ASN과 뷰를 기초로 이들간의 링크를 고려하여 작성된 네비게이션 매트릭스이다. 네모의 검은 부분은 상대편으로의 링크가 존재하여 찾아 갈 수 있는 패스가 존재한다는 것을 의미하며 흰 부분은 대응되는 링크가 없음을 나타낸다. 예를 들어 고객ASN에서 거래고객ASN으로 패스가 있으며 또한 그 반대 방향으로도 패스가 존재한다. 또한 고객ASN에서 키에 의한 검색ASN으로의 패스는 존재하지 않지만 키에 의한 검색ASN에서 고객ASN으로의 패스는 존재한다. 그러나 계좌ASN과 고객ASN간에는 양 방향 패스가 존재하지 않는다.

### 3.3.5 구현 설계

구현 설계단계에서는 실제의 동작 환경 및 개발 도구를 고려하여 전 단계에서 작성된 제반 산출물을 구현에 적합하게 변화시킨다. 구현 설계

| ASN / View | 고객 ASN | 거래고객 ASN | 섭외고객 ASN | 키에 의한 고객 검색ASN | 고객리스트 ASN | 조건에 의한 고객검색 ASN | 지점별계좌 ASN | 계좌 ASN | 거래고객_ View | 지점별 계좌_ View | 지점별 섭외고객_ View | 계좌별 트랜잭션_ View |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 고객 ASN | | | | | | | | | | | | |
| 거래고객ASN | ■ | | | | | | | | | | | |
| 섭외고객ASN | ■ | ■ | | | | | | | | | | |
| 키에 의한 검색ASN | ◥ | ■ | ◻ | | | | | | | | | |
| 고객 리스트ASN | ◥ | ■ | ◻ | ■ | | | | | | | | |
| 조건에 의한 고객검색ASN | ◥ | ■ | ◻ | ■ | ■ | | | | | | | |
| 지점별 계좌 ASN | ◥ | ◥ | ◻ | ■ | ■ | ■ | | | | | | |
| 계좌 ASN | ◻ | ◻ | ◻ | ◻ | ◻ | ◻ | ◻ | | | | | |
| 거래고객_ View | ◻ | ■ | ◻ | ■ | ■ | ■ | ◻ | ◻ | | | | |
| 지점별계좌_ View | ◻ | ◻ | ◻ | ◻ | ◻ | ◻ | ■ | ◻ | ◻ | | | |
| 지점별 섭외고객 _View | ◻ | ◻ | ■ | ◻ | ◻ | ◻ | ◻ | ◻ | ◻ | ◻ | | |
| 계좌별 트랜잭션 _View | ◻ | ◻ | ◻ | ◻ | ◻ | ◻ | ◻ | ◻ | ■ | ◻ | ◻ | |

◥ 첫 칼럼의 View나 ASN이 출발점이고 첫 줄의 View나 ASN이 목적지가 됨.

◤ 첫줄의 View나 ASN이 출발점이고 첫 칼럼의 View나 ASN이 목적지가 됨.

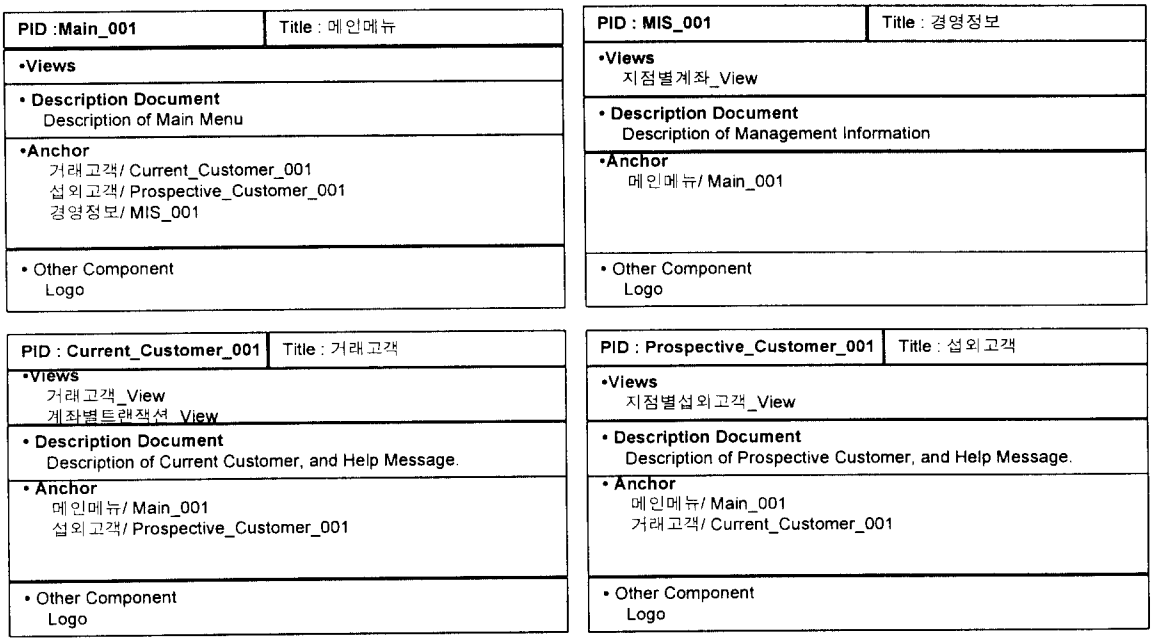
■ 첫 칼럼 및 첫 줄의 View나 ASN이 모두 출발점이되고 목적지가 됨.

〈그림 12〉 네비게이션 링크 매트릭스

단계는 논리적 데이타베이스 설계, 페이지 구조 및 흐름 설계, 사용자 인터페이스 설계와 같은 3가지 하위 단계로 구성된다.

먼저 정해진 물리적 조건에 의거 시나리오를 기초로 하여 단위 정보로부터 다음의 단위 정보를 찾아가는 일련의 순환과정을 페이지 구조와 이의 흐름으로 정의하게 된다. 페이지는 뷰를 근간으로 설정되며 특수 목적의 페이지를 추가하여 전체 페이지 스키마를 설계한다. 전 단계에서의 〈그림 12〉의 네비게이션 링크 매트릭스를 기초로 페이지 구조 및 흐름을 작성하게 된다. 〈그림 13〉은 예에서 사용된 페이지 스키마이다. 거래고객__View와 계좌트랜잭션__View가 중심이 되어 Current__Customer__001페이지가 작성되었으며, 지점별섭외고객__View를 기초로 Prospective__Customer__001페이지, 지점별계좌__View를 이용하여 MIS__001페이지가 각각 설계되었다. Main__001페이지는 시스템 전체를 총괄하기 위한 목적상 작성된 추가적인 페이지이다. 뷰를 중심으로 작성된 각 페이지에 관련 ASN이 할당되어 각 페이지간의 이동을 앵커로 표시한다.

다음은 사용자가 시스템을 사용할 때 접하는 인터페이스를 구체적으로 결정한다. 이에 앞서 시나리오 타입에 의거 사용자 인터페이스 전략[32]을 결정한다. 즉, 시나리오에서 정보를 점진적으로 상세하게 찾아가는 경우 사용자 인터페이스 전략을 하향식으로 구성하는 것이 바람직하다. 또한, 찾고자 하는 정보를 잘 모르고 제시된 안내를 따르면서 정보를 추적하는 형태에서는 메뉴 방식의 인터페이스 전략이 유용할 것이다. 아울러 키, 조건 및 리스트로부터 원하는 정보를 찾아나가는 형태의 시나리오는 사용자 인터페이스로서




〈그림 13〉 페이지 스키마

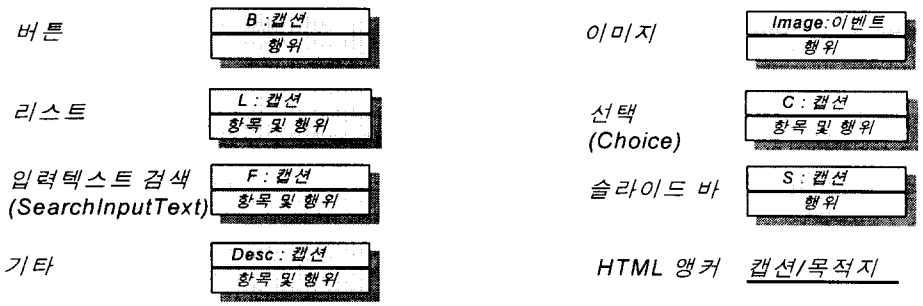
| 시나리오 형태 | 사용자 인터페이스 전략 |
|---|---|
| 점진적으로 정보를 찾아감 | 하향식 (Top-down) |
| 안내에 따라 정보를 찾아감 | 메뉴 (Menu) |
| 리스트 또는 키를 이용하여 정보검색 | 검색 (Search) |
| 시나리오에서 파악 안됨 | 색인방식 (Glossary) |

〈그림 14〉 시나리오 형태와 사용자 인터페이스 전략간의 관계

검색 방법을 채택한다. 이외에도 색인 방식의 사용자 인터페이스 전략이 있을 수 있으나 본 예에는 대응되는 시나리오 형태가 없는 것으로 판단된다. 다음 〈그림 14〉는 시나리오 형태와 사용자 인터페이스 전략간의 상호 관계를 보여준다.

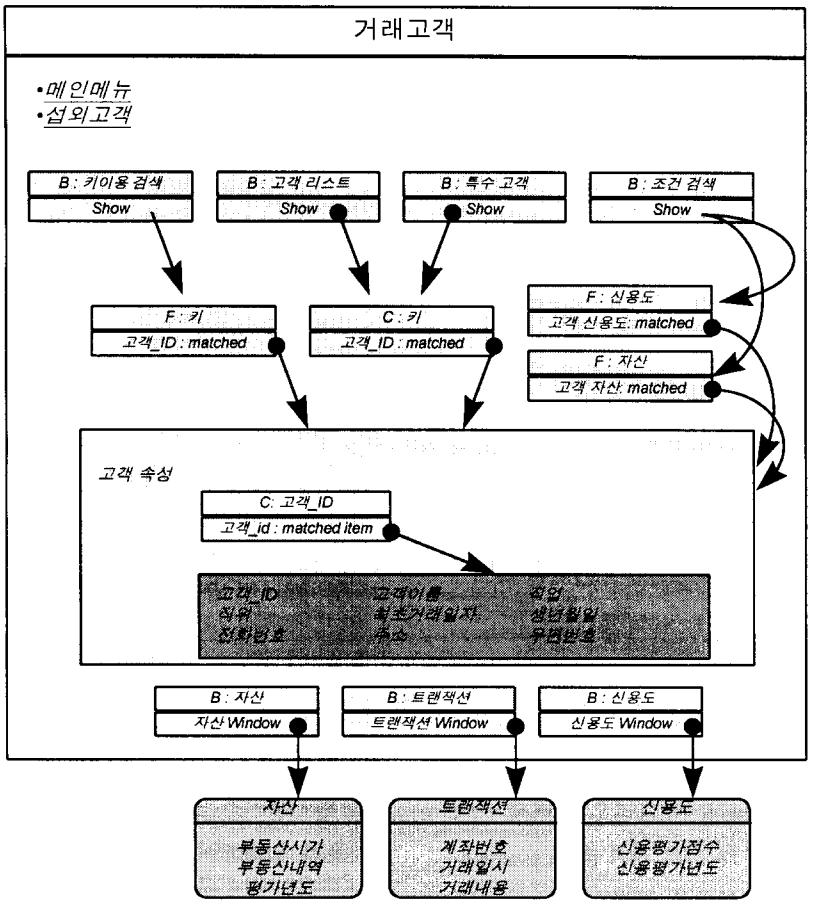
사용자 인터페이스 전략이 정해지게 되면 사용자 인터페이스 설계에서는 페이지 스키마를 이용하여 인트라넷 시스템 구현 시 사용되는 Java [3], HTML 등의 개발 도구에서 제공하는 선택박스(Choice Box), 버튼(Button) 등과 같은 사용자 인터페이스 요소들을 매핑한다. 본 연구에서는 사용자 인터페이스를 단순화하여 〈그림 15〉와 같이 구성 요소들과 표현 기호를 정의하여 사용한다. 제시된 사용자 인터페이스 구성 요소는 사용

구성요소




이동


〈그림 15〉 사용자 인터페이스 구성 요소




〈그림 16〉 사용자 인터페이스 설계 결과

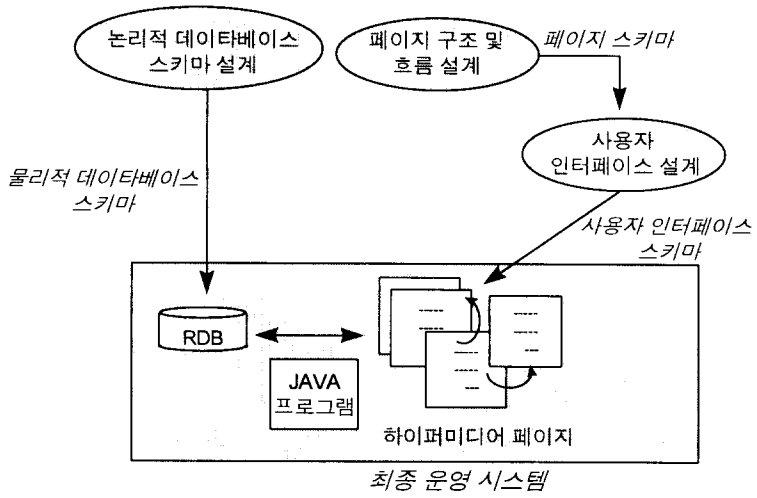
자 인터페이스 전략 4가지를 구현하는데 요구되는 최소한의 집합이라고 할 수 있다. 버튼은 정의된 행위를 실행토록 하며, 리스트는 지정된 항목의 자료들을 리스트 형식으로 나타내 준다. 입력 텍스트 검색은 검색을 원하는 텍스트를 입력하고 일치하는 내용을 찾도록 한다. 이미지는 그래픽 또는 사진과 행위를 링크 시켜 사용자 인터페이스를 구성하는 편리한 방법이다. 선택은 팝업 메뉴로부터 원하는 행위나 항목을 선택하는 방법이며 슬라이드 바는 실제 행위를 바에 의해 선택하여 실행 시키는 사용자 인터페이스 방식이다. 이외에 라인은 목적하는 정보가 있는 곳으로 이동하는 앵커를 나타내며 HTML에서 사용되는 전형적인 방법이다. 또한 다른 곳으로의 이동을 나타내기 위해 화살표를 사용한다.

예를 들어 〈그림 16〉은 앞에서 만들어진 페이지 스키마를 기초로 거래 고객에 대한 Current__Customer__001페이지를 이용하여 사용자 인터페이스를 설계한 결과이다. 거래 고객에 관한 사용자 인터페이스는 거래 고객의 네비게이션 링크 결과를 반영하여 거래 고객의 키 검색, 리스트 및 조건 검색을 할 수 있는 버튼으로 1단계가 구성된다. 예를 들어 키 검색 버튼을 이용한 검색을 할 경우 해당 키를 입력한 후 2단계로 입력 텍스트 검색을 실시한다. 해당된 결과가 일치하면 고객과 관련된 제반 정보가 화면에 전시된다. 사용자 인터페이스 설계 결과에는 이외에도 다른 정보를 찾기 위해 메인 메뉴 및 섭외 고객으로 이동할 수 있는 앵커가 표시되고 해당 거래 고객과 관련된 자산, 트랜잭션 및 신용도 등 상세 정보를 찾을 수 있도록 버튼이 제공된다.

마지막으로 객체 지향 모델로 설계된 클래스 구조도를 데이타베이스 관리 시스템 구조에 맞춰 매핑한다. 본 연구에서는 관계형 데이타베이스 시스템 환경으로 매핑한다. 객체 모델을 관계 모델로 직접 전환하는 것이 가능하며 이에 관한 다수의 방법이 소개되었다 (예를 들면 [6][9]). 본 논문에서는 클래스 구조도를 관계형 테이블로 전환

| 관계형 테이블 이름 | 구분 | 속성 |
|---|---|---|
| 고객 | Table | 고객_Id, 고객이름, 고객주소, 고객신용기록,..... |
| 거래고객 | Table | 고객_Id, 최초거래일자, ..... |
| 섭외고객 | Table | 고객_Id, 접촉기록, 지점_Code,.... |
| 지점 | Table | 지점_Code, 지점명, 주소, .... |
| 계좌 | Table | 계좌_No, 계좌타입, 개설일자, 고객_Id,.... |
| 트랜잭션 | Table | 트랜잭션_Code, 계좌_No, 지점_Code, 발생일자, 트랜잭션내역,.... |
| 거래고객_View | View | 고객_Id, 고객이름, 고객주소, 재산상태, 신용도, ..... |
| 지점별 섭외고객_View | View | 지점_Code, 고객_Id, 접촉기록,.... |
| 지점별 계좌_View | View | 지점_Code, 계좌_No, 트랜잭션집계, .... |
| 계좌 트랜잭션_View | View | 계좌_No, 트랜잭션_Code, 트랜잭션내역, .... |

〈그림 17〉 관계형 테이블 명세



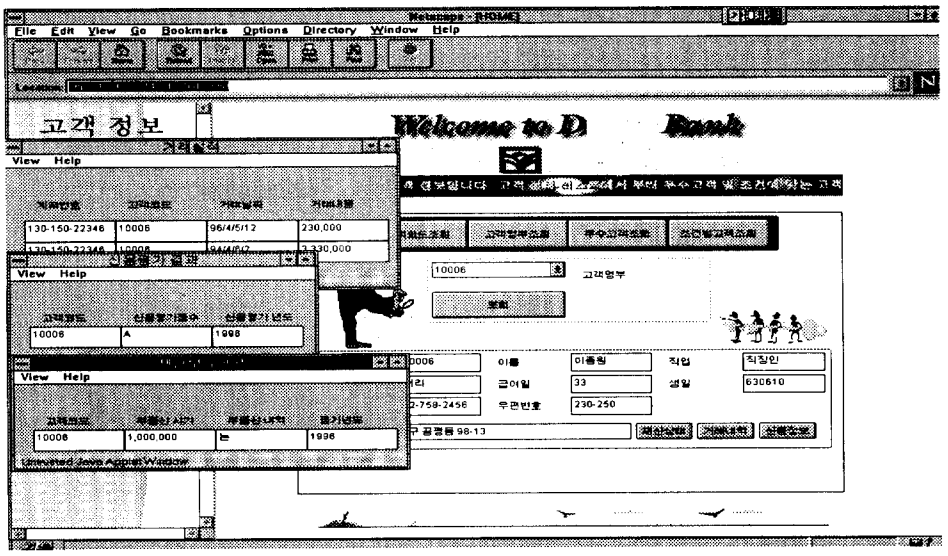

〈그림 18〉 구현 작업 과정

하기 위해 Blaha[6]의 변환 방법을 적용하였다. Blaha방법은 상위 단계, 중위 단계 및 하위 단계의 3가지 표현 과정을 거쳐 객체지향 방법을 이용한 관계형 데이타베이스 설계를 한다[6]. OMT를 이용한 객체지향 모델을 관계형 테이블로 전환하는 것은 중위 단계에 해당한다. Blaha방법에 의하면 객체는 하나의 테이블로 대응되며 객체의 속성들 모두가 테이블의 속성으로 포함된다. 다음은 객체지향 모델에서 객체간의 관계를 위해 사용되는 일반화 (Generalization), 구성품 (Aggregation) 및 제휴 관계에 따라 종속 또는 독립적인 테이블로 전환하게 된다. 이러한 방법으로 〈그림 9〉의 클래스 구조도를 관계형 테이블로 전환한 결과가 〈그림 17〉의 관계형 테이블 명세이다. 뷰는 객체 뷰가 관계형 테이블에 맞춰 전환된 것이다. SOHDM에서는 협업 관계의 전환을 위해 새로운 규칙을 첨가했다. 즉, 협업 관계에 의해 메쏘드를 실행 시킴으로써 유도되는 결과를 나타내기 위해 데이타베이스 내장 함수를 이용한 관계형 사용자 뷰를 정의한다. 〈그림 17〉에서 지점별 계좌 및 계좌 트랜잭션은 트랜잭션의 집계 계산을 위해 메쏘드를 이용하는 협업 관계를 뷰로서 정의한 예가 된다.

### 3.3.6 시스템 구현

시스템 구현 단계에서는 본 개발 방법을 적용한 앞 단계의 산출물들을 활용하여 운용 가능한 하이퍼미디어 시스템을 완성한다. 구현 작업 과정을 구현 설계 내의 세 가지 세부 단계와 산출물들을 연계하여 〈그림 18〉에 도시했다. 그림에서와 같이 논리적 데이타베이스 스키마는 오라클 관계형 데이타베이스 관리 시스템을 이용하여 물리적 스키마로 구현된다. 페이지 스키마와 이를 기초로 만들어진 사용자 인터페이스 스키마를 이용하여 하이퍼미디어 페이지를 제작하고 Java언어를 이용하여 시스템을 구현한다.

구현 과정을 통해 제작된 프로토타입 실행 결과에 대한 주요 내용과 화면을 소개하면 다음과 같다. 〈그림 13〉의 Main_001페이지를 이용하여 ㄷ 은행의 홈 페이지를 작성하여 본 하이퍼미디어 정보 시스템의 메인 화면을 만들었다. 두개의 프레임을 사용하였으며 고객 정보 시스템이라는 프레임에는 거래 고객 정보, 섭외 고객 정보, 경

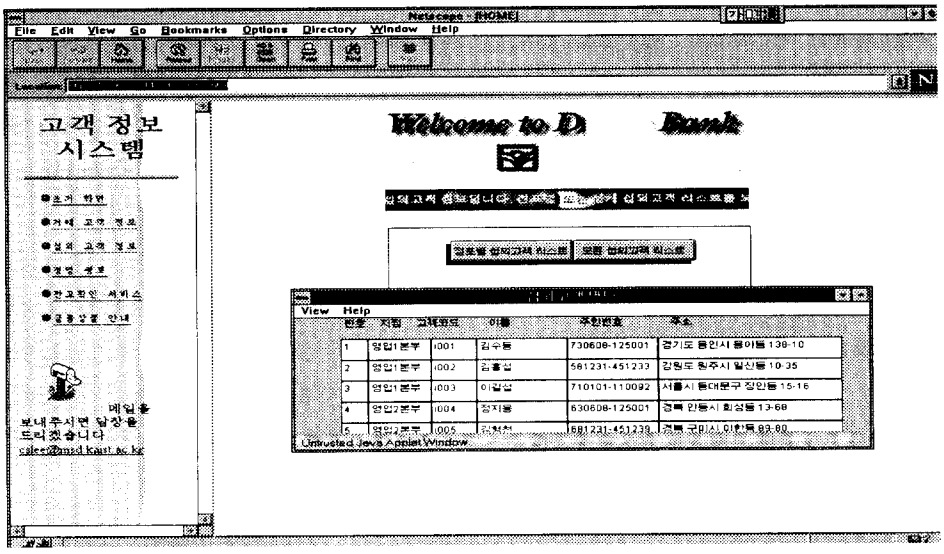



〈그림19〉 거래 고객 상세 정보

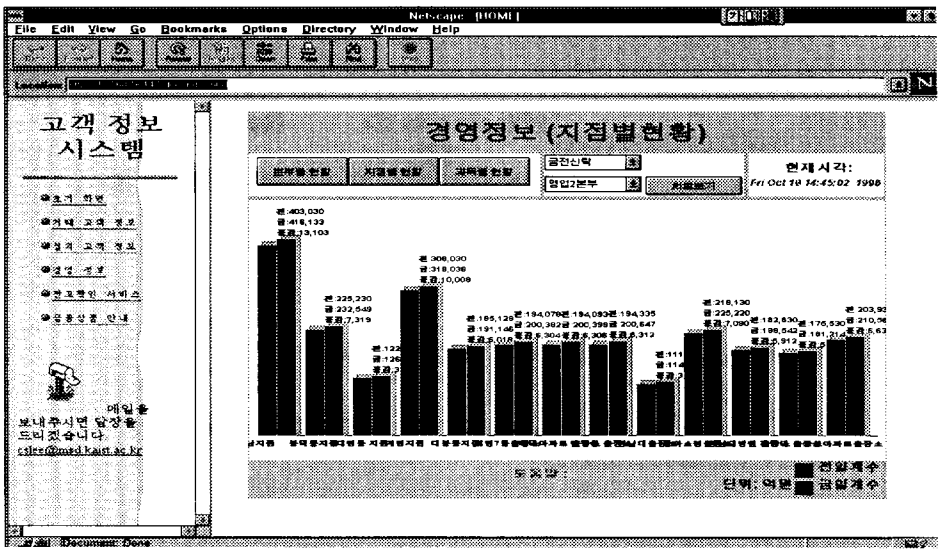
영 정보, 잔고 확인 및 금융 상품 안내 기능이 있고, 중앙의 프레임에는 ㄷ 은행의 소개를 위한 설명과 상징이 그림과 함께 제시되도록 했다.

다음 거래 고객 정보의 메뉴 화면에서는 거래 고객의 명부 조회를 통해 해당 고객의 코드를 키로 하여 고객에 관한 세부 신상 명세를 검색할 수 있으며 사용자는 고객의 명부를 보고 자신이 원하는 고객에 대해 상세한 정보를 찾아볼 수 있다. 본 거래 고객 명부 조회 화면은 〈그림 13〉의 페이지 스키마 중 Current_Customer_001 페이지를 이용하여 제작되었다.

〈그림 19〉는 앞에서 설명한 거래 고객 정보의




〈그림 20〉 지점별 섭외 고객 명단




〈그림 21〉 지점별 수신고 현황 정보

메뉴 화면에서 선정된 거래 고객에 대해 보다 자세한 정보를 찾아 보기 위한 것으로서 Current_Customer_001페이지 스키마를 이용하여 제작된 화면이다. 거래 고객에 관한 재산 상태, 현재까지의 거래 내역 및 신용 평가 정보들을 상세하게 검색할 수 있는 기능을 보여준다. 또한 고객 코드를 키로 이용하거나 고객 명부로부터 원하는 고객을 선택한 후 필요한 정보들을 선별적으로 확인할 수 있다.

〈그림 20〉은 섭외 고객 관리 기능으로서 유치하고자 하는 고객 명단을 해당 지점별로 출력한 결과이다. 본 화면은 〈그림 13〉의 Prospective_Customer_001페이지를 이용하여 작성되었다. 은행의 현 고객은 아니지만 앞으로 유력한 고객으로 전망되는 사람을 접촉하고 접촉한 사항을 기록하여 보관한다.

〈그림 21〉은 MIS_001페이지를 기초로 제작되었으며 경영정보 중 지점별 수신고 현황을 출력한 화면이다. 금전 신탁 과목에 대해 영업 2본부에 속하는 지점별로 전일 대비 금일의 수신고 내역을 막대 그래프로 표시했다. 각 지점별로 고객이 거래하는 내용에 따른 수신고 변화를 실시간으로 확인할 수 있다.

## IV. 결론

본 논문에서는 기업 정보 시스템 구축에 인터넷 기술을 적용하기 위한 인트라넷 기반의 하이퍼미디어 어플리케이션 개발 방법과 최신 기업 정보 시스템 활용에 기여한 내용에 대해 소개하였다. 인트라넷을 이용한 하이퍼미디어 어플리케이션 시스템 구축 시 본 연구에 소개된 시나리오 기반 객체 지향 하이퍼미디어 개발 방법과 프로토타입 시스템을 참조하면 실무에 유익한 결과를 기대할 수 있을 것이다. 더구나 인트라넷을 기반으로 한 정보 시스템 대부분이 하이퍼미디어 어플리케이션으로 구축되는 점을 고려할 때 본 연구의 중요성이 주목된다.

현재 본 논문에서 제안한 개발 방법에 대한 보완 연구가 계속 진행 중에 있으며 이를 통해 하

나의 독립된 개발 방법론으로서 완성할 계획이다. 추가적인 연구 방향은 다음과 같다. 첫째, 개발 방법 측면에서는 시나리오를 기술하는데 있어서 사용자 용이성을 개선시킬 수 있는 연구가 재미있을 것이다. 또한 각 개발 단계간의 논리적 연결이 보다 자연스럽도록 보완할 필요가 있다. 둘째, 보다 큰 프로젝트에 적용하여 유효성을 재검토하는 시도도 가치가 있을 것이다. 마지막으로, Java 언어가 실 개발 환경에서의 적용상 현재는 그 복잡성이 크고 윈도우즈95 이상의 클라이언트 환경에서만 가능한 제한점이 있다. 그러나 이는 Java 개발 환경이 계속 진보하고 컴퓨팅 파워 및 사용자의 컴퓨팅 환경이 향상됨으로써 해결이 기대되는 사안이다.

## 감사의 글

본 논문에 많은 도움을 주신 ㄷ 은행 김재환 사무전산부 부장님이하 직원 그리고 왕컴퓨터 코리아(주) 직원께 감사드립니다.

## 참 고 문 헌

[1] 유천수, 이충석, 김종호, 김영환, 김재환, 조선형, 이희석, "시나리오 객체 기법을 이용한 인트라넷 하이퍼미디어 시스템 개발 사례", 「'96 데이타베이스 심포지움 및 학술대회 논문집」, 한국데이타베이스학회, 1996. 11., pp.265-287.

[2] 한국오라클(주), "인트라넷 그 현재와 미래, 인트라넷 요소 기술", 「ORACLE Magazine 한국판」, 제8권, 제8호(Fall 1996), pp.27-33.

[3] Arnold, K. and James Gosling, *The Java Programming Language*, Addison Wesley, 1996.

[4] Balasubramanian, P., T. Isakowitz, and E. A. Stohr, "Designing hypermedia applications", *Proceedings of the 28th Hawaii International Conference on System Sciences*, Vol.3(1994), pp.354-365.

[5] Bieber, M. and C. Kacmar, "Designing hypertext support for computational applications", *Commun. ACM* ,Vol.38, No.8(August 1995), pp.99-107.

[6] Blaha, M.R., W. J. Premerlani, and J. E. Rumbaugh, "Relational Database Design using an Object-Oriented Methodology", *Commun. ACM*, Vol. 31, No. 4(April 1988)., pp.414-427.

[7] Bolt, R. C., "Intranet Development Tools , *Intranet System*, (May 1996), pp.10-32.

[8] Frank, M., Shifting gears, *Internet Systems*, Vol.1, (1996), pp.6-47.

[9] Fong, J., "Mapping Extended Entity Relationship Model to Object Modeling Technique", *SIGMOD Record*, Vol.24, No. 3(September 1995), pp.18-22.

[10] Garzotto, F., P. Paolini, and D. Schwabe, "HDM: A model-based approach to hypertext application design", *ACM Transaction on Information. Systems*. Vol. 11, No.1(January 1993), pp.1-26.

[11] Garzotto, F., P. Paolini, and L. Mainetti, "Navigational patterns in hypermedia databases", *Proceedings of the 26th Hawaii International Conference on System Sciences* (1993), pp.370-379.

[12] Garzotto, F., L. Mainetti, and P. Paolini, "Hypermedia design, analysis, and evaluation issues", *Commun. ACM*, Vol.38, No.8

(August 1995), pp.74-86.

[13] Gosling, J. and H. McGilton, "The Java Language Environment", Sun Microsystems Computer Company, October 1995.

[14] Halasz, F. G., "Reflection on Notecards: Seven Issues for The Next Generation of Hypermedia Systems", *Commun. ACM*, Vol. 31(July 1988), pp.836-852.

[15] Halasz, F. G. and M. Schwartz, "The Dexter hypertext reference model", *Commun. ACM*, Vol.37, No.2(Feb. 1994), pp.30-39.

[16] Hutt, A. T. F., *Object Analysis and Design-Description of Methods*, A Wiley-QED Publication, John Wiley & Sons, Inc., 1994.

[17] Isakowitz, T., E. A. Stohr and P. Balasubramanian, "Designing hypermedia applications", *Proceedings of the 27th Hawaii International Conference on System Sciences*(1994), pp.354-365.

[18] Isakowitz, T., E. A. Stohr and P. Balasubramanian, "RMM: a methodology for structured hypermedia design", *Commun. ACM*, Vol.38, No.8(August 1995), pp.34-44.

[19] Jacobson, I., *Object-Oriented Software Engineering-A Use Case Driven Approach*, Addison-Wesley, 1995.

[20] Kim, Y. G. , "Process Modeling for BPR: Event-Process Chain Approach", *Proceedings of the Sixteenth International Conference on Information Systems*(December 1995).

[21] Kim, W., *Modern Database Systems - The Object Model, Interoperability, and Beyond*, Addison Wesley, 1995.

[22] Lange, D. B., "Enhanced Relationships in Object-Oriented Database Modeling", *Proceedings of Information Science* '93, Seoul, Korea(October 1993), pp.296-304.

[23] Lange, D. B., "Object-oriented hypermodeling of hypermedia supported information systems", *Proceedings of the 26th Hawaii International Conference on System Sciences*(1993), pp.380-389.

[24] Lange, D. B., "An Object-Oriented Design Method for Hypermedia Information Systems", *Proceedings of the 27th Hawaii International Conference on System Sciences*, Maui, Hawaii, Vol.3(January 1994), pp.336-375.

[25] Lee, H., J. Kim, Y. G. Kim, and S. H. Cho, "A View Based Hypermedia Design Methodology", *Journal of Database Management* (forthcoming).

[26] Lee, H., C. Lee, and C. Yoo "A Scenario -Based Object-Oriented Methodology for Developing Hypermedia Applications", Proceedings of the 31st Hawaii International Conference on System Sciences, 1998.

[27] Oracle Co., *Oracle 7 Server SQL Reference, Release* 7.2, Oracle Corporation, April 1995.

[28] Rumbaugh, J., M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen, *Object Oriented Modeling and Design*, Prentice Hall, New York, 1991.

[29] Schwabe, D. and G. Rossi, "Building hypermedia applications as navigational views of information models", *Proceedings*

*of the 28th Hawaii International Conference on System Sciences*(1995), pp.231-240.

[30] Schwabe, D. and G. Rossi, "The object-oriented hypermedia design model", *Commun. ACM*, Vol.38, No.8(August 1995), pp.45-46.

[31] Schwabe, D., G. Rossi, and S. D. J. Barbosa, "Abstraction, Composition, and Layout Definition Mechanisms in OOHDM", *Electronic Proceedings of the ACM Workshop on Effective Abstractions in Multimedia, Sanfrancisco*, California (November 1995).

[32] Shneiderman, B., *Designing the User Interface*, 2nd edition, Addison-Wesley Publishing Company .Inc., 1992.

[33] Takahashi, T. and A. M. Keller, "Implementation of Object View Query on a Relational Database", *Data and Knowledge System for Manufacturing and Engineering*, Hongkong (May 1994).

[34] Taylor, D. A., *Business Engineering with Object Technology*, John Wiley & Sons, Inc., 1995.

[35] Thuring, M., J. Hannemann, and J. M. Haake, "Hypermedia and Cognition : designing for comprehension", *Commun. ACM*, Vol.38, No.8(August 1995), pp.57-66.

[36] Wirfs-Brock, R., B. Wilkerson, L. Wiener, *Designing Object-Oriented Software*, Prentice Hall, Englewood Cliffs, NJ, 1990.

[37] Yourdon, E., *Modern Structured Analysis*, Prentice-Hall, 1989.