□사례발표□

# 자바 기반의 엔터프라이즈 이-뱅킹 컴포넌트 모델

김 재 훈†

◈ 목 차 ◈

1. 서 론
2. EJB 이-뱅킹의 효과
3. 업무범위
4. 개발환경
5. CBD Methodology
6. 결 론

## 1. 서 론

세계 최초로 Client/Server환경으로 뱅킹 시스템을 도입한 광주은행은 인터넷/인트라넷 기반 뱅킹 시스템을 구축하여 시스템 사용자를 행원 뿐만이 아닌 고객으로 확대하여 전세계 어느 곳에서도 인터넷을 통하여 업무수행을 할 수 있는 시스템을 목표로 세웠다.

이에 객체지향 솔루션을 보유하고 있는 ㈜넥스텍과 지난 1999년11월1일부터 2000년5월31일까지 7개월간 광주은행 본점(정보지원부)에서 객체지향 개발 방법론과 컴포넌트 설계기법을 사용하여 반복적, 점진적 방법으로 3차의 Iteration을 통한 파일럿 프로젝트를 수행하였다.

파일럿 프로젝트의 범위는 은행업무의 핵심인 고객,수신,여신,일계 였으며 객체지향 방법론을 적용하여 체계적으로 시스템을 개발하였고 컴포넌트 기술로 각광받는 Enterprise JavaBeans 기술을 적용하여 서버 프로그램을 컴포넌트화 하였으며 또한 트랜잭션 및 메모리 관리를 자동화 하였다.

기존의 서버프로그램을 프리젠테이션 레이어에서 JSP 및 Applet를 사용하여 웹 기반으로 전향하였으며, 엔터프라이즈 환경에서 동기 및 비동기적 메시지 전달을 JMS를 기반으로 구현하였다.

전체적으로 J2EE 환경으로 개발하여 시스템의 변화시 최소의 비용으로 빠른 시간에 기존의 서버프로그램의 이전이 가능해 졌다.

## 2. EJB 이-뱅킹의 효과

1) 인터넷/인트라넷 기반으로 누구든 세계 어디에서든지 접속이 가능해 졌으며 클라이언트의 배포문제가 해결이 되었다.
2) 은행 업무 중에도 그날의 실적을 즉시 알아볼 수 있는 환경이 되었다.
3) 신속한 유지보수가 가능해졌다.
4) 기본적인 요건들을 갖춘 부모 클래스를 만들어 놓으므로서 신상품 개발시 제반 요건만을 조합하여 신상품을 즉시 개발 할수 있는 환경이 되었다.
5) 뱅킹 시스템의 컴포넌트화를 통해 기능군의 재사용성,개발 생산성, S/W 부품화가 가능해 졌다.
6) 객체지향 분산 트랜잭션 표준인 OTS를 완벽하게 지원하는 어플리케이션 서버로 완벽

† 정 회 원 : 넥스텍 기술개발부 책임연구원

한 분산 트랜젝션 지원을 할 수 있다.

7) 업무개선으로는 누구나 쉽게 접근 할 수 있도록 화면을 대화방식으로 구성했고 계정계/정보계 업무가 일부 통합이 이루어 졌으며, 여신에 전자결제 시스템 도입을 하였으며, Load Balance, Fault Tolerance가 가능해 졌다.

8) 광주은행 환경에 적합한 자체 객체지향 개발방법론 구축이 되었다.

〈표 1〉 EJB 컴포넌트 기반 C/S 시스템과 절차적 시스템의 비교

| 구분 | Main Frame | 구조적 C/S | EJB 컴포넌트 기반 C/S 시스템 |
|---|---|---|---|
| 처리방식 | 중앙집중 처리방식 | 제한적인 분산 처리 방식 | 완전한 분산처리방식 |
| 특징 | 개발 난이도가 높음<br>운영 난이도가 높음<br>집중 데이터 처리 복잡 | 개발 난이도가 높음<br>운영 난이도가 높음<br>집중 데이터 처리 복잡 | 개발 난이도가 상대적으로 낮음<br>운영 난이도가 상대적으로 낮음<br>집중 데이터 처리 간편 |
| 향후 시스템 확장성 | 확장의 어려움 | 확장가능 | 확장성 뛰어남 |
| 장애 복구 | 어려움 | 어려움 | 장애 원인 식별 용이,<br>빠른 복구 지원 |
| 라이트 사이징 | 어려움 | 비교적 용이 | 규모와 성능의 최적화 용이 |
| 유지 보수성 | 중앙처리방식으로 인하여<br>유지보수가 어려우며<br>비용이 많이 듬 | 구조적 방식의 한계로<br>유지 보수 쉽지 않음 | 모듈성 높고 유지 보수성 뛰어남 |
| 트랜잭션 | 처리가능 | 처리가능 | 완벽한 Transaction지원 |
| 정보기술 | 기존 기술 의존 | 기존 기술 의존 | 최신 기술의 지속적 수용 가능 |
| 플랫폼지원 | 맞춤식 개발 | 플랫폼에 맞추어 맞춤식<br>개발로 인한 한계 | Write Once, Run Anywhere<br>어떤 플랫폼에서든지<br>독립적인 실행 |
| 분산환경 | 분산환경지원 불가 | 3-tier환경 | Multi-tier 개발로<br>확장,성능,안정성증가 |
| 보안 | 보안의 어려움 | 추가보안의 어려움 | 추가적인 보안문제의 용이 |
| 기업간 산업표준, 프로토콜 | 불가 | 불가 | 지원가능 |
| 가용성 | 불가 | 거의불가 | 지원가능 |

〈표 2〉 EJB 컴포넌트 기반 C/S 시스템과 절차적 시스템의 비교

| 구분 | 인터넷/ 인트라넷 | 클라이언트/ 서버 |
|---|---|---|
| 사용자 환경 | 일관성 있는 표준 브라우저 이용 | 다양한 개발 도구로 제작된<br>특정 어플리케이션 사용 |
| 이식성 | 이식성 강함(WIN/MAC/UNIX) | WIN/UNIX |
| 네트웍 모듈 | DB NET S/W 불필요 | DB NET S/W |
| 하드웨어 환경 | 가벼운 클라이언트<br>무거운 서버 필요 | 메모리 요구량 많음<br>무거운 클라이언트/서버 |
| 소프트웨어 Deployment | 클라이언트에 프로그램 불필요<br>(자동 Upgrade) | 클라이언트에 프로그램 필요<br>(매번 Upgrade 필요) |
| 애플리케이션 | 화면 크기 자동 | 고정된 화면 크기 |
| 서버 부하 | 필요시에만 서버 접속 | 지속적인 서버 접속 |

9) 사전 신기술의 문제점 및 해결방안을 도출했다.
10) 차세대 계정계 시스템에 대한 시스템 성능 및 안정성등 신기술 평가의 기회가 되었다.
11) 신기술 습득이 되었다.
12) 표준화 작업으로는 객체지향 개발 방법론 산출물 양식, UML, Class Diagram, Sequence Diagram에 대한 Template과 Naming Convention Rule, Validation Rule, 설계의 DB Schema로 자동으로 DB Create와 Entity Bean Source 생성, Source Code Convention Rule, User Interface 표준이 이루어 졌다.

## 3. 업무범위

개발 업무범위를 Sub-System별 주요 Functionality로 분류해 보면 아래와 같다.

| Sub-System | Functionality |
|---|---|
| 공통 | 사용자인증 |
| | 공통함수 |
| | 로그온/오프 |
| 고객업무 | 기본정보등록 |
| | 사고신고등록 |
| | 조회 |
| 수신업무 | 상품관리 |
| | 수신상담 |
| | 계좌신규 |
| | 계좌입금 |
| | 계좌출금 |
| | 계좌해지 |
| | 조회 |
| 여신업무 | 대출상담 |
| | 대출접수 |
| | 전자결제 |
| | 대출실행 |
| | 조회 |
| 일계업무 | 계정처리 |
| | 텔러마감 및 시재등록 |
| | 조회 |

## 4. 개발환경

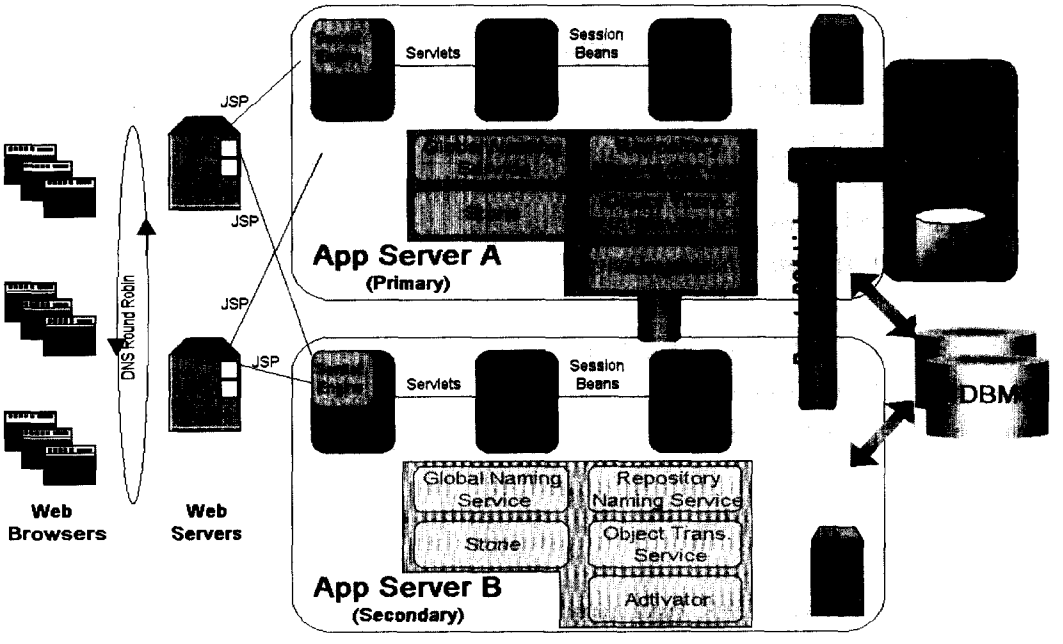
1) 소프트웨어 구성

| 구분 | 제 품 및 Version | 역 할 |
|---|---|---|
| WebApplication Server | Gemstone/J V4.0 | EJB Container 제공 |
| DBMS | Oracle 8i | 데이터 저장 및 관리 |
| JDK | JDK V1.1.7b | JAVA 개발 및 run-time |
| JDBC Driver | Oracle Thin-Driver | Oracle DBMS 연결 |
| Java 개발Tool | JBuilder V3.0 | Java Client 개발도구 |
| Modeling Tool | RationalRose Enterprise 98 | 시스템 분석 및 모델링, 문서화 도구 |
| Web Server | Apach 1.3.9, ServletExec 3.0 | 웹서버 및 JSP엔진 |

2) 하드웨어구성

| 구 분 | 사 양 | 수 량 | 역 할 |
|---|---|---|---|
| AP Server | CPU: Pentium III 500<br>RAM: 256MB<br>OS: Window NT 4.0 | 2 | EJB Server, Business Logic,<br>Persistence Manage,<br>Clustering(Load Balance, Failover) |
| WEB Server | CPU: Pentium III 500<br>RAM: 256MB<br>OS: Window NT 4.0 | 2 | Web Server,<br>Clustering(Load Balance, Failover) |
| DB Server | CPU: Pentium III 500<br>RAM: 512MB<br>OS: Window NT 4.0 | 1 | Repository Server |
| Client | CPU: Pentium III 500<br>RAM: 256MB<br>OS: Window NT 4.0<br>Window 2000<br>Window 98 | 3 | Application 개발용 및 거래 발생용 |

3) 시스템 아키텍처

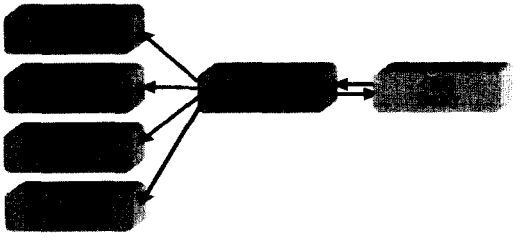



□ Multi-VM architecture:
하나의 분산 클러스터링 아키텍처는 소프트웨어나 비즈니스의 추가 없이 수백개의 자바 VM과 멀티서버의 자원을 관리 할 수 있어 마치 하나 이상의 물리적 시스템을 사용하는 효과를 본다.

□ Smart Load Balance:
서버 레벨에서 로드 밸런싱을 지능적으로 통합함으로써 개발자는 관여할 필요가 없다.

□ HTTP Session상에서 Failover:
Distributed HTTP Session Tracking(DHST)이라는 기술을 사용하여 VM상에 여러 서블릿 엔진을 허용한다. 시스템 요구에 로드밸런스는 물론이고 HTTP Session의 VM들에 문제가 발생하면 DHST는 새로운 VM에 Session을 재시작하고 Genstone/J의 Persistent Cache로부터 그Session의 상태로 복구 시켜준다.

□ 시스템 실패복구 시나리오:
프라이머리 머신에서 복구할 수 없는 문제가 발생하면 Agent는 A 머신에 수행되는 남은 프로세스를 Shut Down 시키고 A 머신에 의존하여 수행되는 B 머신에 VM을 Shut Down 시킨다. Agent는 복구가능 트랙잭션 로그파일(translogs)과 영구데이타파일(extents) 소유권을 B 머신에 이전한다.
머신 B에 시스템 컴포넌트와 VM이 재시작 되고 Web 서버는 자동적으로 VM과 재연결이 된다. 이전에 실패했던 Transaction은 Commit를 수행한다. 머신 A가 Down됨으로써 파워는 반으로 줄지만 다시 시작되면 어떤 지연 없이 Secondary 머신으로 정상 가동 된다.



□ Global Naming Service(GNS)는 JNDI naming service와 Gemstone/J의 관련된 프로세스 모니터링하며 문제가 발생하면 프로세스를 down 시키고 재시작 시킨다.
또한 Primary Server에 GNS는 Secondary Machine에 운영되는 VM을 추적한다.
GNS와 Buddy는 같이 연동되어 GNS에 문제가 있으면 Buddy가 재시작 시킨다.

□ Activator는 Gemstone/J의 위치와 활동 서비스 정보를 수집하며 요구에 따라 적당한 VM을 재조정한다

□ Repository Naming Service(RNS)는 원거리 접근으로 운영중인 시스템에 Repository Name을 관리한다.

□ Stone Transaction Monitor는 Transaction commit 여부를 결정하며 동시처리와 일관성을 유지토록 한다. 또한 Transaction 로그를 남기고 내부 프로세스 채널을 통해 VM과 통신을 한다.

□ Object Transaction Service(OTS)를 사용하여 코바 기반의 분산 Transaction을 지원한다.

□ Gemston/J는 전체 시스템을 차단할 필요가 없는 프로세스 레벨 차단(failover)을 제공한다.

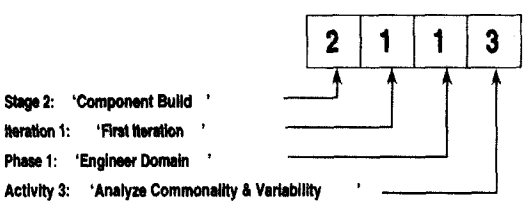
## 5. CBD Methodology

1) Numbering Schema

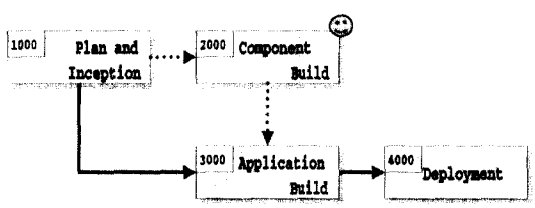
□ 4-Digit Numbering

Stage, Iteration, Phase, Activity, Steps are not numbered.

□ Example




2) Stage




3) 상세내역

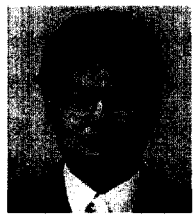
| 절 차 | 내 용 |
|---|---|
| 1000 Plan&Inception | 1999.11.01 - 1999.11.30 |
| 1001 프로젝트계획 | 배경,목적,조직및업무,개발및운영환경,소요자원,개발비용,작업일정 |
| 1002 리스크분석 | 잠재적인 위험영역 설정,위험리스트 작성,중요도를 파악,대책수립 |
| 1003 요구사항정의 | 목적,범위및기능,기대효과,현시스템및업무흐름,기능적,성능적,인터페이스,아키텍처,운영,기타요구사항,추후 인수 테스트시 평가자료로 반영 |
| 1004 용어사전 | 개발자들의 업무이해와 의사소통 통일,구현시 Code Convention에 기초자료 |
| 1005 프로토타입 | 사용자의 요구사항을 빠른 시간내에 도출,시스템의 결점을 찾아 개선방안을 모색,사용자와 시스템간의 상호작용을 검토할 사람을 위한 커뮤니케이션 수단 제공하며 사용자 요구사항을 단계적으로 보여준다. |

| 절 차 | 내 용 |
|---|---|
| 1006 Use-Case | 요구사항을 도출하고 개발자들이 업무를 파악하며 구현 해야할 시스템의 개략적인 범위를 결정 |
| 1007 개략클래스 다이어그램 | 시스템의 정적인 정보를 추출하여 명시 |
| 1008 시스템아키텍처 | 구현하고자하는 시스템 아키텍처를 개략적으로 도식 |
| 1009 프로젝트계획 | 변경된 부분을 감안하여 재작성 |
| 3100 Application Build | Iteration #1 1999.12.01 - 2000.02.29 |
| 3120 분석(Analyze) | Use-Case, 클래스 다이어그램, 용어사전, 시스템 시퀀스다이어그램, Operation 정의, Validation Rule |
| 3140 설계 (Design Application) | 시스템 아키텍처, 사용자 인터페이스, Use Cases,상세 클래스 다이어그램, 시퀀스 다이어그램, 데이터베이스 스키마 |
| 3150 구현(Construct Application) | 서버,클라이언트 구현, 데이터베이스 구축, 단위테스트 |
| 3160 테스트 (Test Application) | 통합 테스트, 시스템 테스트, 인수 테스트 |
| 2200 Component Build | Iteration #2 2000.03.01 - 2000.03.31 |
| 2210 Engineer Domain | |
| 2211 Family Spec | Family 내에서 먼저 컴포넌트가 생성될 관심 영역을 정의한후 관심영역을 포함하는 Family Member로부터 요구사항들을 수집한다 |
| 2212a Term Dictionary | |
| 2212b Normalized Requirement | 구현해야 할 은행 Domain의 구현목적과 업무범위및 기능을 기술한다. |
| 2212c Function Table | Family 요구사항 명세서를 정규화 시키고 정규화된 요구사항을 이용하여 Function Table을 작성한다. |
| 2213 Analyze Commonality | Function Table을 근거로 하여 Common Function을 추출하고, 다시 Common Function으로부터 Use Case를 식별한다. 식별한 Use Case를 근거로 Use-Case Diagram을 작성하고, 각 Use Case별로 명세한다. |
| 2214 Analyze Variability | Use case별로 차이성(Variability)들을 식별하고 분석한다. 차이성은 각각속성, 오퍼레이션, 워크플로우로 구분될 수 있으며,식별된 use case에 대해서 정규화된 Family Member의 요구사항별로 식별한다. |
| 2215 Family Object Model | 수집한 요구사항 명세서로부터 구현 대상 객체(Object)들을 추출하고 관계를 표현함으로써 도메인의 전체 구조를 파악하고 이해하는데 있다. |
| 2216 Specific Component | Family 모델로 추출된 Use Case 다이어그램과 객체모델을 기반으로 컴포넌트를 추출한다. Use Case/Class 메트릭스(Matrix)를 작성한 후, 클러스터링을 하여 후보(Candidate) 컴포넌트를 식별한 후, 명세한다. |
| 2221 Define Family Object Model | 전체 Object Model 제시 |
| 2222 Component Interface | Family Use-Case Model과, Family Object Model을 근거로 하여 컴포넌트 인터페이스의 Public Operation을 식별한다 |
| 2223 Component Workflow | Function에 따른 Sequence Diagram 작성 |
| 2224 Define Customization | 각 Variability들의 형태를 결정하고, 그 차이에 따라 구현형태를 결정한다. |
| 2225 Refined Component Spec | 최종적으로 컴포넌트를 식별한 후, 명세한다. |
| 3300 Application Build | Iteration #3 2000.04.01 - 2000.05.31(Deployment포함) |
| 4300 Deployment | 4301 Write Manual, 4302 Final Acceptance, 4303 Training, 4304 InstallSystem, 4305 Establish Parallel Runs and Crossover, 4306 Establish Support |

## 6. 결 론

본프로젝트에서 즉 관심 대상은 시스템의 안정성과 OLTP상에 성능의 우수성이었다. EJB를 지원하는 어플리케이션 서버의 Fault Tolerance, Load Balance, 2 Phase Commit, Roll Back등 기술적인 문제와 성능에 대해 광주은행에서는 긍정적인 평가를 했고 앞으로 업무의 개선과 기술발전 속도로 많은 향상이 있을 것으로 예측된다.

앞으로 금융환경 변화가 과거 규제변화에서 정보기술의 혁신으로 변화되고 있으며 전자금융의 사업 기회의 여지가 다른 업종에 비해 상대적으로 크고 정보기술의 발달로 고객의 다양한 요구를 충족 시키며 새로운 상품 고안에서 나아가 세계적인 상품 수준의 금융서비스, 상품, 브랜드에 대한 고객수요 자극을 요하고 있다. 인터넷 비중이 증권거래에서 급속히 진행되는데 비해 곧 은행,보험 및 금융정보서비스로 응용범위가 확대될 것으로 예측된다.



**김 재 훈**

1986년 광운대학교 전자계산학과
1985년 두산상사 전산실 입사
1997년 두산정보통신 MIS개발, 품질보증,프로젝트관리, 개발방법론도입
1997년 KCC정보통신 비씨카드신 시스템 총괄 및 제휴카드 업무 팀장
1998년 어울림정보기술 국산 방화벽 인증작업
1999년 넥스텍 기술개발부 책임연구원
관심분야 : e-Banking, 전자상거래, XML, e-Catalog