Design and Implementation of the CORBA based Intranet System
(CORBA 기반의 인트라넷 시스템 설계 및 구현)

이세훈* 황희정** 손완기***
(Seihoon Lee) (Heejoung Hwang) (Wanki Son)

ABSTRACT

According to the internet and the Web are being used more and more, There are many try to construct a intranet to utilize their benefits. In order to these tendency, various ways and techniques are proposed. But to satisfy for adaptability, extendability, flexibilities for rapid technical change, there must be a open architectured framework using standard distributed object techniques.

In this paper, we propose a intranet system framework based on CORBA and JAVA techniques. Also a electronic messaging system was developed on the framework, and have been evaluated by apply to a hospital information system.

요 약

인터넷과 웹의 사용이 급속하게 확산되어 감에 따라 적은 비용으로 이러한 기술의 장점을 기업 내부에 적용하고자 하는 인트라넷의 구축이 활발해지고 있다. 이에 따라 효과적인 인트라넷의 구축을 위한 다양한 방법이 제시되고 있다. 그러나 인트라넷 시스템의 특징인 빠른 기술 변화에 대한 적응성, 확장성, 유연성을 갖추려면 표준 분산 객체 기술을 수용한 보다 개방형의 인트라넷 시스템 프레임워크가 필요하다.

따라서 본 연구에서는 CORBA와 JAVA를 기반으로 한 인트라넷 시스템의 프레임워크를 제안하였으며, 이 프레임워크에 기반을 둔 전자결재 시스템을 개발하여, 종합 병원에 적용하고 평가하였다.

1. 서론

인터넷이라는 새로운 네트워크 환경이 컴퓨터 분야뿐만 아니라 기업 환경을 바꾸고 있다[1]. 정보의 바다로만 인식되어 오던 인터넷이 이제는 기업 네트워크 환경을 혁신적으로 변화시킬 대안으로 떠오르고 있는 것이다. 적은 비용으로, 언제 어디서나, 누구나, 손쉽게 사용할 수 있는 기업 정보 인프라의 구축이 바로 인트라넷이다[1,2,3].

인트라넷의 가장 큰 장점은 인터넷과 웹이 가지고 있는 장점을 그대로 기업 내부에 적용할 수 있다는 것이다. 어느 기업에 이미 인터넷

* 인하공업전문대학 전자계산기과 조교수
** 가천길대학 부설 뉴미디어연구소 선임연구원
*** 한전정보네트웍(주) 주임

환경이 구축되어 있다면 전세계 지사 및 모든 직원들을 쉽게 연결할 수 있기 때문에 인트라넷 환경은 이미 구축되어 있다고 할 수 있을 것이다. 인트라넷은 개발 및 유지보수 면에서 이미 구축되어 있는 하드웨어나 운영체제에 상관없이 표준화된 소프트웨어 개발환경을 이용할 수 있으며, 개방형 구조로서 확장성이 뛰어나다는 장점이 있다. 이 밖에도 각종 서류의 표준화를 비롯한 조직의 표준화, 별도의 교육 없이 웹브라우저 교육만으로 그룹웨어의 사용이 가능한 점, 인터넷 기반의 각종 멀티미디어나 하이퍼텍스트 문서 제작과 이용, 인터넷을 이용해 네트워크를 구축하므로 지역에 관계없이 신속한 업무처리가 가능하며, 적은 추가 비용으로 보수, 유지 및 업그레이드가 가능한 점들로 사용자 중심의 편리한 환경을 네트워크에 구축할 수 있다[1,2,3,4].

그러나 인트라넷 시스템의 특징인 빠른 기술 변화에 대한 적응성, 확장성, 유연성을 갖추려면 표준 분산 객체 기술을 수용한 보다 개방형의 인트라넷 시스템 프레임워크가 필요하다.

현재 분산 객체 기술의 국제 표준인 CORBA(Common Object Request Broker Architecture)는 마이크로소프트사를 제외한 전세계 800여 업체가 참여하고 있는 대규모 컨소시엄인 OMG(Object Management Group)에서 발표한 분산객체와 관련된 표준이다[5].

따라서 본 연구에서는 CORBA와 인터넷의 표준 언어로 자리잡고 있는 Java를 기반으로 하는 인트라넷 시스템의 구조를 제안한다. 또한 제안한 구조를 기반으로 하는 전자 결제 시스템을 구현하고, 종합 병원에 적용함으로써 그 효과를 입증한다.

2. 기존 인트라넷 시스템 고찰

웹 기반 응용프로그램의 개발 혹은 전자결재 및 메시징 시스템을 포함하는 인트라넷 시스템들을 개발하기 위한 기술로 CGI가 등장한 이래 웹브라우저 제작회사, 데이터베이스 전문회사들에 의해 자사 제품의 특징을 살린 다양한 개발 솔루션이 제공되고 있다. 지금까지 발표된 대표적인 개발방법 및 기반환경으로는 다음과 같은 것들이 있다[6,7,8,9,10,11,12].

- CGI(Common Gateway Interface)
- ActiveX
- Oracle Web Application Server
- NSAPI(Netscape Server API)
- ISAPI(Internet Information Server API)
- ASP(Active Server Page)
- Informix Web Data Blade
- CORBA
- Sapphire Web 등

본 연구에서는 이들 중에서 가장 대표적인 개발 방법인 CGI, Fast-CGI라고 불리는 NSAPI와 ISAPI, 그리고 인트라넷 시스템의 가장 중요한 부분인 데이터베이스 전문회사에 의한 솔루션과, 최신의 기술들인 ASP/ActiveX, CORBA/Java 기술에 대해 비교 분석한다.

■ CGI

가장 전통적인 방법이며 비교적 쉽고 빠르게 웹을 기반으로 한 서비스를 개발할 수 있다. C, C++, Perl, Shell Script, Tcl/tk 등으로 개발할 수 있다.

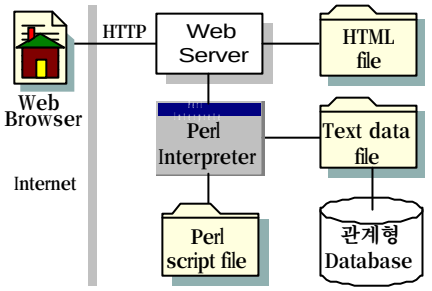

그림 1. 전형적인 CGI 방식 구조

그림1은 Perl 스크립트를 사용해 데이터베이스와 연동하는 전형적인 CGI의 구조이다. 정적인 CGI 방식의 접근에서 웹 서버는 로드, 실행, 종료의 작업을 모든 사용자 프로세스에 대해 시스템 fork()를 사용해 처리해야 한다. 따라서 서버에 과중한 부하가 걸리게 되고 전체적인 성능 저하를 가져오게 된다. 만일 Perl 스크립트가 아닌 C 언어를 사용할 경우에는 데이터베이스 개발업체에서 제공하는 API나 혹은

ANSI 표준인 Embeded SQL 방식으로 프로그램한 뒤 SQL*Net 등의 데이터베이스의 종류에 따른 연결 채널을 통해 연결될 수 있다.

CGI의 장점은 W3C의 표준이며 가장 범용적인 방법으로 어떤 시스템에서도 호환성을 유지할 수 있다는 점을 꼽을 수 있으며 단점으로는 서버 오버헤드가 크기 때문에 대규모의 업무처리에는 적합하지 않다는 문제가 지적되고 있다.

■ Fast-CGI

기존 CGI의 문제로 인하여 넷스케이프사와 MS사는 자사의 웹서버의 기능을 확장하여 다양하고 빠른 처리속도와 안정성을 가지는 전용 CGI API를 제공하게 되었고, 이를 일반적으로 Fast-CGI라는 용어로 부른다. 기존의 CGI는 완전한 정적 실행 방식이지만 Fast-CGI는 동적인 라이브러리 링킹을 통해 런타임 시에 서버의 기능을 확장 및 변경할 수 있는 인터페이스를 제공한다.

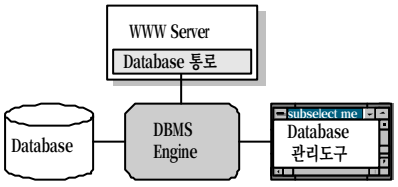

그림 2. Fast CGI 구조 (NSAPI)

이 방식은 데이터베이스의 자료가 데이터베이스 엔진에서 바로 웹 서버를 통해 전송되므로 프로세스간 자료복사 및 프로세스 교체로 인한 부가 비용을 줄일 수 있는 장점을 가지고 있다.

반면 구현하는데 비용이 많이 든다는 점과 특정 웹서버의 전용 API를 사용하는 경우 다른 웹서버와의 호환이 어렵다는 단점을 가지고 있다. 또한 데이터베이스 접근에 대한 요구 사항이 확장 또는 변경될 경우, 웹서버를 수정하거나 API로 프로그램된 데이터베이스 통로를 수정해야 하며, 무엇보다도 웹서버의 확장 API는 현재로서는 표준화가 미흡하다.

■ 오라클 웹 어플리케이션 서버

인트라넷 시스템에서 가장 하단에 위치하는 것이 데이터베이스이며 데이터베이스를 제외하고는 인트라넷 시스템 구축을 할 수 없을 정도로 중요한 요소이다. 현재 오라클, 인포믹스 등은 자사의 데이터베이스를 이용해 보다 쉽고 안정적인 인트라넷 시스템을 개발할 수 있도록 다양한 솔루션을 제공하고 있다. 이 중 오라클은 국내는 물론 전세계 데이터베이스 시장을 가장 많이 점유하고 있는 대표적인 데이터베이스 전문회사로서 자사의 데이터베이스를 사용하는 인트라넷을 위해 웹 어플리케이션 서버를 제공하고 있다. 이 방법은 웹서버를 기반으로 CORBA 규격을 준수하는 Web Request Broker를 통해 다양한 프로그래밍 언어로 개발된 서비스들이 Inter-Cartridge Exchange를 통하여 통합되는 구조를 가지고 있다. PL/SQL 및 웹 패케이지를 사용해 HTML 과 PL/SQL을 함께 사용하는 방식으로 운영할 수 있다.

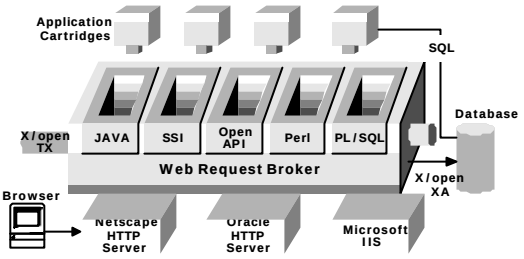

그림 3. 오라클 웹 어플리케이션 서버의 동작

이 방식은 오라클 데이터베이스를 사용하여 PL/SQL 등 오라클에 특화된 기능을 효과적으로 사용할 수 있지만 시스템 구축에 비용이 많이 들며 시스템의 규모가 방대해 진다. 특히 반드시 오라클 데이터베이스를 사용해야 한다는 문제를 가지고 있다.

■ ActiveX / ASP

MS사 에서는 자사의 웹서버 제품인 IIS (Internet Information Server)를 기반으로 확장 CGI인 ISAPI를 제공하며 ISAPI를 확장해 HTML 내에 VBscript 와 연동하여 손쉽게 인트라넷 시스템을 구축할 수 있는 ASP를 제공한다. ActiveX는 MS사의 인터넷 및 인트라넷 전략의 중심에 있는 기술 플랫폼중 하나로서

ActiveX 컨트롤, 스크립트, Active Document, 자바애플릿을 포함하는데, 이중 ActiveX 컨트롤은 경량화된 컴포넌트들로 클라이언트일 경우 Visual Basic/C++에서 작성된 어플리케이션 혹은 OCX 들이 웹브라우저 상에서 바로 실행될 수 있다.

ActiveX와 ASP는 Visual Interdev, Visual Basic 등을 이용하여 비교적 손쉬운 환경에서 시스템을 구축할 수 있는 컴포넌트 모델로서 다양한 사용자 인터페이스 구현이 가능하다.

그러나 IIS와 IE(ASP는 Netscape에서도 동작)에서만 동작하며 속도가 느리다는 것이 단점으로 지적되고 있다.

- CORBA/Java

CORBA는 OMG에서 제정한 객체 통신의 표준 구조로서 현재 2.x가 발표되었으며, 조만간 3.0이 발표될 예정으로 있다[5,13]. CORBA의 특징으로서 어플리케이션 혹은 사용자간의 정보를 공유, 여러 개의 서로 다른 시스템에서 동작 가능, 특정 비즈니스 서비스에 대해 성능 향상, UNIX, PC, MAC 등 상이한 시스템에 설치된 각각의 어플리케이션 간의 정보 공유 가능, 서로의 어플리케이션 객체를 변경하지 않고 비즈니스 프로세스를 변경하거나 재구축 가능, 분산된 네트워크에 흩어져 있는 다양한 객체 서비스를 효율적으로 관리할 수 있는 등의 중대규모의 분산 객체 시스템을 구축하기 위한 최적의 솔루션으로 평가되고 있다.

현재 CORBA를 기반으로 한 인트라넷 시스템의 구축에는 자바와 결합한 ObjectWeb이 최적의 해결책으로 알려지고 있다.

3. CORBA 기반 인트라넷 시스템 설계

본 연구에서는 CORBA/Java 기반으로 인트라넷 시스템을 설계한다.

CORBA는 분산객체의 표준으로 개방형 구조로 현재 운영되고 있는 여러 기간 업무들과의 연동이 용이하며, 향후 개발될 다른 프로그램들과의 인터페이스도 용이하기 때문에 점차적으로 시스템을 확장해 나가면서도 유연성을 유지할 수 있다는 이유가 CORBA를 선정한 가장 큰 이유이다. 본 연구에서는 CORBA를 기반

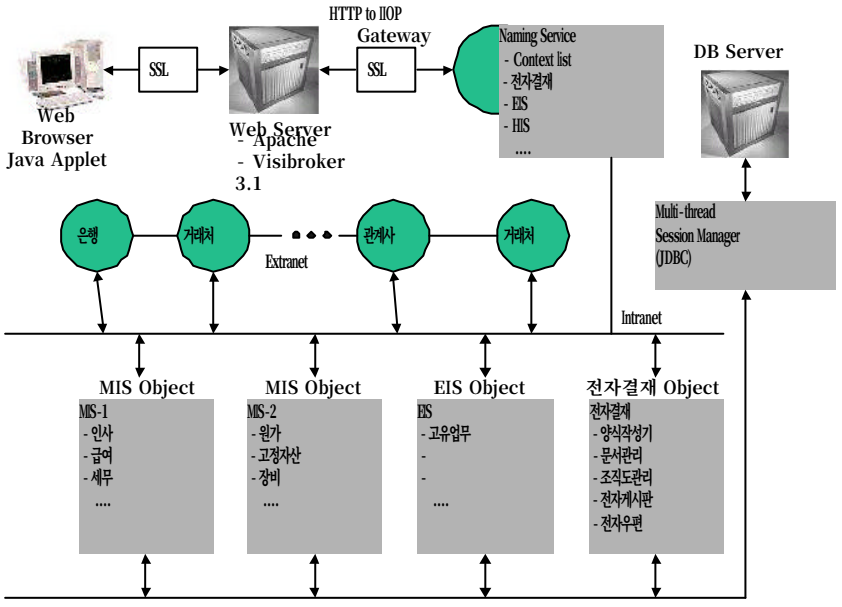

그림 4. 인트라넷 시스템 구성도

으로 하는 인트라넷 프레임워크를 설계한다.

3.1 설계 개요

시스템 구성은 그림4와 같으며 전자 결재를 근간으로 EIS 및 MIS 업무들을 통합하고 업무상 연관되는 다른 기관이나 기업들에서 서비스되는 분산 객체들에 대해서도 상호 연동이 가능한 개방형 분산 객체시스템으로 구성된다.

· 전자결재 : 전자결재 시스템은 단순히 기안문서에 대한 결재 처리라고만 생각할 수 있다. 하지만 진정한 의미의 전자결재시스템은 단순히 문서만을 결재하는 것이 아니라 결재시점에서 조직의 현재 업무의 진행상황이나 관리적인 차원에서 의사결정에 필요한 다양한 정보를 제공할 수 있어야 한다.

· MIS(Management Information System)

MIS는 경영정보 시스템으로 관리적인 차원에 포함되는 모든 업무에 대한 통합 정보 시스템으로 인사, 급여, 세무, 소모품, 고정자산, 원가 관리 등의 업무가 포함된다.

· EIS(Executive Information Systems)

EIS는 조직 업무중 관리적인 차원이 아닌 수행을 중심으로 한 업무 분야에 대한 통합정보 시스템을 말한다.

3.2 시스템 구성 요소 설계

■ 클라이언트

클라이언트 시스템은 모든 웹브라우저에서 지원하는 자바 애플릿을 사용한다. 자바에서 기본적으로 제공하는 그래픽 툴킷인 AWT의 빈약한 인터페이스를 JBCL(Java Beans Component Library) 및 JDK1.2에 포함되어 있는 스윙(swing) 등을 사용하여 보완한다. 또한 DHTML 및 자바스크립트를 사용해 애플릿과의 통신 및 서버 부하를 줄일 수 있도록 하였다.

■ 웹 서버

웹서버는 클라이언트와 CORBA 객체서버와의 중간에 위치하며 HTTP를 통하여 데이터를 주고받고 내부적으로는 IIOP 게이트웨이와 연동하여 클라이언트로 하여금 서버객체와 곧바로 통신할 수 있도록 연결해 주는 역할을 한다. 본 시스템에서는 웹서버의 특별한 기능은 요구되지 않기 때문에 대부분의 서버에서도 운영이 가능하다.

■ HTTP to IIOP 게이트웨이

클라이언트 인터페이스인 애플릿에서 웹 서버와의 통신은 HTTP로 이루어진다. 따라서 CORBA 프로토콜인 IIOP와의 매핑을 위해 게이트웨이를 사용해야 한다. 본 연구에서 제안하는 시스템에서는 자바 ORB를 사용하기 때문에 게이트키퍼(gatekeeper)라는 모듈을 사용한다. 게이트키퍼는 다중 웹서버와의 통신을 위한 체인을 제공하며, interior와 exterior에 대해 SSL을 지원한다.

■ 네이밍 서비스

분산된 CORBA 객체를 참조하기 위한 서비스로 CosNaming을 사용한다. 분산된 서비스 객체의 컨텍스트를 중심으로 하위 서비스를 등록해 객체의 관리 및 참조를 용이하게 한다.

■ 서비스 객체

전자결재를 포함한 EIS 시스템들은 하나의 서버에 존재하거나 분산된 형태로 존재할 수도 있다. 또한 DII(Dynamic Invocation Interface) 및 POA(Portable Object Adapter)를 기반으로 설계해 다른 조직에서 개발될 분산 객체들과도 연동할 수 있는 기반을 확립한다.

■ 멀티스레드 세션 관리자

데이터베이스와의 연결을 관리하는 모듈로 데이터베이스 트랜잭션 및 세션을 모니터링하고 부하 분담이 이루어질 수 있도록 자주 사용되는 데이터셋 등에 대해서는 캐시방식으로 보다 신속한 서비스가 이루어질 수 있도록 한다. 향후 객체 트랜잭션 모니터와 연동할 수 있도록 한다.

3.3 시스템 요소 모듈 설계

본 구현 시스템의 기본적인 사양은 7개의 모듈로 구성되어 있다.

양식작성기는 비주얼한 도구를 사용하여 병원 내에서 사용되는 각종 양식을 디자인할 수 있도록 하였다. 일반적인 GUI 컴포넌트뿐만 아니라 정의해 놓은 특별한 기능을 가진 컴포넌트 즉 달력, 조직도 트리, 화폐단위 등을 제공한다.

전자결재는 상신, 결재, 전결, 후결, 반려, 위임 등의 결재처리 기능과 결재 시 의견첨부기능, 결재 진행 상태 모니터링, 파일첨부 기능 등이 있다.

문서관리는 개인문서함과 부서 문서함을 이용해 결재양식을 포함 다양한 문서를 관리할 수 있다.

조직도 관리는 전자결재처리에 있어 중심부분으로 결재라인의 지정, 자동 Pass 기능, 결재권한 지정 및 개인별 전결한도 지정 등 다양한 설정이 가능하다.

사용자 관리는 조직원들에 대한 개인신상정보, 부서 정보, 인사관련 정보들을 관리한다.

전자우편은 POP3를 지원하여 사내메일 및 인터넷 메일과 연동 가능하도록 한다.

전자게시판은 사내정보공유를 위한 다양한 목적의 게시판을 생성, 관리할 수 있다. 각 개인에 대해 개별 게시판에 대한 사용권한을 지정할 수 있다.

3.4 설계상의 특징

■ 데이터베이스 독립적인 구조

일반적으로 데이터베이스에 따라 다소 차이가 나는 stored procedure, 오라클의 PL/SQL, 데이터베이스 시퀀스(sequence) 등을 사용하지 않았으며, 멀티스레드 세션 관리자가 데이터베이스 연결을 별도로 관리하도록 하여 부하 분담이 이루어질 수 있도록 하고, 데이터베이스 및 시스템 구성 변경 시에도 쉽게 유지관리가 될 수 있도록 설계하였다.

■ 업무로직과 데이타처리의 분리

본 시스템의 서버 모듈은 클라이언트의 요청을 받는 부분과 실제적으로 업무로직(business logic)이 있는 부분 그리고 물리적으로 데이터베이스와 연결해 데이터를 처리하는 부분이 분리되어 모듈의 독립성과 가독성 그리고 이식성을 높일 수 있도록 하였다.

그림 5의 IDL 에서는 인터페이스를 정의한다. createUser라고 하는 메소드를 정의하고 있다. 그림 6은 ORB로부터 서비스 요청을 받을 수 있도록 CORBA 서비스 서버를 등록하는 부분이다. BOA(Basic Object Adapter)를 사용해 Visibroker의 osagent(BOA 데몬)에 객체를 등록하도록 되어 있다.

```
interface subjectManager
[
    long createUser(in string mstID,
                    in string newLogin,
                    in string passwd,
                    in string kind,
                    in string name,
                    in long cls,
                    in boolean senior
        ) raises (subjectException);
        ..........
        ..........
        ..........
]
```

그림 5. 인터페이스 정의어(IDL)

```
class subjectManagerServer {
  public static void main(String args[]) {
    try {
      org.omg.CORBA.ORB orb =
            org.omg.CORBA.ORB.init();
      org.omg.CORBA.BOA boa =
            orb.BOA_init();
      subjectManagerImpl     smgrObj =
            new subjectManagerImpl("IMS");
      boa.obj_is_ready(smgrObj);
      System.out.println(smgrObj + " is ready.");
      boa.impl_is_ready();
    }
    catch (org.omg.CORBA.SystemException e) {
      System.out.println("CORBA Exception");
      System.out.println(e);
    }
  }
}
```

그림 6. 서버 객체

```
public class SubjectManagerImpl
        extends Subject._SubjectManagerImplBase {
  public int createUser(String mainID,
            String    newLogin, String passwd,
            String kind, String name, int cls,
            boolean senior)
          throws Subject.SubjectException {
    try {
      return dbPeer.createUser(newLogin, passwd,
            kind, name, cls, senior);
    }
    catch (Exception e) {
      System.err.println("DB Peer error");
      throw new Subject.SubjectException("DB
            Operation fails");
    }
  }
}
```

그림 7. 업무 로직 구현 객체

그림 7은 IDL로부터 생성된 서버 스켈레톤의 _*ImplBase를 상속받아 실제로 클라이언트의 요청을 처리하는 부분이다. 대부분의 시스템은 구현부분에서 필요한 데이터 핸들링을 관리하지만 본 시스템에서는 데이터처리 부분을 따로 구현해 모듈의 독립성과 유연성을 높이고, 보다 쉽게 유지보수가 이루어질 수 있도록 구현하였다.

```
class SubjectManagerSQL {
  int createUser(String newLogin, String passwd,
                 String kind, String name, int cls,
                 boolean senior)
    throws SQLException {
    try {
      PreparedStatement pStmt;
      Statement stmt;
      Date d = new Date();
      int newID = getNewID("BANC_SUBJECT");
      pStmt = conn.prepareStatement (
            "insert into BANC_SUBJECT "
            +"(SUBJ_ID, KIND, REGTE) "
            +"values (?, 'LOGIN', ?)" );
      pStmt.setInt(1, newID);
      pStmt.setTimestamp(2,new
            Timestamp(d.getYear()+
            d.getMonth()+
            d.getDate()+
            d.getHours()+
            d.getMinutes()+
            d.getSeconds()+ 0);
      pStmt.executeUpdate();
            ..........
            ..........
      conn.commit();
      pStmt.close();
      stmt.close();
      return newID;
    }
  }
}
```

그림 8. 데이터 처리 클래스

그림 8은 실제로 데이터베이스와 연결을 하고 데이터를 핸들링 하는 부분이다. 로직이 수정될 경우에는 구현객체만 수정하면 되고 데이터 핸들링 에 대한 부분이 수정될 경우에는 이 부분만 바꿔주면 되기 때문에 유지 관리 및 향상된 응용 서비스를 제공할 수 있다.

- 컴포넌트화된 객체의 문서내 삽입

양식 작성기에서 다양한 기능을 가진 컴포넌트 사용할 수 있도록 되어 있어 단순한 양식이 아닌 각각의 양식이 기능을 가진 하나의 프로그램처럼 사용할 수 있도록 되어 있다. 현재는 구현된 자바클래스와의 링크정보를 가지는 메타GUI 만 제공 하지만 향후 Bean Box 형태로 대체할 예정이다.

4. 실험 및 평가

본 연구의 결과를 1,000개 이상의 병상을 가진 종합 병원에 적용하여 실험하였다. 초기에는 전자결재를 운영하고 점진적으로 기존의 기간업무 및 새롭게 개발되는 정보 서비스들과의 연동 및 통합운영을 위한 프레임워크 구축을 목표로 CORBA/Java 기반의 전자결재 시스템을 설계 구현하였다. 개발된 시스템은 서버로썬 워크스테이션 상에서 Inprise의 Visibroker for Java 3.1[14]을 사용하였으며, 클라이언트 개발도구로는 Inprise의 Jbuilder를 사용하였으며, 시스템 구성도는 그림 9와 같다.

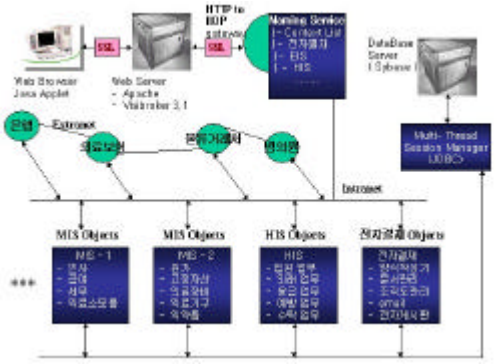

그림 9. 전자결재 시스템 구성도

4.1 병원 인트라넷

병원에서의 EIS라 함은 HIS(Hospital Information System), 즉 환자를 대상으로 한 입원, 외래, 응급, 예방, 수납 업무와 의료보험 조합, 의원 등과의 상호 연관된 업무의 통합 시스템이다.

MIS는 관리적인 차원에 포함되는 모든 업무에 대한 통합 정보 시스템으로 인사, 급여, 세무, 의료소모품, 고정자산, 의료장비, 의료기구, 의약품, 원가 관리 등의 업무가 포함된다.

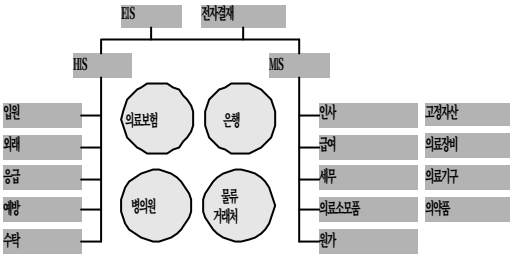

그림 10. 업무 측면의 시스템 구성

4.2 동작 시나리오

· 클라이언트가 웹서버에 전자결재 서비스를 요청

· 클라이언트와 웹서버 간에는 SSL(Secure Socket Layer)를 통한 암호화로 데이터가 보호됨

· 웹서버는 인증된 사용자일 경우 사용자 인터페이스인 자바 애플릿을 전송

· 자바 애플릿은 HTTP to IIOP 게이트웨이와 네이밍 서비스를 통해 서버 객체와 IIOP로 연결된다.

· 서버 객체는 클라이언트 요청에 따른 구현객체를 호출하게 된다.

· 구현객체는 다시 실제로 데이터를 핸들링하는 데이터 처리 객체를 호출한다.

· 데이터 처리 객체는 데이터베이스 세션 관리자를 통해 데이터베이스 에 접속 필요한 정보를 얻어 온다.

· 얻어진 데이터들은 업무 로직이 있는 구현서버에서 가공처리 된 뒤 클라이언트에게 전달된다.

현재 본 시스템은 1차로 각기 업무가 다른 3개 부서에 대해 테스트베드를 구축하여 운영중이다. 적용 결과 다양한 형태의 양식을 직접 제작할 수 있는 양식 작성기가 좋은 반응을 보였으며 기존에 운영중이던 결재 시스템에 비해 운영적인 측면에 있어서 표1과 같은 성과가 있는 것으로 나타났다. 항목별 설문은 3개 부서 총 70명에 대한 설문의 통계이다.

표 1. 적용 평가 항목

측정 항목	효과
문서작성시간	22%
문서의 전송 및 분배시간	30%
문서의 결재처리 시간	30%
실제 전자결재 활용 비율	70%
단위업무 라이프 사이클	50%

5. 결론

본 연구에서는 CORBA를 기반 기술로 인트라넷 시스템의 프레임워크를 제안하였으며, 시스템의 효용성을 위해 전자결재 부분을 구현 모델로 개발하였으며, 자바 빈즈 컴포넌트를 클라이언트로 사용하여 엔터프라이즈급 웹 어플리케이션에서 자바의 적용 가능성을 제시하였다. 또한 인트라넷 부분에 있어서는 적용 사례가 없는 병원의 다른 기간업무 즉 HIS, EIS 시스템과의 연동을 위한 프레임워크를 제안하였다. 향후 연구로써는 Bean Box 형태의 양식작성기 구현, 스크립트 지원, 완벽한 자바 빈즈 컴포넌트로 구성, 객체 트랜잭션 모니터 제공, 전자서명 등의 기능을 구현하는 것이다.

※ 참고문헌

[1] Petrie,C., "INTRANETS: YOUR NEW STANDARD," IEEE INTERNET COMPUTING Vol. 1, No. 5, 1997.

[2] Scacchi, W. and Noll,J., "PROCESS-DRIVEN INTRANETS: Life-Cycle Support for Process Reengineering," IEEE INTERNET COMPUTING, Vol. 1, No. 5, 1997.

[3] Kaneshige,T., "Is the US prepared for cyberwar?," IEEE COMPUTER, Vol. 29, No. 7, 1996.
[4] 가천길대학, 병원 전자 결제 시스템, 최종보고서, 뉴미디어 연구소, 1998.
[5] OMG, http://www.omg.org/
[6] Lazar,P. and Holfelder,P., "CGI vs. Server-Side JavaScript for Database Applications," Netscape DevEdge, July 1997.
[7] Mowbray T. and Zahavi R., *The Essential CORBA Systems Integration Using Distributed Objects*, OMG, John Wiley & Son,Inc., 1995.
[8] Orfali,R. and Harkey, D., *Client/Server programming with JAVA and CORBA*, 2nd Ed.,, WILLEY 1998.
[9] Vogel,A., and Duddy K., *Java Programming with CORBA*, 2nd Ed.,, WILLEY, 1998.
[10] Vogel,A., Building *Distributed Systems in Java*, WILLEY, 1996.
[11] 김평철, 김상욱, WWW와 데이터베이스 시스템의 통합, 제2차 WWW Workshop, 1995.
[12] 황창종, 이세훈, 분산객체컴퓨팅 CORBA프로그래밍, 도서출판 대림, 1998.
[13] OMG, *The Common Object Request Broker Architecture and Specification Revision 2. x*, OMG, 1998.
[14] Inprise, *Visibroker for Java Programmer's Guide Version 3.1*, 1998.

이세훈

1985년 인하대학교 전자계산학과 졸업(이학사)
1987년 인하대학교 대학원 전자계산학과 졸업(이학석사)
1996년 인하대학교 대학원 전자계산공학과 졸업(공학박사)
1987~1990년 해병대 전산실 분석장교
1991~1993년 비트컴퓨터 기술연구소 선임연구원
1993~현재 인하공업전문대학 전자계산기과 조교수
관심분야 : 분산객체컴퓨팅, 멀티미디어, 소프트웨어공학, 원격교육

손완기

1993년 인하대학교 전자계산공학과(공학사)
1995년 인하대학교 대학원 전자계산공학과(공학석사)
1995년~현재 한전정보네트웍(주) 주임

황희정

1995년 경기대학교 경영학과 졸업(경영학사)
1998~현재 인하대학교 대학원 전자계산공학과
1995~1997년 넥스텔
1998~현재 가천길대학 뉴미디어 연구소 선임연구원
관심분야 : 분산객체시스템, 인트라넷, 원격 진료/강의