공 학 석 사 학 위 논 문

# 스마트 클라이언트를 기반으로 한 관세환급 EDI 시스템의 설계 및 구현


2008 년 8 월

부경대학교 산업대학원

전 산 정 보 학 과

박 성 만

공 학 석 사 학 위 논 문

# 스마트 클라이언트를 기반으로 한 관세환급 EDI 시스템의 설계 및 구현

지도교수 박 만 곤

이 논문을 공학석사 학위논문으로 제출함.


2008 년 8 월

부경대학교 산업대학원

전 산 정 보 학 과

박 성 만

박성만의 공학석사 학위논문을 인준함.

2008년 8월 27일


주　　심　공학박사　윤 성 대　인

위　　원　공학박사　정 목 동　인

위　　원　공학박사　박 만 곤　인

# 목 차

# 표 목 차

# 그 림 목 차

# Design and Implementation of Smart Client based Customs Refund EDI System

Seong-Man Park

*Department of Computer and Information, Graduate School of*
*Pukyong National University*

## Abstract

Several years ago, business information system was composed of one application software and related database systems. But now business environment has become more larger and complicated in scope and size due to huge environment of systems which have to think about the complex infrastructure and operation from initial step to end stage considering the system development process and maintenance. The internet bring us the business process to be simplified and focused on the user-oriented interface and function embodied, it can be requires just a little information to users, but nowadays an internet application requires much information and process to satisfy users more carefully. because we know the possibility of base addition capacity of the

Client/Server type application, it is very difficult for user to understand HTML base web application which is limited-convenience capacity.

Needless to say that we can remove a little discomfort through embodying a separate component such as ActiveX, Java-Applet and distribute to users or use AJAX skill when it embodied .But it difficult to security management and the resource of server and client use .Specially, it is so hard to maintain for the developer, and they take so much time on handling the exception of difference version.

As a result, there are so many issues about production such as profitability, convenience and security, each time to set up an Internet application .People have studied many to resole these problems x-Internet has been developed out recently smart client technology of x-Internet mode has been tipped in Visual Studio.Net that was developed by Microsoft corporation which widely used in windows application development. When using the smart client technology to realize application, It has made up the defection that application not being able to accept users' relapse and the just demands.

The purpose of this thesis is to set up a tax refund EDI system that is based on Microsoft smart client technology with actualize disperse treatment in enterprise environment for the people to easy work, to high manufacturing performance, and good for maintenance our systems.

# 1. 서론

기업과 가정에 고속 인터넷 전용선이 보급되면서 사람들은 많은 일들을 컴퓨터를 통해 처리하고 있다. 인터넷을 통한 비즈니스 모델이 등장하고 기업 및 가정 내 사용자들도 인터넷을 통한 업무 처리와 편의 생활 영위에 익숙해져 있다.

몇 년 전만 해도 PC에 소프트웨어를 설치하고 하나의 애플리케이션 및 데이터베이스 서버를 구성하여 운영해오던 업무 환경[1]은 그 규모가 점점 커져 시스템 구축 초기 단계에 관련 인프라 구축과 운영관리까지 고려해야 하는 대규모 시스템 환경으로 발전해왔다. 그리하여 인터넷 전용선이 설치된 곳이라면 어디서든 업무 처리를 하고자하는 사용자들의 욕구가 증가되었으며 그 결과, 인터넷 브라우저만 있으면 업무 처리가 가능한 고성능 애플리케이션 구축에 많은 기업들이 관심을 가지고 투자하게 되었다[2].

인터넷이 가져다준 여러 가지 편리함은 비즈니스 프로세스를 단순화 하고 사용자 위주의 인터페이스 및 기능구현에 역점을 두어 최소한의 정보만을 요구하였기에 가능한 것이었으나[3] 산업 현장같이 업무처리를 위해 많은 정보와 절차를 필요로 하는 곳에서의 인터넷 애플리케이션은 사용자에게 세심한 주의를 요구하였다. 더욱이 클라이언트/서버 형 애플리케이션에서 기본적으로 가능했던 부가기능들을 머릿속에 간직한 채 HTML로 구성된 UI에서 제한된 편의 기능을 이해하기란 어려운 현실이었다.

물론, ActiveX, Java- Applet 같은 별도의 컴포넌트를 구현하여 사용자에게 배포하거나 AJAX (Asynchronous Javascript and Xml)같은 기술을 애플리케이션 구축 시 사용함으로써 불편함을 일부 해소할 수 있었지만 서버 및 클라이언트의 자원  활용 과 보안 관리에 더욱 어려움을 겪게 되었

다. 특히, 개발자들은 유지보수와 컴포넌트의 버전 차이에서 오는 알 수 없는 예외 처리에 많은 시간을 투자해오기도 하였다. 그로인해 생산성, 수익성, 편의성, 안전성에 대한 이슈는 인터넷 애플리케이션 구축 시 매번 발생되는 문제가 되었고 이를 해결하기 위한 각종 연구가 진행되어온 근래에 X-internet[4]이 등장하게 되었다.

윈도우즈 애플리케이션 개발 시 대중적으로 사용되는 마이크로소프트사에 의해 개발된 Visual Studio.NET에서는 이러한 관점에서 X-internet의 형태인 스마트 클라이언트라는 기술을 제시하였다. 스마트 클라이언트 기술은 애플리케이션 구현 시 현업 사용자의 반복적이고 정당한 요구를 외면하게끔 만든 웹 환경에서의 제약을 충분히 해소시키며 빈번한 업무 절차 변경에 따른 애플리케이션 재배포, 유지관리 등의 문제를 해결 할 수 있도록 설계된 기술이다.

본 논문에서는 기존의 관세환급 Web EDI 시스템 사용자의 불편사항을 수렴하여 마이크로소프트사의 스마트 클라이언트 기술을 도입하고 엔터프라이즈 환경에서 분산 처리가 가능한 프레임워크 및 제반 기술을 적용하여 개발자의 생산성향상 및 유지보수 효율에 상승효과를 가져다 줄 수 있는 새로운 관세 환급 EDI 시스템 구축을 목적으로 하였다. 1장 서론을 시작으로, 2장에서는 관련 연구로 관세 환급 EDI시스템 및 기반 기술에 대해 분석하고, 3장에서는 스마트 클라이언트 EDI 시스템의 구성도, 분산처리, 환경 설정에 대한 부분을 기술한다. 4장에서는 스마트 클라이언트 관세 환급 EDI를 설계하고 5장에서는 시스템을 구현 하고 효용성을 고찰한다. 마지막 6장에서는 논문의 결론을 맺고 향후 과제를 제시하였다.

# 2. 관련연구

## 2.1 관세 환급 EDI 시스템

가. 관세 환급

관세 환급이란 납세의무자가 물품의 수입 시 납부한 관세, 가산금, 가산세 등을 과/오납 하였거나 수입 원재료를 제조 가공하여 수출하는 등 관세 관계 법률에서 정하는 일정한 사유가 발생하였을 때 이미 납부한 관세 등을 되돌려 받는[5] 관세법이다.

관세환급의 절차는 업체에서 환급신청서를 작성한 후 EDI로 관세청 환급시스템에 전송하면, 서류제출이 필요 없는 건은 전자서류만 심사하여 환급금을 결정하고 서류제출이 필요한 건은 3일내에 환급신청서등을 제출하면 세관에서 전자서류와 제출서류를 대조 확인한 후 환급금을 결정, 한국은행과 지급은행에 이체 및 지급 의뢰하여 환급신청인의 환급금 전용계좌에 입금[6]된다.

나. EDI(Electronic Data Interchange)

EDI(Electronic Data Interchange)는 전자문서교환, 전자 자료교환으로 해석되며, 서로 다른 기업 또는 조직 간에 표준화된 상거래서식 또는 공공서식을 서로 합의한 통신표준에 따라 컴퓨터 간에 교환하는 새로운 정보전달방식으로 정의한다[7]. 우리나라의 EDI의 표준으로 KEDIFACT[8]

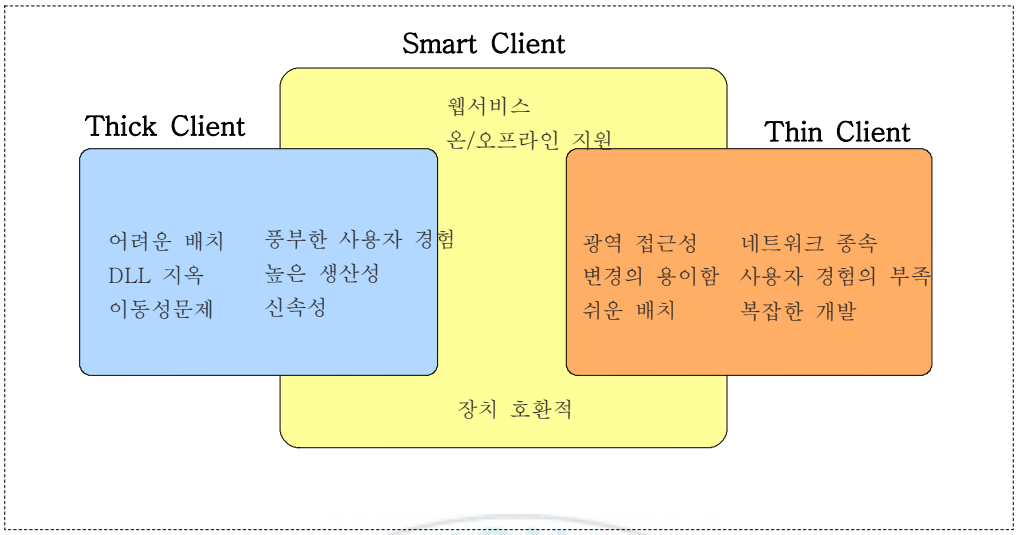
(Korea Electronic Data Interchange for Administration, Commerce and Transport) 이 있으며 UN/EDIFACT[11]에 기반 하여 작성되었다.

다. 관세 환급 EDI 시스템

관세 환급 EDI는 납세의무자가 기 납부한 물품의 관세를 돌려받기 위해 환급신청서를 작성하여 직접 세관에 제출하는 업무를 인터넷을 통해 전자문서로 신청 가능하게 한 방식이다. 관세 환급 EDI 시스템은 수입된 물품에 대한 정보, 납세 증명 정보 등을 관리하고 이를 세관에 전송하기 위한 전자문서 형태로 바꾸어 TCP/IP를 통해 세관에 전송하며 이에 대한 응답문서를 수신 받아 관리용 데이터로 변환하는 기능을 가진 소프트웨어이다.

## 2.2 스마트 클라이언트(Smart Client)

스마트 클라이언트(Smart Client)는 로컬 리소스를 활용하고 분산된 데이터 소스에 인텔리전트 하게 연결됨으로써 뛰어난 적응력과 대응성을 갖춘 클라이언트 응용 프로그램으로 풍부한 대화형 경험을 제공하고 쉽게 배포 및 관리할 수 있는 기술이다[12]. 스마트 클라이언트 기술을 적용한 애플리케이션의 동작 방식은 접근대상 URL에 대한 로컬 PC의 런타임 보안 정책을 변경 적용 시키고 “Application Configuration”에 의거하여 실행에 필요한 Assembly를 사용자 PC에 실시간으로 다운로드 받은 후 Local 애플리케이션 도메인을 생성하여 닷넷 Framework에 포함된 CLR (Common Language Runtime)하에서 작동하게 된다




<그림 1> 스마트 클라이언트의 특성

<표 1> 클라이언트의 특성 비교

| 구분 | Smart Client | Thin Client |
|---|---|---|
| 기능성 | 다양한 기 개발 3rd Party 컴포넌트 및 BackOffice Resource 이용. | 한정된 HTTP 프로토콜 기반 브라우저 기능에 의존한 기능제공 한계. |
| 데이터 처리 | 안정적인 대량의 자료 처리. Remoting, 로컬 다운로드&캐쉬 메커니즘에 의해 처리속도 다소 지연 | 대량의 자료 처리 어려움. 프로세스, 로직 튜닝에 의한 속도 상승 |
| 유지보수성 | RAD 개발,자동배포 방식에 의한 높은 생산성 및 유지보수성. | 생산성이 비교적 낮으나 유지보수성은 높음 |
| 보안성 | Formatter에의해 형식화된 메시지 전달방식으로 인한 보안성 높음. | 보안에 취약(전송 자료 노출위험). 별도의 보안 정책 필요. |
| 실행속도 | 버전확인, 로컬 다운로드, Memory Load에 이르는 속도 비교적 느림. | 비교적 빠른 속도 |

## 2.3 닷넷 Enterprise Library

닷넷 Enterprise Library는 Microsoft 에서 권장하는Best Practices의 캡슐화된 Library로 닷넷 기반 엔터프라이즈 애플리케이션 개발을 위해 일반적이고 공통적인 기능을 구조화된 Pattern으로 구현해놓은 모듈이다.[13] 이 모듈을 닷넷 Framework1.1 에서는 “Application Block”으로,, 닷넷 Framework 2.0 이상에서는 “Enterprise Library”로 명명하고 있다. 닷넷 Enterprise Library는 7개의 프로젝트로 구성되어 있는데 각각의 네임스페이스 및 용도는 다음과 같다.

가. Microsoft.ApplicationBlocks.ConfigurationManagement
환경설정과 관련된 기능들에 대한 Helper Classes

나. Microsoft.ApplicationBlocks.ConfigurationManagement.Interfaces
환경설정과 관련된 기능들에 대한 Class Interface 정의

다. Microsoft.ApplicationBlocks.ApplicationUpdater
프로그램의 자동 업그레이드를 위한 Helper Classes

라. Microsoft.ApplicationBlocks.ApplicationUpdater.Interfaces
프로그램 자동 업그레이드와 관련된 기능들에 대한 Class Interface 정의

마. Microsoft.ApplicationBlocks.Data
데이터베이스 처리를 위한 Helper Classes

바. Microsoft.ApplicationBlocks.ExceptionManagement
예외사항 처리를 위한 Helper Classes

사. Microsoft.ApplicationBlocks.ExceptionManagement.Interfaces
예외사항 처리와 관련된 기능들에 대한 Class Interface 정의

Enterprise Library는 개발자들이 노력과 비용을 들여 구현 해야 하는 반복적 기능들에 대해 패턴을 적용하여 확장 사용을 용이하게 하고, 오류나 예외사항을 적절히 무마시켜 유연한 개발환경 및 개발 생산성을 제공한다.

본 논에서는 7개의 프로젝트 중 데이터베이스와 예외처리 부분에서 Enterprise Library를 이용하였고, 이를 통해 분산된 환경에서의 오류사항 Notification과 COM+를 통한 데이터 처리 부분에서 상당 분량의 코드 감소와 서비스 패킷 전달 방식의 구조적 안정성이 확보된 시스템 구현이 가능하였다.

## 2.4 분산처리 및 닷넷 Remoting

분산처리는 여러 개의 처리시스템에 서비스를 분산시키고 시스템의 유기적인 연결을 통해 시스템을 통합하는 방식으로 구현된 서비스 방식이다. 닷넷 Remoting은 대규모 사용자 트랜잭션을 수용하고 서비스 확장을 용이하게 하며 닷넷 플랫폼간의 통신을 가능케 하는 기술로 분산 컴포넌트(COM+)를 호출하는 RPC 메커니즘을 추상화하는 명확하고 간단한 아키텍처를 제공해 원격 메소드 호출을 가능하게 한다. 닷넷 Remoting은 객체를 활성화시키고, 모든 객체의 수명을 통제하고, 통신채널(Communication Channels)을 사용해서 원격 개체들 간에 메시지를 전송한다[9].

가. HTTP Channel

원격 객체들 간에 메시지를 전송하기 위해 HTTP 프로토콜을 사용하는 채널 클래스를 제공한다. HTTP Channel은 메시지 payload를 채널을 통해 전송하기 전에 SoapFormatter 클래스를 사용해서 SOAP 프로토콜을 사용

하는 XML 포맷으로 메시지를 Serialize 한다. 이기종 원격 객체의 메소드를 호출할 수 있는 범용성을 가지지만 TCP Channel에 성능이 뒤진다.

나. TCP Channel

응용프로그램 도메인 간에 TCP 프로토콜을 사용해서 메시지를 전송한다. TCP Channel은 원격 객체에 메시지를 전송하기 전에 Binary Formatter 클래스를 사용해서 Binary 포맷으로 메시지를 Serialize 한다.

<표 2> HTTP/TCP Channel 비교

| HTTP Channel | TCP Channel |
|---|---|
| HTTP 프로토콜을 사용한 메시지 전달 | TCP 프로토콜을 사용한 메시지 전달 |
| SoapFormatter를 이용한 메시지 Serialize/DeSerialize | BinaryFormatter를 이용한 메시지 Serialize/DeSerialize |
| 범용성, 확장성 지향 | 성능 지향 |

HTTP 또는 TCP Channel을 통해 원격 객체간 메시지를 주고 받기 위해서는 객체를 활성화 시켜야 하는데 서버 객체를 활성화(Server-Activated) 시키는 방법과 클라이언트 객체를 활성화(Client-Activated) 시키는 방식이 있다.

가. 서버 활성화(Server-Activated)

서버 클래스의 메서드를 호출하는 경우에 객체들이 만들어진다. new를 통해 서버 클래스 인스턴스를 생성하는 경우 서버 클래스의 객체 프록시가 생성되고 메서드를 호출하는 시점에 객체가 활성화 된다.

객체의 라이프 사이클은 서버에 의해 관리된다.

나. 클라이언트 활성화(Client-Activated)

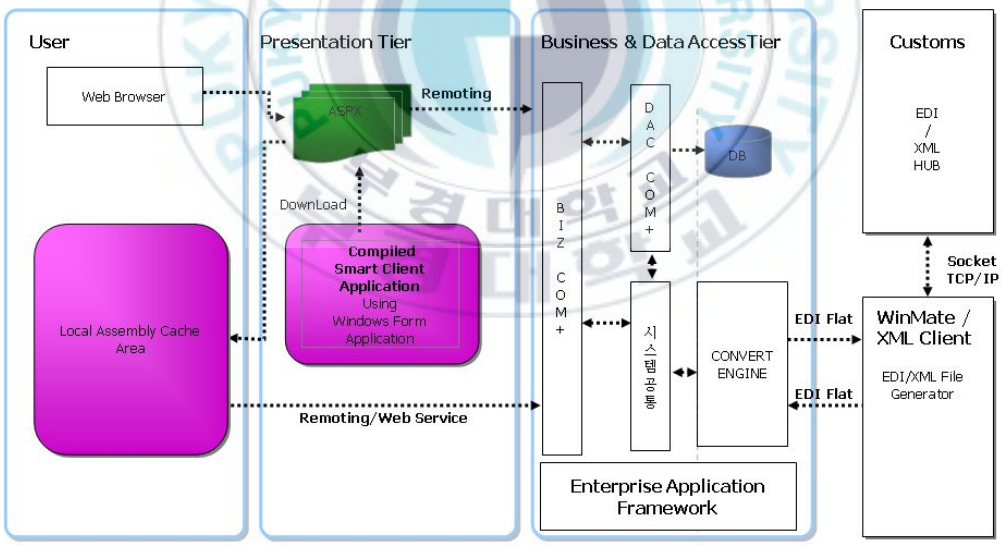
서버 클래스의 생성자를 호출하는 경우에 객체들이 만들어진다. new를 통해 서버 클래스 인스턴스를 생성하는 시점에 객체가 활성화 되고 생성된 객체는 클라이언트의 참조 해제시 삭제된다.

Remoting 서버는 COM+로 구현된 서버 객체를 호스팅하고 이에 접근하기 위한 접근 채널 및 URL을 외부로 노출시킨다. 호스팅은 일반적으로 윈도우즈 서비스로 구현되며 노출된 URL을 통해 Remoting 클라이언트와 데이터를 주고 받는다. Remoting 메커니즘은 이미 구현되고 감추어져 있어 간단한 설정과 구현만으로 분산 객체에 대한 접근이 가능하다.

# 3. 스마트 클라이언트 EDI System

## 3.1 스마트 클라이언트 EDI 시스템 구성

본 논문에서 구현하고자 하는 EDI 시스템은 UN/EDIFACT에서 제공하는 표준에 따라 관세청에서 제정한 메시지수행지침서(MIG)에 기반한 관세환급 EDI 애플리케이션으로 닷넷 Framework 기반 하에서 설계되었으며 스마트 클라이언트 애플리케이션의 구동 및 관련 Assembly 배포를 위한 웹 서버, 비즈니스 컴포넌트와 EDI 변환 및 송수신을 담당하는 애플리케이션 서버, 데이터베이스 서버 등으로 구성된다.




<그림 2> 스마트 클라이언트 EDI 시스템 구성

<그림 2>에서 보는 바와 같이 스마트 클라이언트 EDI 시스템은 총 3개의

영역으로 구성된다.

### 3.1.1 User 영역

사용자 PC 영역으로 스마트 클라이언트 애플리케이션을 구동하기 위한 기본 환경 설정이 필요하다. 사용자가 웹 브라우저 상의 실행 파일을 포함하는 부트스트랩(Bootstrap) 파일의 URL을 클릭하면 부트스트랩은 사용자의 PC에 닷넷 Framework설치 여부, Data Access Control의 최신 버전의 설치여부를 조사하여 인스톨한다. 또한 웹 서버로부터 Local Assembly Cache에 업그레이드 된 파일 다운로드를 위한 보안 정책 설정을 실시한 후, 프로그램을 구동한다.

### 3.1.2 Presentation Tier

웹 서버는 UI(Aspx)를 통해 사용자 인증을 요청하고 애플리케이션을 배포 디렉토리에 배치하여 사용자에게 제공한다. 디렉토리에 배치된 애플리케이션 파일은 모듈별로 버전정보를 포함하고 있으며 단위 모듈로 구성된 DLL은 닷넷 리플렉션 기능에 포함된 버전 컨트롤 기능으로 업그레이드 된 DLL만을 새로이 Local Assembly 영역에 다운로드 받아 Local 애플리케이션 도메인 상에서 애플리케이션을 구동한다. 이후, 클라이언트 애플리케이션은 닷넷 Remoting을 이용해 서버의 Business 객체 Proxy를 생성시키고 이를 통해 데이터 입출력을 처리한다.

### 3.1.3 Business & Data Access Tier

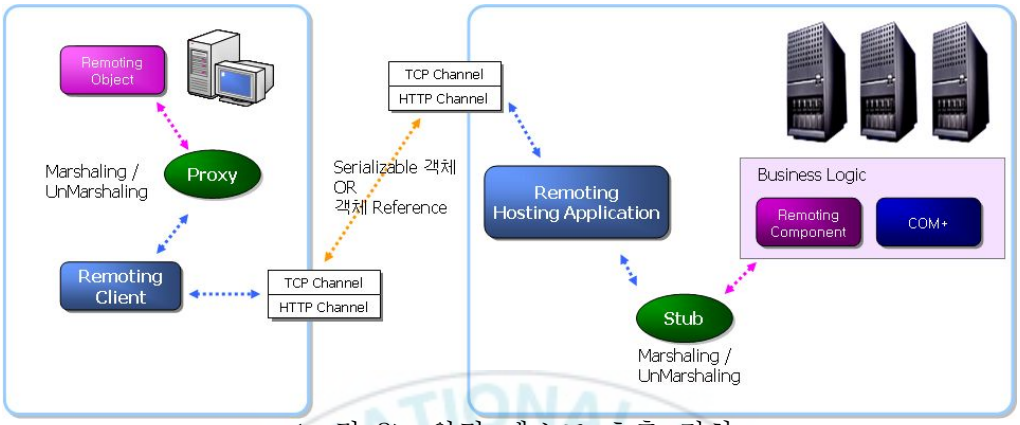
관세환급 EDI 업무를 위한 비즈니스 로직은 업무단위별 COM+로 구성하고 이를 Remoting 객체로 구현하는 DLL로 생성시킨다. 각각의 DLL은 클라이언트 애플리케이션에 의해 최초 호출시 COM+ 서비스에 등록되기 때문에 최초 호출시 약간의 Delay Time이 발생된다. 비즈니스 로직은 데이터 베이스와의 트랜잭션을 담당하고 저장된 데이터로부터 사용자가 요청한 결과를 생성한다. EDI 전송을 위한 스케줄링 된 Flat File 생성 엔진은 주기적으로 Flat File을 생성하고 이를 EDI 파일로 변환하는 기능을 가진 WinMate 애플리케이션을 실행시켜 무역EDI VAN 사업자인 KT-NET (한국무역정보통신)으로 송신한다. KT-NET은 전송된 문서의 정합성을 검증하고 관세청 실무자들이 관세 행정 처리를 하기 위한 데이터로 생성한다.

## 3.2 원격 메소드 호출

본 시스템은 사용자 로컬 PC에 다운로드 된 애플리케이션에서 애플리케이션 서버에 구성된 비즈니스 컴포넌트를 호출하기 위해 닷넷 Remoting을 이용한다. <그림 3> 는 Local PC에 다운로드 된 Assembly에서 원격지에 구성된 COM+ 컴포넌트의 원격 메소드를 닷넷 Remoting으로 호출하는 절차를 나타낸다.

닷넷 Remoting 클라이언트는 서버 객체의 Proxy를 생성하고 요청할 메시지를 Marshalling하여 Channel을 통해 서버로 전달한다. 서버에 생성된 Stub 객체는 메시지를 UnMarshalling하고 비즈니스 로직으로 전달하여 업

무를 처리한다. 처리된 결과를 클라이언트로 전달하는 방식도 이와 같다.




<그림 3> 원격 메소드 호출 절차

## 3.3 Configuration 설정

애플리케이션 구동 프로그램은 웹 서버의 애플리케이션 배포 디렉토리에서 Local PC로 다운로드 되어 실행된다. 웹 서버를 통해 실행되는 스마트 클라이언트 애플리케이션이 존재하는 디렉토리에 <그림 4>와 같이 app.config 파일을 구성하여 애플리케이션 구동 파일이 참조하는 필수 참조 DLL 목록을 등록 하여 다운로드 될 수 있도록 해야한다.

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <configSections>
    <!--Configuration Application Block-->
    <section name="applicationConfigurationManagement"
             type="Microsoft.ApplicationBlocks.ConfigurationManagement.ConfigurationManagerSection
                   Handler,Microsoft.ApplicationBlocks.ConfigurationManagement,
                   Version=1.0.0.0,Culture=neutral,PublicKeyToken="/>
    <section name="ApplConfig1" type="Microsoft.ApplicationBlocks.ConfigurationManagement.XmlHasht
                   ableSectionHandler,Microsoft.ApplicationBlocks.ConfigurationManagement,
                   Version=1.0.0.0,Culture=neutral,PublicKeyToken="/>
  </configSections>
<!--*****************************************************************************************-->
  <applicationConfigurationManagement defaultSection="ApplConfig1">
    <configSection name="ApplConfig1">
      <configCache      enabled="true"
                  refresh="1 * * * *" />
```

기본적으로 다운로드 받아야하는 DLL목록


<그림 4> Configuration 설정

## 3.4 Runtime 보안정책 설정

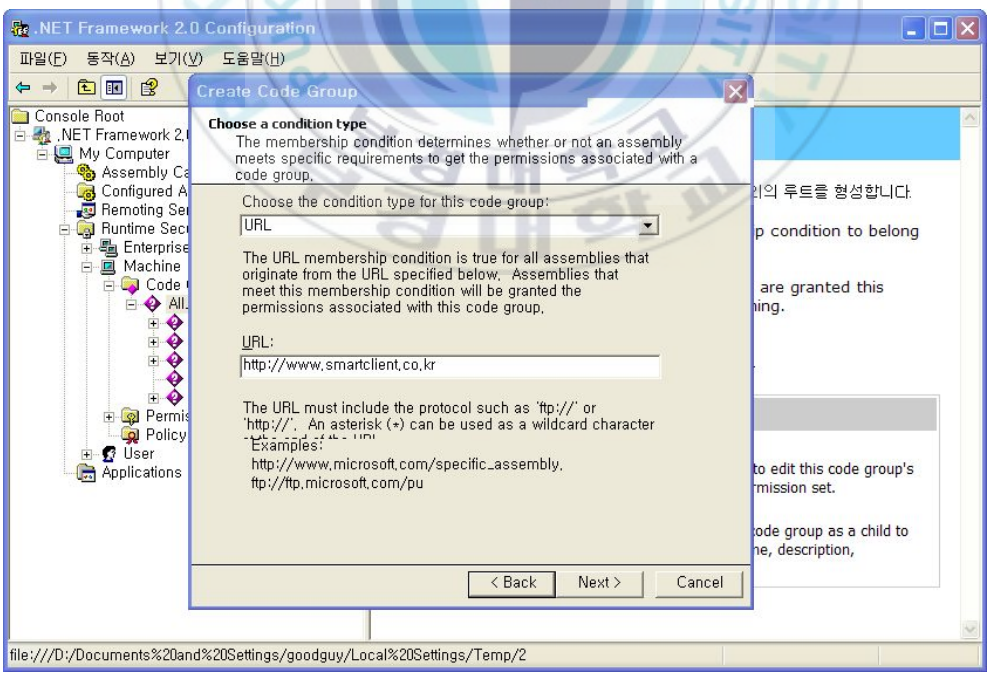
ActiveX의 경우 인증서를 배포 패키지에 Signing하여 사용자의 PC에 다운로드 됨을 알리고 Local PC에 설치한다. 스마트 클라이언트는 닷넷 Framework Configuration의 Runtime 보안 정책에 배포 사이트를 허용하고 Local PC에 대한 접근 정책을 설정해야 한다. 이러한 보안 정책 설정은 Web 사이트에서 프로그램 최초 구동시 부트스트랩 파일로 제공하여 자동으로 닷넷 Framework의 설치 여부를 확인하고 보안 정책을 Local PC에 적용한다. 제어판 -> 관리도구 -> Microsoft .NET Framework 2.0 Configuration을 선택하여 등록된 정책을 확인할 수 있다.

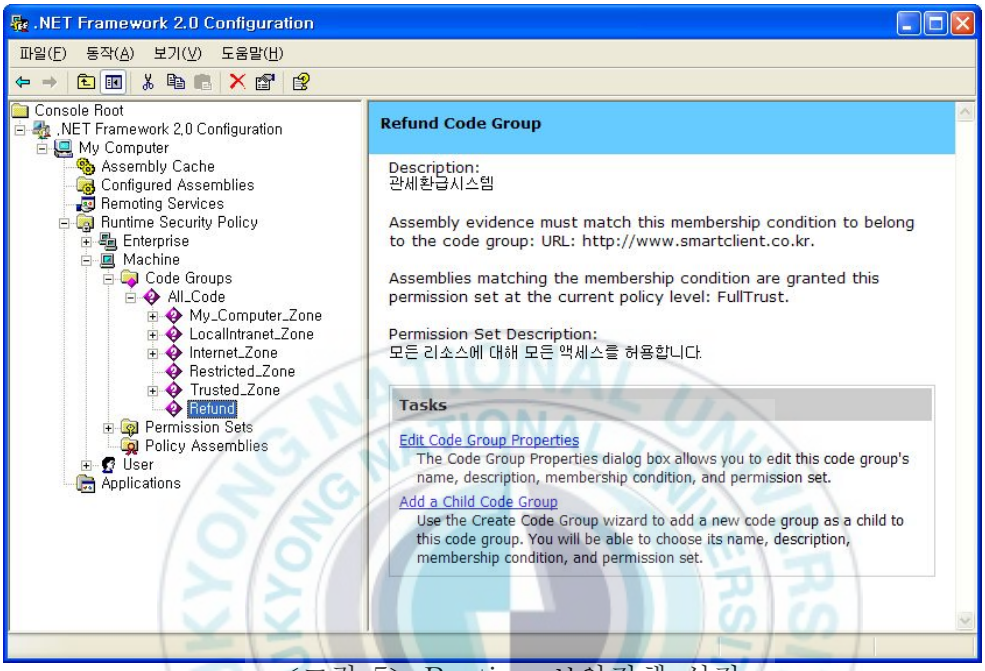
<표3 MemberShip Condition Type>

| Condition Type | 설 명 |
|---|---|
| Application Directory | 실행프로그램과 동일한 디렉토리(하부 포함)내의 모든 |

| | |
|---|---|
| | 어셈블리에 대한 보안 해제 적용 |
| Hash | MD5, SHA1 알고리즘에 매칭되는 해시 어셈블리에 대한 보안 해제 적용 |
| Publisher | 배포자명, 생성자명, 해시방법등에 매칭되고 전자서명되어 인증된 어셈블리에 대한 보안해재 적용 |
| Site | 특정 사이트명에 대해 접근시 보안해제 적용 |
| Strong Name | 표기된 공용키에 의해 강력한 이름이 부여된 어셈블리에 대해 보안해제 적용 |
| URL | 특정 사이트에 접근하는 모든 어셈블리에 대해 보안해제 적용 |
| Zone | Internet, Local Intranet, MyComputer, Trusted Site, Untrusted로 분류된 Zone에 대해 보안해제 적용 |

본 논문에서는 <그림 5>와 같이 특정 URL에 대한 보안 해제 정책을 이용하였다.






<그림 5> Runtime 보안정책 설정

# 4. 스마트 클라이언트 관세환급 EDI 시스템 설계

본 논문에서 구현하고자 하는 시스템은 Windows XP/2000/2003 환경에서 Visual Studio .NET 2005의 객체지향 언어인 C#을 이용하여 프레임워크를 설계하였고 유용한 헬퍼 클래스를 제공하는 닷넷 Enterprise Library를 참조하였다.

<표 4> 구현 환경

| 구 분 | 사 양 |
|---|---|
| 사용자 시스템 사양 | Pentium4 2GHz 512Mb RAM 이상 |
| DBMS | Microsoft SQL Server 2000 |
| Application Server(COM+) | Windows 2000 Standard Server |
| Case Tool | Microsoft Visio 2003 |
| Web Server | IIS 5.0 |
| 사용언어/툴 | Visual C#, Visual Studio .NET 2005, ASP.NET |
| 리포팅 툴 | Crystal Report built in .NET |
| 3rd Party Control | .NET Magic Library<br>DevExpress Control |

## 4.1 프로그램 목록

무역업체에서 EDI를 통해 기관에 관세환급을 신청하기 위해서는 신청 -> 검증 -> 결과통보의 절차를 가져야 한다. 각각의 신청 문서는 신청에 관련된 문서의 종류, 화물의 수입일자, 화물관리번호, 환급금액 등 여러 가지 항목이 포함되어야 하는데 이러한 항목 및 규정은 관세청에서 정의하여 배포하고 있고 개발 업체에서는 각 항목을 구분하여 데이터 베이스에 저장

하고 관리하여야 한다. 이러한 항목을 정의한 문서를 전자문서 규약(MIG)이라 하고 아래 <그림 6>과 같은 항목명, 세그먼트, 문자타입, 길이, 기본값 및 반복횟수 등이 정의되어 있다. 따라서 데이터를 효율적으로 관리하기 위한 방법으로 문서의 종류에 따라 프로그램 목록을 설정하여 관리할 수 있도록 설계하였다.

환급신청 (무역업체 -> 세관) 1/17

| 서식 순번 | 항목이름 | 항목설명 | LINE REF. | SEGMENT | DATA EL. | DATA REP. UN DIR. LEN. | DATA REP. KCS LEN. | DATA REP. M/C | 기타 |
|---|---|---|---|---|---|---|---|---|---|
| | 전자문서 참조번호 | 발신인이 부여하는 전자문서 참조번호를 기재 | 1 | UNH | 0062 | an..14 | an..14 | (M) M | ⓒ |
| | 전자문서 형태 | 전송될 전자문서의 유형을 기재 | | | S009 0065 | an..6 | an6 | M \| M | CUSDRW |
| | 전자문서 개정번호 | 전자문서 유형의 개정번호를 기재 | | | 0052 | an..3 | an1 | \| M | S |
| | 전자문서 배포번호 | 현행 개정번호내의 배포번호를 기재 | | | 0054 | an..3 | an3 | \| M | 98A |
| | 관리기관 | 전자문서의 규정, 유지 및 공포를 관리하는 기관을 기재 | | | 0051 | an..2 | an2 | \| M | KE |
| | 문서형태구분 | 전자문서의 이름을 나타내는 부호 (환급신청서임을 나타냄) | 2 | BGM | C002 1001 | an..3 | an3 | (M) M \| M | 5DA : 환급신청 |
| 갑-3 | 제출번호 | 신청서를 세관에 제출시 신청인이 부여하는 번호(신청인부호(5) + 연도(2) + 일련번호(6)) | | | 1004 | an..35 | an13 | M | ⓒ |

<그림 6> 환급신청서 MIG 형식

### 4.1.1 관세환급 전자 문서 관리 목록

관세환급 EDI 애플리케이션에서 관세청으로 각종 신청을 하기 위해서는 전자문서에 대한 규약이 존재해야 한다. 관세청에서는 이를 위해 전자문서 규약집을 EDI 개발업체, 무역업체에 배포하고 개발된 응용프로그램은 정해진 기간내에 검증 및 인증을 득해야 사용이 허가된다. 전자문서는 보통 무역업체에서 세관 또는 관세청으로 송신하는 문서가 있고, 세관 또는 관세청에서 무역업체로 송신하는 문서가 있다. 아래 문서는 무역업체에서 세관

으로 송신하기 위해 관리해야하는 전자문서이다.

- 환급신청서관리
- 분할증명서 관리
- 제증명 정정/취하신청서
- 기초납세증명서 관리
- 평균세액증명서 관리
- 정산신고서 관리

이러한 전자문서를 세관에 신청하게 되면 세관 담당자는 문서를 검토한 후 다음과 같은 응답 문서를 회신해야 한다.

- 접수통보 확인
- 심사완료통보
- 분할증명서 통보
- 정정처리결과 통보 확인
- 오류통보 확인
- 기초납세증명서 통보
- 반입확인서 통보

관세환급 EDI 시스템은 위와 같은 전자문서 관리 메뉴들을 포함하고 있으나 전체를 구현하기에는 적지 않은 인력과 기간이 필요한 바, 본 논문에서는 환급신청서 관리에 중점을 두었고, 프로그램 운영상 환급신청서와 연관되어 구현되어야 하는 접수/오류통보, 코드관리, 송수신 관리 및 환경설정 기능만을 설계, 구현하였다.

### 4.1.2 코드관리

전자문서에 포함되는 데이터들은 관세청에서 기준한 코드성 데이터들과 항목의 값들로 구성되어야 한다. 또한 무역업체에서 데이터의 무결성을 보장하기 위해 생성한 코드들도 관리되어야 한다. 이러한 코드들을 하나의

대메뉴에 포함하여 아래와 같이 설계하였다.

- 간이정액환급율표
- 전자문서코드 관리
- 정정항목코드관리
- 관세청표준코드관리
- 거래처코드관리
- 물품세번부호(HS)
- 우편번호
- 환급금지급은행코드관리
- 세관부호관리

### 4.1.3 송수신 관리

무역업체는 관할 세관이나 관세청으로 최종 목적인 관세환급을 위해 전자문서를 송신해야한다. 사용자는 한건 또는 여러건을 한번에 송신해야하는 경우도 존재한다. 문서를 송신하기 위해서는 EDI Flat 파일을 WinMate의 Out 디렉토리에 생성해야 하는데 이러한 기능을 담당하는 송수신 엔진을 설계하여 시스템 상주형 프로세스로 구동시키고 송수신 엔진은 송신준비 테이블을 읽어 체크 사항을 확인한 후 EDI Flat 파일을 생성시키도록 하였다.

- 송신대기문서목록
- 센터정보조회
- 송수신로그조회
- 통관정보조회

### 4.1.4 환경 설정

기본적으로 회사의 정보와 사용자 관리, 권한 관리등은 관리되어야 하며 차후 전자문서 송/수신에 따른 과금을 부과할 시 참고할 수 있는 자료로

준비되어야 한다. 이는 분쟁의 소지를 감쇠시키고 오류 발생시 책임소지를 분명히 할 수 있는 자료로 활용 될 수 있다. 아래의 메뉴를 통해 각 업무 담당자에게 메뉴를 허용하고, 권한을 부여할 수 있도록 하였다.

- 메뉴관리
- 권한관리
- 문서번호관리
- 개인정보관리
- 자사정보/사용자관리
- 거래상대방설정
- 프린터설정

## 4.2 데이터 베이스 설계

EDI는 일반적으로 관세청에서 제공한 MIG에 기반하고 있다. 단순하고 명확한 MIG의 기준과 업무적으로 필요한 정보를 추가하여 데이터 베이스 테이블을 설계하였다. 이후에 언급될 모듈 설계에서 자동 쿼리 생성기능이 포함되도록 구성하였기에 신청서에 관계된 Master-Detail-Sub Detail 구조의 테이블은 참조키로 연결하되 Delete/Update Cascading을 명시하지 않았고, 검색시간을 향상시키기 위해 인덱스만 설정하였다. 스마트클라이언트를 기반으로한 관세환급 EDI 시스템 구현을 위한 주요 테이블 설계 및 관계를 살펴보면 다음과 같다.

### 4.2.1 테이블

중/소/대규모 사용자용 시스템을 위해서는 각각 별도로 생성하여 관리하

였으나 업체구분을 두어 중/대규모로 확장하고, UI 프로그램 구현시 구분자를 두어 컴파일 하는 형태로 소규모 사용자를 수용하는 방안을 모색하였다. 컴파일시 솔루션 속성에 하나의 컴파일 구분자를 두고 이를 조정하면 다양한 규모를 지원하는 형태의 패키지로 구성 가능하다. 아래의 <그림 7>은 중/소/대형 규모를 지원하고 환경 설정을 위해 구성한 관리용 테이블의 목록이고 <그림 8>은 관세환급 신청서 및 송수신 문서를 관리하는 EDI용 테이블 설계의 일부분이다.

가. 관리용 테이블목록

| NO | 테이블ID | 테이블명 | 내용 |
|---|---|---|---|
| 1 | COMPANY | 사용업체정보 | 업체 정보 |
| 2 | USR | 사용자정보 | 사용자 정보 |
| 3 | AUTHORITY | 사용자별 권한정보 | 사용자별 업무권한 설정정보 |
| 4 | EDI_PARTNER | EDI 거래상대방 관리 | EDI 거래상대방 부호 및 정보 관리 |
| 5 | COMPANY_SVCINFO | 업체 사용 서비스 | 업체별 사용서비스(환급, 기타) |
| 6 | PROGRAM_LIST | 업체 사용 프로그램 | 업체별 사용 프로그램 |
| 7 | SERVICE_LIST | 서비스 목록 | 서비스 목록 |
| 8 | MIG_CODE | 미그 목록 | 세관 송수신 미그 정의 |
| 9 | SND_READY | 송신 대기목록 | 송신을 위한 데이터 변환 상태 저장 |
| 10 | SNDRECV_LOG | 송신/수신 로그 | 송신/수신 내역 로그 |

<그림 7> 관리용 테이블 목록

나. EDI용 테이블 목록

| NO | 테이블ID | 테이블명 | 내용 |
|---|---|---|---|
| 1 | CusApe_M | 오류통보-Master(화물오류통보공유) | 오류통보 수신시 마스터 저장 테이블 |
| 2 | CusApe_s | 오류통보-Detail(화물오류통보공유) | 오류통보 상세내역 저장 테이블 |
| 3 | DutyCustom | 환경설정 TABLE | 사용자 정보 및 환경설정 테이블 |
| 4 | BANKCODE | 은행코드 | 은행코드 저장 테이블 |
| 5 | Hea_Gong | 해외공급자코드 | 거래처(해외 물품 공급자) 테이블 |
| 6 | HSJUNGAK | 간이정액율표 | 세관 정의 간이 정액율표 저장 테이블 |

| | | | |
|---|---|---|---|
| 7 | HSMST | 세번부호코드 | 세관 정의 세번부호코드 저장 테이블 |
| 8 | Place | 장치장소코드 | 장치장(보세창고)코드 저장 테이블 |
| 9 | SAEKOAN | 세관코드 | 세관코드 저장 테이블 |
| 10 | STMST | 거래처코드 | 거래처코드 저장 테이블 |
| 11 | STN_CODE | 표준코드 | 각종 코드 저장 테이블 |
| 12 | TongKoan | 통관고유부호 | 거래처별 통관고유부호 저장 테이블 |
| 13 | AVE5DC_K | 평균세액증명서 갑지 | 평균세액증명서 갑지1 테이블 |
| 14 | AVE5DC_S | 평균세액증명서 갑지2 | 평균세액증명서 갑지2 테이블 |
| 15 | AVE5DC_U | 평균세액증명서 을지 | 평균세액증명서 을지 테이블 |
| 16 | GY_5DB_B | 기초원재료납세증명서 병지 | 기초원재료 납세증명서 병지 테이블 |
| 17 | GY_5DB_K | 기초원재료납세증명서 갑지 | 기초원재료 납세증명서 을지 테이블 |
| 18 | GY_5DB_U | 기초원재료납세증명서 을지 | 기초원재료 납세증명서 갑지 테이블 |
| 19 | TAX5DA_B | 환급신청서 병지 | 환급신청서 병지 테이블 |
| 20 | TAX5DA_J | 환급신청서 정지 | 환급신청서 정지 테이블 |
| 21 | TAX5DA_K | 환급신청서 갑지 | 환급신청서 갑지 테이블 |
| 22 | TAX5DA_U | 환급신청서 을지 | 환급신청서 을지 테이블 |

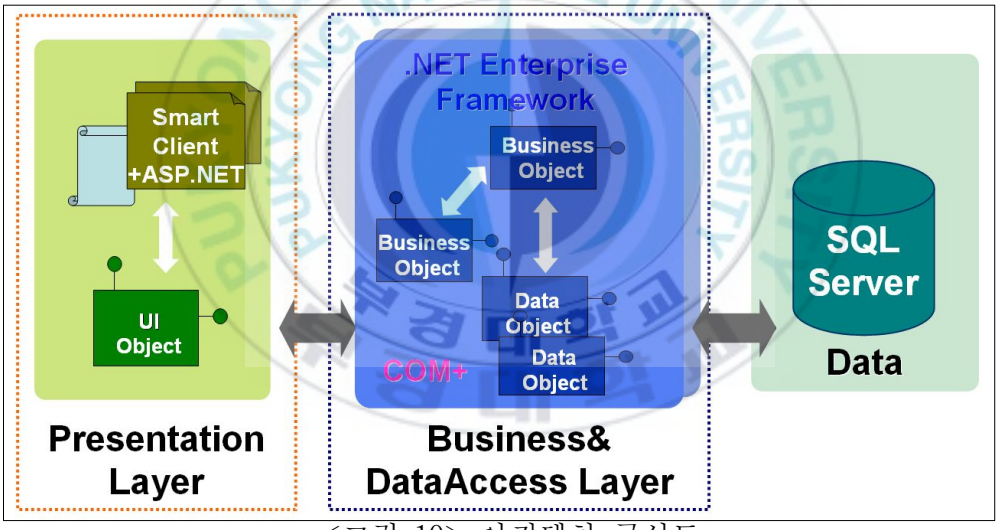
<그림 8> EDI용 주요 테이블 목록

## 4.3 프로그램 설계

### 4.3.1 아키텍처 설계

스마트 클라이언트 애플리케이션에서 닷넷 Remoting은 TCP/Binary 조합에서 가장 높은 성능을 가질수 있다. Internet 상에서는 이를 위해 방화벽의 TCP Port를 Open 해야하는 문제가 있을 수 있다. 만약, 라우터 내부의 Intranet 용으로 구성하는 경우 외부 Port Open없이도 사용 가능하다. HTTP/SOAP 조합을 이용해 Web Service를 이용한다면 방화벽에 관계없이 이기종 간의 서비스를 이용할 수 있다. 하지만 라우터 외부로 서비스 범위를 확장시켜 HTTP/SOAP 조합을 이용 한다면 다소 느린 속도를 감안해야 한다. 본 논문에서는 다소 느린 속도를 보완하기 위해 내부 데이터 전달 방식에 TCP/Binary 조합을 이용하도록 구성하였다.

아키텍처 계층은 Presentation, Business, Data Access Layer 로 구성하여 이 후 다양한 방법의 Presentation Layer 에서 접근 가능토록 하고, Business Layer 및 Data Access Layer는 COM+ 서비스 객체로 구현하여 구성요소 서비스에 등록하였다. 사용자 PC에서 구동되는 스마트 클라이언트 애플리케이션은 Presentation Layer 에서 원격지 서버의 DataBase에 직접적인 트랜젝션 처리를 하지 않고, 서버의 Business Logic 컴퍼넌트의 클라이언트 Proxy를 생성하여 통신 한다. Business Layer와 Data Access Layer는 Enterprise Library와 결합시켜 단일 창구형(Facade) 프레임워크로 구성하여 단순화 하였다.

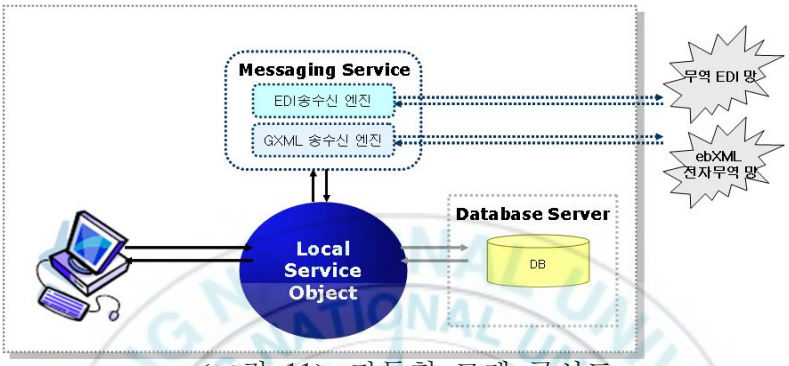



<그림 10> 아키텍처 구성도

이러한 아키텍처 설계는 총 3가지 모델 유형으로 구성이 가능하다.

가. 단독형

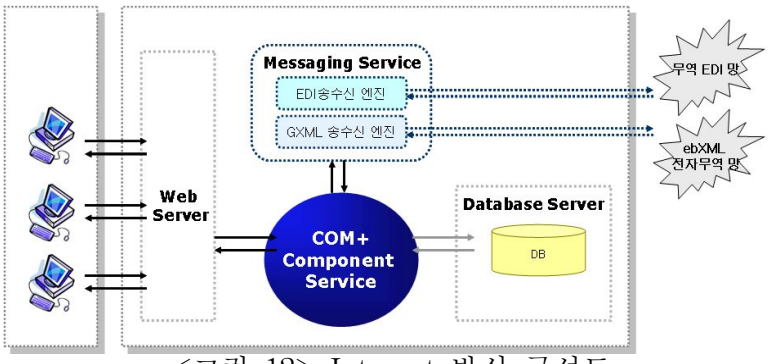
사용자 PC에서 Standalone 패키지 방식으로 구성한다. 이때에는 컴파일 옵션을 이용해 서버의 객체 Proxy가 아닌 로컬 PC에 컴파일된 DLL을 참조하도록 구현이 가능하다. 규모가 작은 무역업체의 경우 이러한 형태를 이용한다면 PC 1대로 업무가 가능하도록 설계하였다.




<그림 11> 단독형 모델 구성도

나.Intranet 형

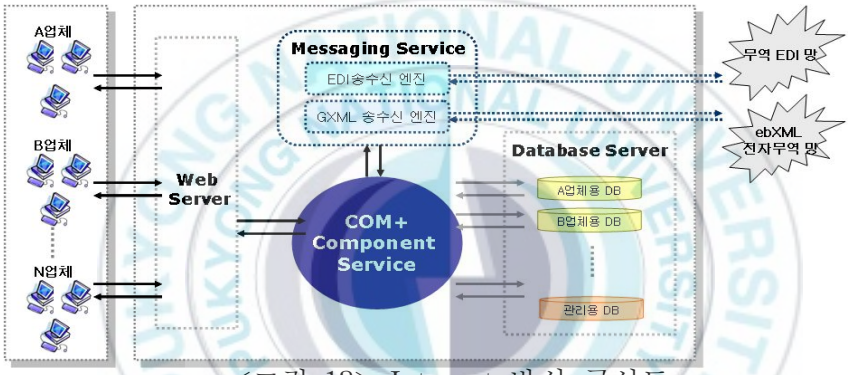
애플리케이션 서버를 중간에 두고 사용자 PC에서는 서버를 통해 데이터를 주고받는 방식으로 구성한다. 일반적인 C/S 형태와 유사하고 200명 이하 사용자를 수용할 수 있도록 하였다.




<그림 12> Intranet 방식 구성도

다. Internet 형

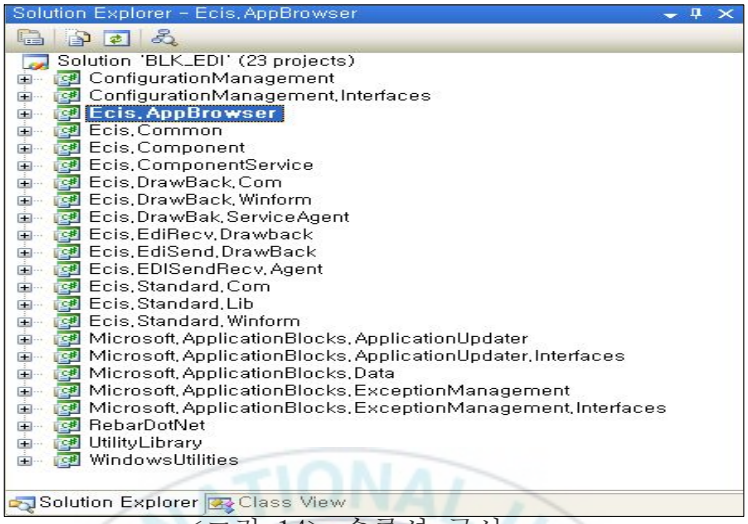
특정 사용자 그룹의 집합체의 형태로 ASP(Application Service Provider) 가 가능한 방식이다. EDI 관련 서비스의 일반적인 유형으로 본 논문에서는 Internet형 모델을 기반으로 구현하여 여러 업체에서 시험하고 리포팅 사용할 수 있는데에 초점을 맞추고자 하였다.




<그림 13> Internet 방식 구성도

### 4.3.2 솔루션 설계

스마트 클라이언트를 이용한 관세환급 EDI를 구축하기 위한 솔루션은 업무단위의 프로젝트로 구성되고 각각의 프로젝트는 비즈니스 로직, 별도의 시스템 상주를 위한 로직, 닷넷 Remoting 서비스 로직, 라이브러리 로직들로 구성된다. 또한 마이크로소프트사에서 제공하는 Enterprise Library를 포함한다.



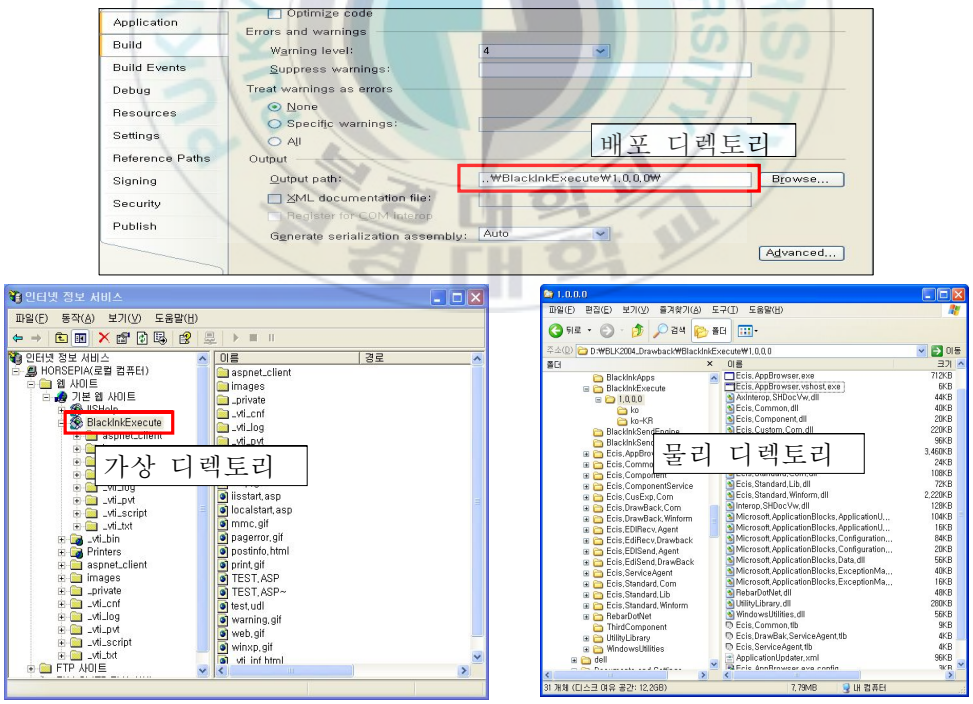

<그림 14> 솔루션 구성

<표 5> 프로젝트별 업무내용

| 프로젝트명 | 역할 |
|---|---|
| ConfigurationManagement 외 6개 프로젝트 | Microsoft에서 제공하는 Enterprise Library 프로젝트.(p.6 참조) |
| Ecis.AppBrowser | 구동용 브라우저를 포함하는 실행파일 |
| Ecis.Common | 환경설정, 컴포넌트, 예외처리, Remoting을 위한 베이스 클래스 구현 |
| Ecis.ComponentService | Remoting 서버용 윈도우즈 서비스 구현 |
| Ecis.DrawBack.Com | 관세환급 EDI 비즈니스 로직을 포함하는 Remoting 클래스 구현 |
| Ecis.DrawBack.Winform | 관세환급 EDI UI용 폼 구현 |
| Ecis.DrawBak.ServiceAgent | COM+ 구현을 위한 베이스 클래스 구현 |
| Ecis.DrawBak.ServiceAgent | COM+ 구현을 위한 베이스 클래스 구현 |
| Ecis.EdiRecv.Drawback 외 8개 프로젝트 | 업무 처리를 위한 클래스 구현 |

### 4.3.3 배포 설계

클래스는 상기 구분된 프로젝트 단위로 생성하고 컴파일된 프로젝트는 하나의 실행 파일과 프로젝트 단위의 DLL로 구성되어 저장된다. 실행 파일은 웹 브라우저를 통해 구동되고 이때 윈도우즈 폼으로 제작된 구동용 브라우저는 메뉴를 클릭하는 시점에 리플렉션(Reflection)에 의해 관련 DLL을 Web Server로부터 자동으로 사용자의 도메인에 다운로드 시킨 후 구동시키게 되는데 이러한 자동 배포 기능은 XML로 구성된 환경 파일과 엔터프라이즈 라이브러리를 통해 감추어져 있다. 따라서 컴파일된 파일들이 Web Server의 특정 디렉토리에 저장되어 있어야 하기 때문에 IIS에 BlackinkExecute 라는 가상디렉토리를 설정하고 컴파일시 해당 디렉토리에 모듈이 생성되도록 설정 하였다.

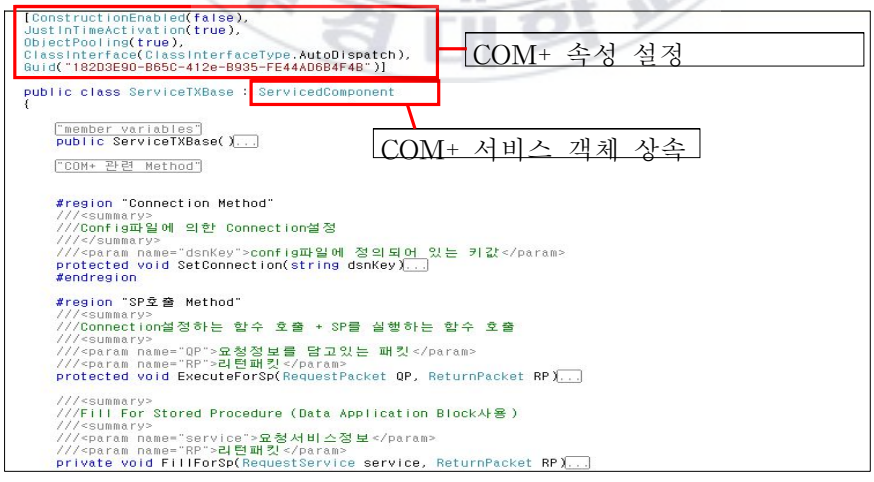



<그림 15> 배포 환경 구성

### 4.3.4 COM+ 설계

가. ServicedComponent 상속

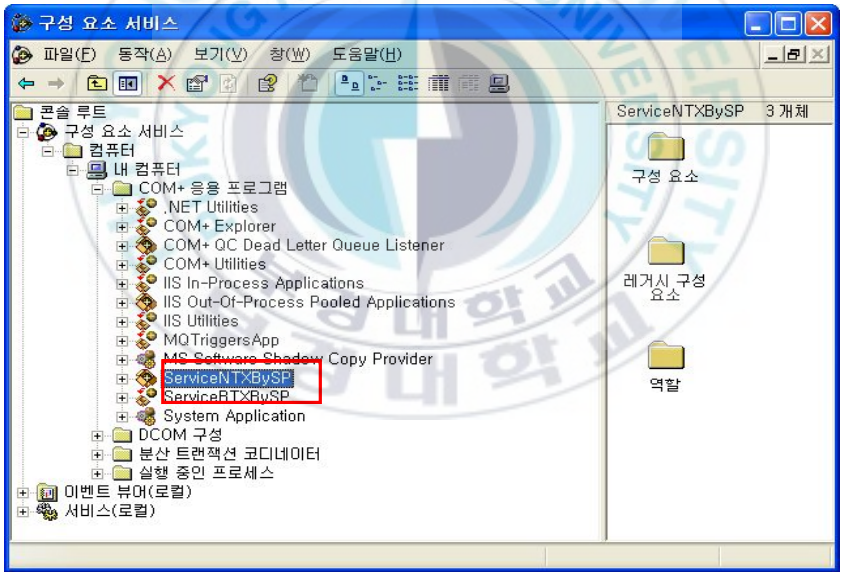
트랜젝션을 지원하기 위한 COM+와 트랜젝션을 지원하지 않는 COM+ 두 개의 형태를 서비스 베이스 클래스 ServiceTXBase.cs로 구성하고 Remoting 객체와 COM+간의 데이터 전달은 Ecis.DrawBack.ServiceAgent 프로젝트에서 베이스 클래스를 상속한 ServiceNTXBySP.cs, ServiceTXBySP.cs 클래스에서 담당한다. 이는 중,소규모 시스템 개발시 클래스 각각을 COM+ 서비스에 등록하여 관리하는 것보다 효율적인 것으로 판단되는데 많은 COM+ 객체를 구현하기보다 단순한 형태를 적용하여 COM+는 단지 비즈니스 클래스를 인스턴스화 하는 부분만 관장하고 파라미터 값을 전달하는 통로 역할만 할 수 있도록 구성하였다. ServiceTXBase.cs는 데이터 접속을 관장하고 쿼리나 프로시져를 호출하여 리턴된 결과를 Data Enterprise Library의 추상화 된 헬퍼클래스를 이용하여 데이터셋에 담는 기능을 포함하고 있다.




<그림 16> COM+ 구성 예

나. 구성요소 등록

COM+ 객체는 구성요소 서비스에 등록되어야 정상적으로 서비스가 가능한데 사용자에 의해 COM+ 객체가 생성되는 시점에 자동으로 구성요소 서비스에 등록된다. 이는 ApplicationActivation 속성이 기본적으로 Library로 설정이 되어 있어 ActivationOption을 Server로 설정하는 경우보다 간단하다. 최초 실행시 구성요소 서비스에 등록되기 위한 딜레이 타임이 존재한다. 여러 대의 분산 애플리케이션 서버를 구축해야 하는 경우 ActivationOption을 Server로 설정하여 시스템이 제공하는 프로세스에 의해 등록되는 형태로 전환이 가능하다.




<그림 17> COM+ 서비스 등록

### 4.3.5 닷넷 Remoting 설계

가. Remoting 객체 상속

닷넷 Remoting은 분산된 서버간의 데이터를 주고 받기 위해 구성된 진화된 RPC로 정의된다. 닷넷 Remoting 기능은 물리적으로 떨어진 시스템에 존재하는 기능을 사용하는 데에 소모되는 노력을 절감해준다. 단지 Remoting 객체인 MarshalByRefObject를 상속받고 메시지 전달 채널을 정의하면 된다. COM+ 객체를 생성하기 위해 상속받았던 ServicedComponent 객체 역시 MarshalByRefObject를 상속받아 구현된 객체이다. 따라서 COM+로 구성하게 되면 MarshalByRefObject를 상속받을 필요는 없으나 구성요소 서비스에 개별 등록되므로 관리가 용이하지 못하다. 그러한 관계로 비즈니스 객체들은 MarshalByRefObject를 상속받도록 하였다.

나. Remoting 클라이언트/서버 생성 및 URL 등록(채널)

Remoting 서버는 Remoting 객체를 외부로 노출시켜 클라이언트와 통신을 하기 위한 서버로 채널을 설정하고 URL을 등록하는 기능을 담당한다. 서버는 지속적인 서비스를 보장하기위해 윈도우즈 서비스로 등록하고 클라이언트는 별도의 서비스 없이 해당 URL만 찾아갈 수 있도록 클래스를 구성하였다. 클라이언트 서비스는 서버 서비스와 동일하게 윈도우즈 서비스로 구현이 가능하고 채널 및 URL 설정 XML 파일을 이용한다면 서비스 추가 및 채널, Port 변경의 작업들을 용이하게 처리할 수 있다.


<그림 18> Remoting 서비스 구성

### 4.3.6 모듈 설계

대부분의 개발자가 애플리케이션 구축시 중요시 생각하는 부분으로 생산성을 들 수 있다. 잘 짜여진 프레임을 기반으로 다양한 기능을 제공하는 함수와 라이브러리들을 조합하고, 추상화된 패턴들을 적용하여 개발시 반복되거나 비슷한 형태의 작업을 단순화 시키는 경우 생산성이 뒷받침 된다. 스마트 클라이언트 관세환급 EDI에서는 MVC에서 Model과 Controller에 해당되는 부분을 Enterprise Library를 이용해 구성하고 View에 해당하는 사용자용 UI 부분에 대한 수행속도와 생산성 향상을 위해 다중 데이터 셋을 이용하여 한번에 여러개의 결과셋을 전달받아 이용하도록 구성하였고, 이로부터 SQL 문장을 자동으로 생성하여 또한

다중 SQL을 전달하여 트랜젝션을 줄였다. <그림 19> 에서는 하나의 데이터셋에 특정 관세환급 신청서의 갑, 을, 병, 정지에 해당하는 데이터를 모두 불러와 조회, 수정, 삭제가 가능하도록 하였다.

```
private void cisTablesBinding(bool bStructure)
{
    CisTax5dak_nTx  _cisTax5dak_nTx     = (CisTax5dak_nTx)Activator.GetObject(typeof(CisTax5dak_nTx), Dic.Service.URLCisTax5dak_nTx);
    CisTemplate_nTx _cisTemplate_nTx    = (CisTemplate_nTx)Activator.GetObject(typeof(CisTemplate_nTx), Dic.Service.URLCisTemplate_nTx);


    SortedList srtCondition = new SortedList();
    srtCondition.Add("fldRST_KEY1", _sRST_KEY1);
    srtCondition.Add("fldRST_KEY2", _sRST_KEY2);

    if (DsTAX5DA.Tables.IndexOf("TAX5DA_K") >= 0)
        DsTAX5DA.Clear();

    _cisTax5dak_nTx.GridTBIEF001S(_sDB_NAME, ref DsTAX5DA, srtCondition, bStructure);

    DsTAX5DA.Tables["Table"].TableName  = "TAX5DA_K";
    DsTAX5DA.Tables["Table1"].TableName = "TAX5DA_U";
    DsTAX5DA.Tables["Table2"].TableName = "TAX5DA_B";
    DsTAX5DA.Tables["Table3"].TableName = "TAX5DA_J";

    DtTAX5DA_K          = DsTAX5DA.Tables["TAX5DA_K"];
    DtTAX5DA_U          = DsTAX5DA.Tables["TAX5DA_U"];
    DtTAX5DA_B          = DsTAX5DA.Tables["TAX5DA_B"];
    DtTAX5DA_J          = DsTAX5DA.Tables["TAX5DA_J"];

    // PrimaryKey 지정
    if (DtTAX5DA_K.PrimaryKey.Length == 0)
    {
        DataColumn[] dcKey = new DataColumn[2];
        dcKey[0] = DtTAX5DA_K.Columns["RST_KEY1"];
        dcKey[1] = DtTAX5DA_K.Columns["RST_KEY2"];
        DtTAX5DA_K.PrimaryKey = dcKey;
    }
```

일괄 Query 및 데이터셋 설정

데이터 셋 상호 참조키 설정

<그림 19> 데이터셋 구성

또한 데이터셋의 각각의 테이블에 저장된 데이터를 자동으로 SQL 문장으로 전환하여 배열에 담은 후 1회 트랜젝션으로 데이터베이스에 저장한다.

```
ArrayList arSql = new ArrayList();
UserLib.cisDataSetToSql(ref arSql, ref DtTAX5DA_K);
UserLib.cisDataSetToSql(ref arSql, ref DtTAX5DA_U);
UserLib.cisDataSetToSql(ref arSql, ref DtTAX5DA_B, "TmpTOT_TAX_AMT");
UserLib.cisDataSetToSql(ref arSql, ref DtTAX5DA_J);


CisDrawback_Tx _cisDrawback_Tx = (CisDrawback_Tx)Activator.GetObject(typeof(CisDrawback_Tx), Dic.Service.URLCisDrawback_Tx);


if (!_cisDrawback_Tx.cisStartTransaction(_sDB_NAME, arSql))
{
    MessageBox.Show(this, "데이타 저장중에 에러가 발생했습니다.", "저장", MessageBoxButtons.OK, MessageBoxIcon.Error);
    return;
}
else
{
    MessageBox.Show(this, "저장되었습니다.", "저장", Message
}
```

저장 Query 자동 생성

<그림 20> SQL 자동생성

나. 리플렉션

스마트 클라이언트에서 자동 업그레이드를 담당하는 기능은 리플렉션(Reflection)이 담당한다. 내부적으로 버전확인을 통해 새로운 버전이 Web Server에 존재하게되면 이를 다운로드 받아 구동하는 메커니즘을 가진다.

```
System.Windows.Forms.Form MyForm;
Assembly thatAssembly;
Object obj = new Object();
try
{
    thatAssembly  = System.Reflection.Assembly.Load(sDllName);

    if (args == null)
        obj = thatAssembly.CreateInstance(sDllName + "." + sFormName);
    else
        obj = thatAssembly.CreateInstance(sDllName + "." + sFormName, true,System.Reflection.BindingFlags.CreateInstance,
                null, args, null, null);
    if (obj == null)
    {
        MessageBox.Show(string.Format("{0} 호출에 실패하였습니다.", sDllName + "." + sFormName));
        return null;
    }
    MyForm = (System.Windows.Forms.Form) obj;
}
catch (Exception e)
{
    string a = e.Message;
    MessageBox.Show(string.Format("{0}.DLL 호출에 실패하였습니다.", sDllName));
    return null;
}
```

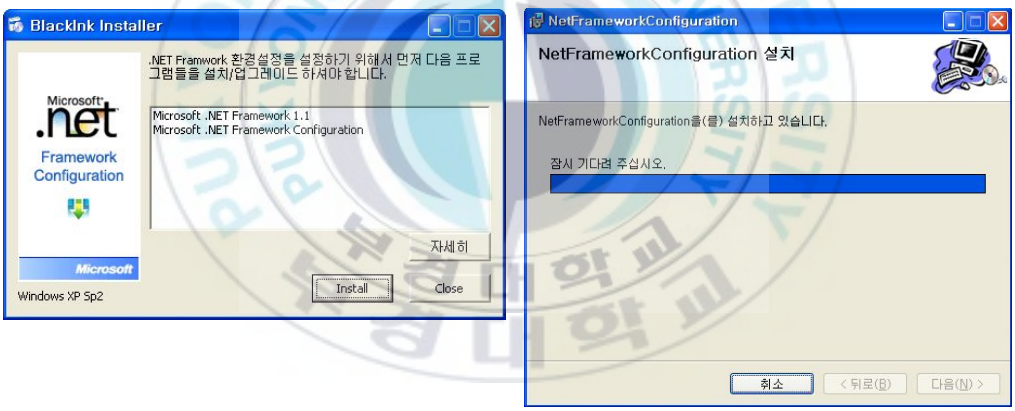
원격지 DLL 호출로 자동업그레이드 실행

<그림 21> 자동 업그레이드(Reflection) 구성

# 5. 스마트 클라이언트 관세환급 EDI 시스템의 구현

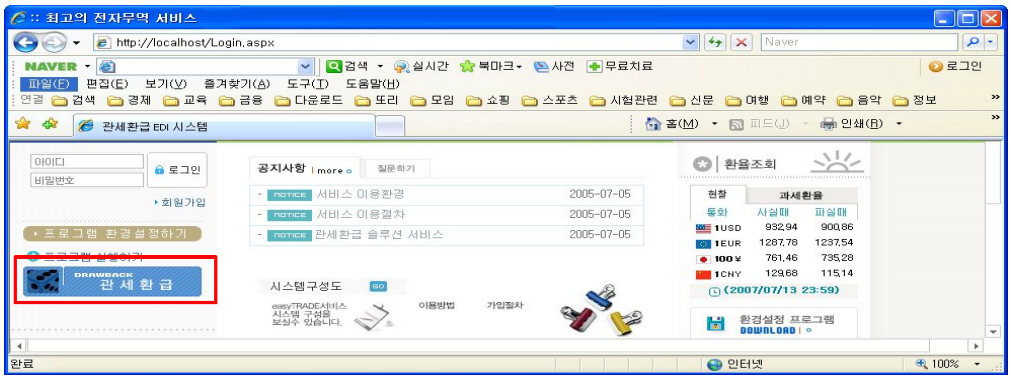
## 5.1 프로그램 구동

본 논문에서 구현한 스마트 클라이언트 관세환급 EDI 시스템은 Web 페이지를 통해 사용자가 프로그램을 구동시킬 수 있다. Web 페이지에서 프로그램 실행 버튼을 클릭하면 연결된 부트스트랩 파일은 닷넷 Framework 1.1 또는 2.0이 설치되어 있는지와 MDAC 2.7 이상이 설치되어 있는지, 닷넷 Configuration에 사용권한이 있는지를 판단하고 설치한다.



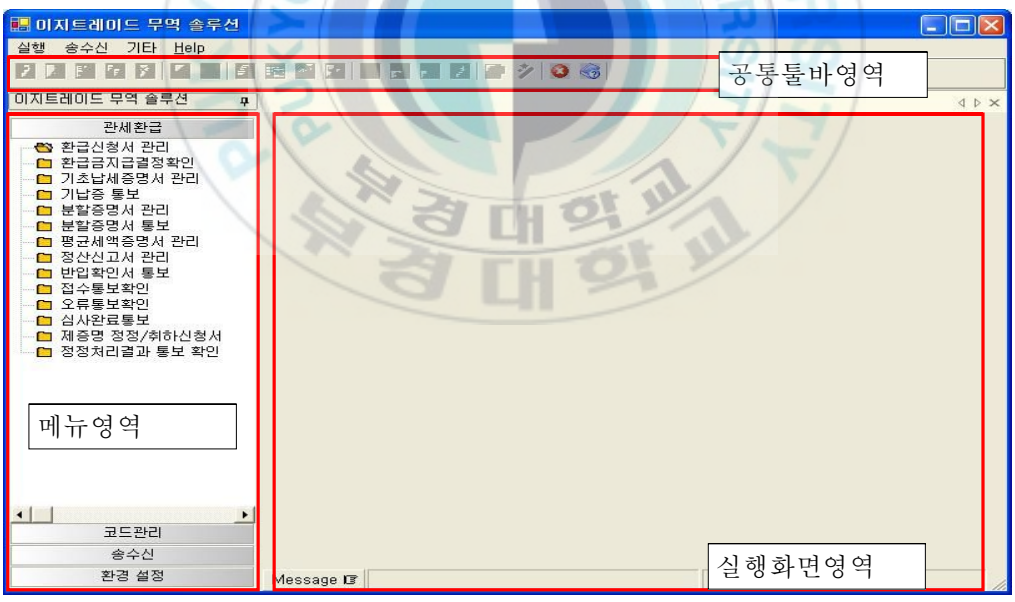

<그림22> 닷넷 Framework 설치

또한 사용자 로그인을 통해 적정한 사용자임을 판단하는 기능을 제공하도록 구현하였다. 사용자가 인증을 통과하여 애플리케이션 구동 버튼을 클릭하면 브라우저를 통해 애플리케이션이 호출된다.




<그림 23> 로그인 웹 페이지

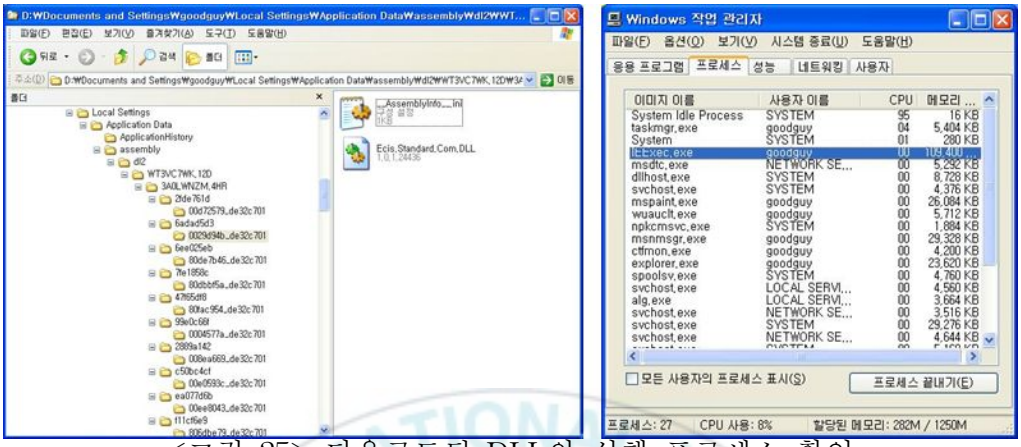
클릭 후 아래의 <그림 24>과 같은 전용 브라우저가 사용자 PC에서 구동되고 로그인 화면이 표시된다. 로그 인 후, 좌측의 메뉴를 클릭하면 해당하는 프로그램의 DLL 버전을 비교 확인한 후 업무 화면이 나타난다.




<그림 24> 관세환급 EDI 실행 화면

프로그램은 사용자 PC에 다운로드 된 후 닷넷 Framework의 Local Domain 하에서 실행되는데 <그림 25>에서 다운로드된 DLL을 확인 할 수

있다.

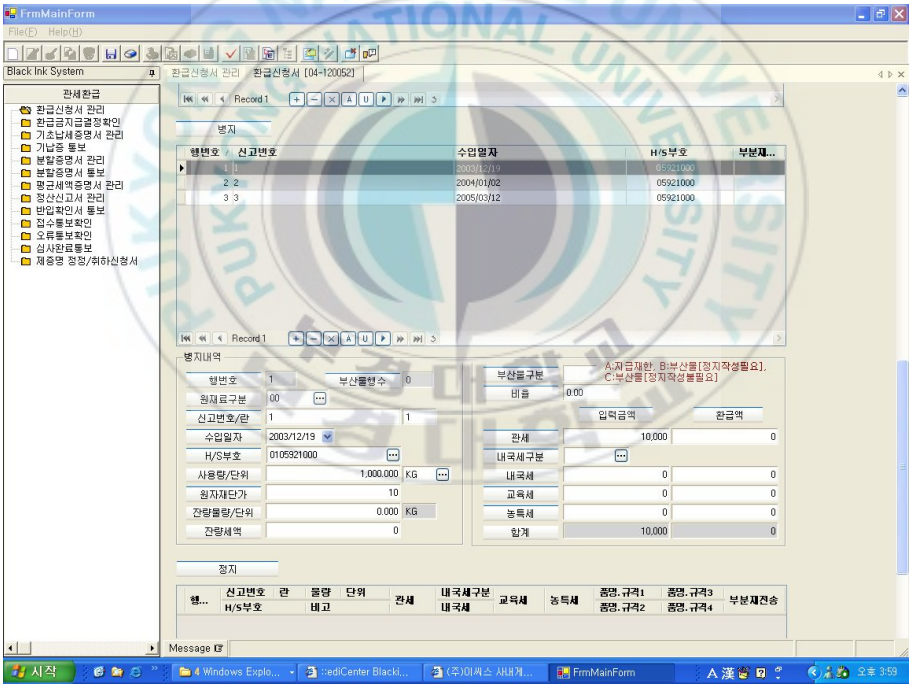


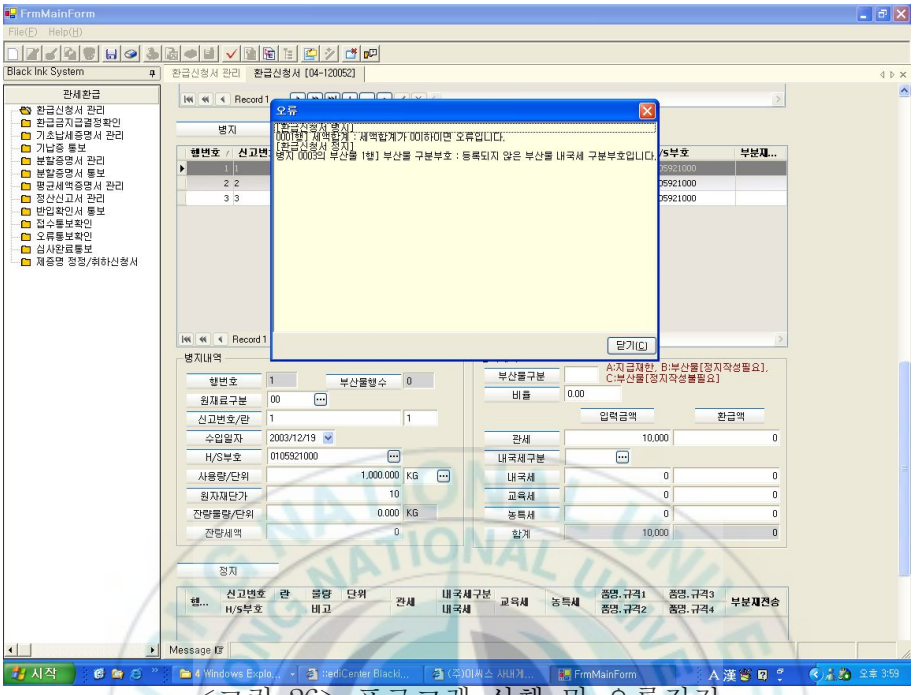
<그림 25> 다운로드된 DLL의 실행 프로세스 확인

## 5.2 화면 구성

스마트 클라이언트 관세환급 EDI가 실행되는 환경은 클라이언트 PC상이다. 화면만 보면 여타 클라이언트/서버 프로그램과 다른점이 없으나 클라이언트에는 Presentation Layer만 구성되어 있고 Business Layer 및 Data Access Layer는 원격지 서버의 COM+로 구성되어 있고 서버와 클라이언트를 연결하는 닷넷 Remoting 서버가 중간에서 연결자 역할을 하고 있다. 전용 브라우저의 좌측 메뉴를 클릭하면 프로그램이 실행된다. 이때 서버에 배포를 위한 DLL을 살펴 새로운 버전이 DLL이 빌드되어 있으면 로컬 PC로 다운로드 받아 실행된다. 다운로드된 DLL은 전역 어셈블리키에 의해 자동으로 로컬 PC의 GAC(Global Assembly Cache)에 자동 등록되고 상호간에 참조된다. 업무 실행 화면에서는 Master - Detail - SubDetail 형태의 구조를 가지는 데이터를 모두 표시하고 사용자가 각각의 테이블에 저장된 데이터를 수정하고 저장하면 일시에 각각의 테이블에 저

장한다. 따라서 Master 데이터에 연결된 Detail 및 SubDetail 데이터를 가져오기 위해 별도의 이벤트를 두어 처리하지 않는다. 요컨대 데이터를 가져오거나 저장하기 위한 트랜젝션은 단 1회에 그친다. 그 이외에도 클라이언트에서는 서버로 많은 조회를 요구하기 때문에 조회나 저장단계의 수행속도를 높이기 위해 메모리 데이터베이스 형태의 데이터셋을 이용하여 상호간의 Relation과 키 참조를 추가하였다. 이 경우 데이터셋 내부의 오류 발생에 대한 검증 부분 역시 저장 전 일괄 검증토록 구성하여 입력에 제약을 두지 않고 최종 입력된 데이터를 기준으로 오류사항을 알려 주도록 하였다.






<그림 26> 프로그램 실행 및 오류점검

## 5.3 EDI 전송

입력된 데이터들은 관세청에 전달되기 전에 다음과 같은 절차를 거친다.

<표 6> EDI 전송 절차

| 구분 | 성공 | 실패 |
|---|---|---|
| 1 단계 | 테이블에 저장된 신청서의 키를 송신준비 테이블에 저장 | |
| 2 단계 | 송신 엔진이 주기적으로 저장된 데이터를 읽어 Flat 파일로 변환. Winmate 전송디렉토리로 Flat 파일 이동 | |

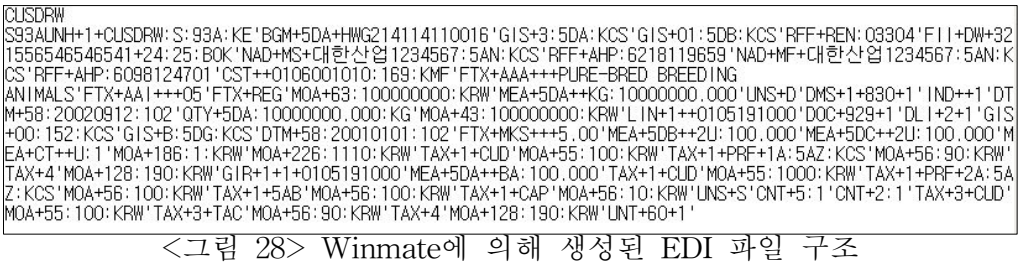
| | | |
|---|---|---|
| 3 단계 | WinMate에서는 전송 디렉토리에 있는 Flat File을 EDI 파일로 변환. | |
| 4 단계 | Winmate는 주기적으로 관세청에 EDI 파일 송신/수신 | 송/수신 로그 생성 |
| 5 단계 | WinMate에서 수신된 EDI 파일을 Flat 파일로 변환 | |
| 6 단계 | Flat 파일을 수신엔진이 주기적으로 변환하여 접수/오류통보등을 구분하여 데이터 생성/업데이트 | |

아래의 <그림 27>은 관세환급 업무 데이터를 읽어 생성한 Flat 파일[10]의 구조이다. 생성된 Flat 파일은 관세청으로 보내기 위해 EDI 파일로 변환해야 하고 Winmate 프로그램은 이러한 작업을 진행한다.

```
KCS0005          *                CUSDRW S   93A 5DBHWG214104110070
UNH
BGM 1       10
    5DB
    HWG214104110070
DTM 1       11
    126
    YYYYMMDD
    102
GIS 1       12
    3
    5DA
    KCS
GIS 2       12
    N
    5BH
    KCS
RFF 1       13
    DM
    01604
RFF 2       13
    LC
```

<그림 27> 송신 엔진에 의해 생성된 Flat 파일 구조

Winmate에 의해 EDI 파일로 변환된 Flat 파일은 아래 <그림 28>과 같은 구조를 가진다. XML이 EDI를 위한 데이터 구조로 적합함을 알 수 있고 현재 XML을 이용한 EDI 역시 구현되어 많은 분야에 이용되고 있다.




<그림 28> Winmate에 의해 생성된 EDI 파일 구조

## 5.4 효율성 분석

스마트 클라이언트 관세환급 EDI 시스템의 경우, 속도 부문에서 최장 5초 이내의 응답 시간이 소요되었고 DLL이 다운로드 되는 시점과 COM+ 서비스 등록시점에서 약 1~2초 정도의 딜레이 타임이 발생되었다. 이는 ActiveX를 다운받는 시간과 업무 처리 시간에 뒤지지 않은 결과를 보여주었고, 다운로드되어 저장된 어셈블리 캐쉬는 다음 트랜잭션 수행시 더 빠른 응답 속도를 가져왔다. 개발자는 HTML을 몰라도 인터넷 애플리케이션을 작성 할 수 있으며 개발자 PC에서 실행파일을 클릭하면 클라이언트/서버 애플리케이션 처럼 실행이 가능하여 개발 시간을 단축할 수 있었으며 단순 디버깅과 프로세스 디버깅이 가능하여 유지보수의 향상을 꾀할 수 있는 장점이 있었다.

<표 7> EDI 시스템 효율성 비교

| 구분 | 클라이언트/서버형 | 웹브라우저형 | 스마트 클라이언트형 |
|---|---|---|---|
| 편의성 | **상** | **하** | **상**<br>C/S형 대비: 동일<br>Web형 대비: 상승 |
| 처리속도 | **상** | **중** | **중** |

| | | | |
|---|---|---|---|
| | | (대량 처리-하) | C/S형 대비: 감소<br>Web형 대비: 동일 |
| 보안성 | **중**<br>(DB에만 의존) | **하**<br>(보안 정책 관리) | **상**<br>(형식화된 메시지 전달)<br>C/S형 대비: 동일<br>Web형 대비: 상승 |
| 실행/<br>다운로드 | **상** | **중**<br>(서버 사양에 좌우) | **중**<br>(다운로드, COM+<br>등록시 대기시간)<br>C/S형 대비: 일부감소<br>Web형 대비: 상승 |
| 장소제약<br>없음 | **하**<br>(프로그램 설치 필수) | **상** | **상**<br>C/S형 대비: 감소<br>Web형 대비: 일부상승 |
| 생산성 | **상** | **하**<br>(Html, JavaScript<br>모듈화 어려움) | **상**<br>C/S형 대비: 동일<br>Web형 대비: 상승 |
| 유지보수 | **하** | **중**<br>(Html, JavaScript,<br>Server Page) | **상**<br>C/S형 대비: 상승<br>Web형 대비: 상승 |
| 만족도<br>100%기준 | 76.2% | 57.1% | 90.5% |

관세환급 EDI 시스템을 사용하고 있는 무역업체 약20 개 업체를 대상으로 테스트 및 만족도를 조사한 결과 위 표에서 스마트 클라이언트 관세환급 EDI은 클라이언트/서버형 및 웹 브라우저형 EDI 시스템 사용자의 불만을 해소하기 위한 시스템으로 기존 시스템 대비 클라이언트/서버형은 약 13%, 웹 브라우저형은 약 33% 이상 향상된 효율을 가져온 것으로 파악되었고 특히, 클라이언트/서버형 애플리케이션에 익숙한 사용자가 웹 브라우저형 애플리케이션을 통한 업무처리시 공통적으로 느껴오던 조작의 어려움을 해소 할 수 있는 기능상의 장점을 가질 수 있었다.

# 6. 결론

본 논문인 스마트 클라이언트를 이용한 관세환급 EDI 시스템은 사용자 편의성에 가장 주안점을 두고 구축하였고 다음으로 생산성 및 유지보수, 성능(속도) 순으로 설정하였다. 사용자는 웹 브라우저를 사용하면서도 클라이언트/서버 형태의 다양한 기능이 구현되기를 원한다. 반면 단순화된 방식을 원한다. 보편화된 속도를 기반으로 하여 쉽고 간편한 기능이 제공되기를 원하고 있고 생산성이나 유지보수는 그다지 관심의 대상이 되지 않는다.

닷넷 Framework 2.0 에서는 Click Once라는 배포 자동화 기능을 통해 스마트 클라이언트를 위한 다운로드 환경 구성이 간단해졌고 일부 PC에서 Local Assembly Cache 영역에 다운로드가 정상적으로 이루어지지 않는 오류를 해결할 수 있었으나 여기서 소개하고자 하는 기능의 중점을 본문에 소개한 내용으로 제한 하였고 아울러 하위 버전에서 시작한 구현의 중도에 상위버전이 발표됨으로서 구 버전상의 시스템 구현이 되었기에 효용성에서 일부 아쉬운 점이 있으나 상위 버전에서도 수용가능한 호환성을 가지므로 큰 변경 없이 구현이 가능하리라 생각된다. 또한 본 구현에서는 기업 환경 전반에 걸쳐 사용될 수 있는 스마트 클라이언트 기술을 관세환급 EDI 시스템에 적용시켜 구현하였지만 앞으로 더욱 다양한 기업 및 개인 응용 애플리케이션에 적용되어 사용될 수 있을 것이다. 무엇보다도 Thin Client와 Thick Client를 아우르는 기술적인 요소들이 사용자가 요구하는 다양한 기능들을 충족시킬 수 있고 향 후 더욱 전문화된 기술 연구를 진행할 수 있으리라 예상된다.

# 참 고 문 헌


[1] 김영경, “웹사이트 개발 방법론 개선에 관한 연구”, 건국대학교 석사학위논문, 2005

[2] 박지호, “개별 데이터베이스와 웹 서버를 이용한 자료의 빈번한 업데이트의 불편함 해소”, 목원대학교 석사학위논문, 2005

[3] 박종현, “스마트 클라이언트 응용을 이용한 문서 시스템의 설계 및 구현”, 고려대학교 석사학위논문, 2006

[4] 박성국, “Curl중심의 X-Internet Web응용에 관한 연구”, 배재대학교 석사학위논문, 2005

[5] 한국관세연구소, “알기 쉬운 관세환급 실무”, 1995

[6] 관세청, “관세환급 안내”, http://www.customs.go.kr/kcsweb/user.tdf?a=common.HtmlApp&c=1001&page=/korean/html/kor/entry/tax/tax_04_01.html&mc=WWW_ENTRY_TAX_040

[7] 한국전자거래진흥원,“EDI 표준정보”, http://www.kiec.or.kr/edstandard/edistandard.asp, 2007

[8] 한국전자거래진흥원, “한국전자문서표준위원회”, http://www.kiec.or.kr/edstandard/keccommit.asp, 2007

[9] 이준학, “개방형 제어 플랫폼 구현을 위한 웹 서비스 분산환경 구축”, 중앙대학교 석사학위논문, 2006

[10] 김문하, “XML/EDI에 의한 기업간 정보 교환에 관한 연구”, 연세대학교 석사학위논문, 2001

[11] UN/ECE Trade Division, Part4 United Nations Rules for Electronic Data Interchange Transport, http://www.unece.org/trade/untdid/texts/unredi.htm,1995

[12] Microsoft, "스마트 클라이언트의 정의", http://www.microsoft.com/korea/msdn/smartclient/understanding/definition/, 2007

[13] Microsoft Pattern & Practices Enterprise Library , http://msdn2.microsoft.com/en-us/library/aa480453.aspx, 2007

# 감 사 의 글

더 견고한 전문 지식을 쌓고 사회 각 요소에서 빛을 발하며 활동하고 있는 여러 사람들과의 교류를 통해 유연한 사고와 경험을 쌓고 싶었던 즈음 대학원의 문을 두드리게 되었습니다.

생각보다 어려운 현실에서 중간 중간 그만두고 싶을 때 마다 처음을 생각했고, 주위의 여러 지인들의 너무나 감사한 도움으로 삶의 새로운 과제를 찾아 여러 방면의 도전을 시도하고자 했던 처음이 어느덧 마무리 해야 하는 시간에 이르렀습니다.

국내외 활발한 연구 활동으로 귀감이 되시고 간결하면서도 명확한 논리에 특유의 넉넉함으로 몇 번이고 주저앉고 싶었던 현실을 다독이며 격려해주신 박만곤 교수님께 먼저 감사의 말씀을 올립니다. 다소 부족함을 허물로 여기지 않고 오히려 이해와 배려로 채워주신 윤성대 교수님, 정목동 교수님, 진지하고 깊이 있는 지도를 해주셨던 여정모 교수님, 포기하지 않고 끝까지 마무리 할 수 있도록 용기와 격려를 아끼지 않았던 김수도, 장수미, 이소영 선생님께도 감사의 말씀을 드리며 특히, 힘든 현실을 자신의 일처럼 안타까워하며 끝까지 살펴주시던 박상기 선생님께 고마움을 전합니다.

새로운 목표를 위해 무작정 회사를 그만두었던 때에도, 학업을 위해 아빠의 역할을 미루어 두었던 때에도, 전혀 다른 삶의 방식과 문화로 인해 힘들어해야 했던 시간에도 끝까지 믿어주고 용기를 잃지 않았던 고마운 아내에게, 늘 먼곳에서 기도해주시고 응원해 주시는 양가 부모님께도 감사의 말씀을 드리며 이 논문을 바칩니다.